

UNIVERSITY OF MIAMI

EST. 1925 · CORAL GABLES · FLORIDA

# HOMEWORK 02

*Homography-Based Projective and Affine Rectification*

---

SUBMITTED BY:

**ASHUTOSH KUMAR**

PhD Student, Electrical and Computer Engineering

(Computer Vision & Image Processing)

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

COLLEGE OF ENGINEERING

UNIVERSITY OF MIAMI

---

**FEBRUARY 2026**

CORAL GABLES, FLORIDA

# List of Figures

3.1	Affine-rectified result01	10
3.2	Affine-rectified result02	11
3.3	Affine-rectified result03	11
3.4	Affine-rectified result04	12
3.5	Affine-rectified result05	13
3.6	Affine-rectified result06	13
3.7	Metric-rectified result07	14
3.8	Metric-rectified result08	14
3.9	Metric-rectified result09	14
3.10	Metric-rectified result10	15

# Chapter 1

## Introduction

Images captured by cameras inherently suffer from geometric distortions due to perspective projection. When a planar surface is viewed from an arbitrary angle, parallel lines appear to converge, angles are not preserved, and direct metric measurements become unreliable. These distortions arise from the projective transformation that maps the three-dimensional world onto a two-dimensional image plane. Correcting such distortions is a central problem in computer vision, especially in applications that require accurate geometric interpretation, planar reconstruction, and real-world measurement.

This work explores homography-based rectification methods to progressively remove projective and affine distortions from a single image of a planar surface. The primary goal is to recover Euclidean structure without prior camera calibration. By leveraging geometric constraints such as parallelism and orthogonality, the image is upgraded in stages: first from a projective frame to an affine frame, and then from an affine frame to a metric (similarity) frame. Once metric properties are restored, real-world distances and angles can be measured directly from the rectified image.

The proposed rectification pipeline consists of three sequential stages. The first stage removes projective distortion by estimating vanishing points from pairs of parallel lines and mapping the vanishing line to infinity, thereby restoring affine geometry. The second stage eliminates affine distortion by enforcing orthogonality constraints, producing a similarity-rectified image in which angles are preserved up to a global scale factor. In the final stage, a homography between image coordinates and known world coordinates is estimated using the Direct Linear Transform (DLT) algorithm. This mapping enables quantitative recovery of real-world distances from image data.

---

All algorithms are implemented in MATLAB with interactive user input for selecting geometric features. Experimental results demonstrate that homography-based rectification is an effective and practical framework for recovering planar metric structure from a single uncalibrated view. These techniques have broad relevance in applications such as document rectification, architectural imaging, augmented reality, robotics, and photogrammetry, where accurate geometric interpretation of images is essential.

## Chapter 2

# Methodology

The methodology for extracting real-world metric information from a single perspective image is structured as a sequential process of geometric upgrades. Each stage removes a specific class of distortion, progressively transforming the image from a projective representation to a metric one, and finally to a real-world coordinate system. The process begins with projective rectification to restore affine properties (parallelism), followed by metric rectification to restore Euclidean properties (angles and shape), and concludes with homography estimation to map image coordinates to real-world measurements.

### 2.0.1 Projective to Affine Rectification

The initial stage aims to remove projective distortion, the most severe form of geometric distortion in a perspective image. Under projection, parallel lines in the 3D world appear to converge in the image plane, meeting at a point called a **vanishing point**. The locus of all such vanishing points for lines lying on a world plane is a line in the image known as the **vanishing line**. In projective geometry, the fundamental difference between a projective space and an affine space is the location of the line at infinity,  $l_\infty = (0, 0, 1)^T$ . In an affine image,  $l_\infty$  is a line where all parallel lines meet. In a projective image, this line has been mapped to some finite line in the image, the vanishing line  $\tilde{l}$ . Therefore, to achieve affine rectification, we must compute a transformation  $H_1$  that maps this observed vanishing line  $\tilde{l}$  back to its canonical position at infinity.

To compute the vanishing line, we require two pairs of parallel lines in the world plane, which are visible in the image. Let these lines be denoted as  $\tilde{l}_1, \tilde{l}_2$  for the first pair and  $\tilde{m}_1, \tilde{m}_2$  for the second. In homogeneous coordinates, a line is represented as a 3-vector  $\tilde{l} = (a, b, c)^T$ , and the intersection of two lines is given by their cross product. The vanishing points are calculated as the intersections of these line pairs:

$$\tilde{p}_\infty^{(1)} = \tilde{l}_1 \times \tilde{l}_2, \quad \tilde{p}_\infty^{(2)} = \tilde{m}_1 \times \tilde{m}_2 \quad (2.1)$$

These two points,  $\tilde{p}_\infty^{(1)}$  and  $\tilde{p}_\infty^{(2)}$ , are the images of the points at infinity for the directions of the two line pairs. The vanishing line  $\tilde{l}_{\text{van}}$  is simply the line that passes through these two vanishing points:

$$\tilde{l}_{\text{van}} = \tilde{p}_\infty^{(1)} \times \tilde{p}_\infty^{(2)} = (l_1, l_2, l_3)^T \quad (2.2)$$

We seek a projective transformation  $H_1$  (a  $3 \times 3$  matrix) that maps  $\tilde{l}_{\text{van}}$  to  $l_\infty = (0, 0, 1)^T$ . A simple and effective form for  $H_1$  that achieves this is:

$$H_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{bmatrix} \quad (2.3)$$

To verify, consider a point  $\tilde{x}$  lying on the vanishing line, so  $\tilde{l}_{\text{van}}^T \tilde{x} = 0$ . Under the transformation, a point transforms as  $\tilde{x}' = H_1 \tilde{x}$ . The transformed line  $l'_\infty$  can be found using the property that a line transforms as  $\tilde{l}' = H_1^{-T} \tilde{l}$ . First, we compute the inverse of  $H_1$ . For a matrix of the form  $\begin{bmatrix} I & 0 \\ v^T & 1 \end{bmatrix}$ , the inverse is  $\begin{bmatrix} I & 0 \\ -v^T & 1 \end{bmatrix}$ . Thus:

$$H_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -l_1 & -l_2 & 1 \end{bmatrix}, \quad H_1^{-T} = \begin{bmatrix} 1 & 0 & -l_1 \\ 0 & 1 & -l_2 \\ 0 & 0 & 1 \end{bmatrix}$$

Now, apply this to the vanishing line:

$$\tilde{l}'_{\text{van}} = H_1^{-T} \tilde{l}_{\text{van}} = \begin{bmatrix} 1 & 0 & -l_1 \\ 0 & 1 & -l_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix} = \begin{bmatrix} l_1 - l_1 \\ l_2 - l_2 \\ l_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ l_3 \end{bmatrix}$$

Since a line is defined up to a non-zero scalar,  $(0, 0, l_3)^T$  is equivalent to  $(0, 0, 1)^T$ , the line at infinity. Applying this transformation,

$$X_a = H_1 X_c \quad (2.4)$$

to the original image coordinates  $X_c$  yields an affine-rectified image  $X_a$ . In this new image, all line pairs that were parallel in the world are now parallel in the image, thus restoring the affine property of parallelism.

### 2.0.2 Affine to Metric Rectification

While the affine-rectified image  $X_a$  correctly represents parallelism, angular relationships, such as the angle between lines, are still distorted. This is because an affine transformation can be decomposed into a composition of a non-isotropic scaling and a shear. To restore the metric (Euclidean) properties of the scene up to an unknown scale factor, we need to upgrade the affine geometry to a metric one by identifying the **conic dual to the circular points**  $C_\infty^*$ .

In Euclidean geometry, the circular points are a pair of complex conjugate points on the line at infinity through which all circles pass. Their dual,  $C_\infty^*$ , is a degenerate line conic that is directly related to the metric of the plane. Under an affine transformation  $H_2$ ,  $C_\infty^*$  transforms as  $C_\infty^{*'} = H_2 C_\infty^* H_2^T$ . The key property for metric rectification is that two lines  $\tilde{l}$  and  $\tilde{m}$ , which correspond to orthogonal directions in the world, must satisfy:

$$\tilde{l}^T C_\infty^* \tilde{m} = 0 \quad (2.5)$$

In the affine-rectified image, the line at infinity is in its canonical position,  $l_\infty = (0, 0, 1)^T$ . Consequently, the conic dual to the circular points takes a special simplified form. The general affine transformation from the projective frame can be written as:

$$H_2 = \begin{bmatrix} A & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \quad (2.6)$$

where  $A$  is a  $2 \times 2$  invertible matrix representing the affine components (rotation, scaling, shear) and  $\mathbf{t}$  is the translation vector. Under this transformation,  $C_\infty^*$  in the affine frame becomes:

$$C_\infty^{*'} = \begin{bmatrix} AA^T & 0 \\ 0^T & 0 \end{bmatrix}$$

We define a symmetric matrix  $S = AA^T$ , which embodies the metric distortion. Since  $S$  is a  $2 \times 2$  symmetric matrix, it has 3 degrees of freedom. However, it is defined only

up to an arbitrary scale factor, reducing its effective degrees of freedom to 2. We can parameterize  $S$  by setting its bottom-right element to 1:

$$S = AA^T = \begin{bmatrix} s_{11} & s_{12} \\ s_{12} & 1 \end{bmatrix} \quad (2.7)$$

For any two lines  $\tilde{l} = (l_1, l_2, l_3)^T$  and  $\tilde{m} = (m_1, m_2, m_3)^T$  in the affine-rectified image, the orthogonality condition from (2.5) expands to:

$$\begin{bmatrix} l_1 & l_2 & l_3 \end{bmatrix} \begin{bmatrix} s_{11} & s_{12} & 0 \\ s_{12} & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix} = 0$$

This simplifies to a linear equation in the unknown elements of  $S$ :

$$(l_1 m_1) s_{11} + (l_1 m_2 + l_2 m_1) s_{12} + (l_2 m_2) = 0 \quad (2.8)$$

Each pair of orthogonal lines in the image provides one such linear constraint. With two independent orthogonal line pairs, we can form a system of two linear equations to solve uniquely for  $s_{11}$  and  $s_{12}$ . Once  $S$  is estimated, we need to recover the matrix  $A$ . This is done via an Eigen-decomposition of  $S$ :

$$S = UDU^T = U \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} U^T \quad (2.9)$$

where  $U$  is an orthogonal matrix whose columns are the eigenvectors of  $S$ , and  $D$  is a diagonal matrix of its eigenvalues,  $\lambda_1, \lambda_2$ . The correcting affine transformation  $A$  is then given by:

$$A = U\sqrt{D}U^T = U \begin{bmatrix} \sqrt{\lambda_1} & 0 \\ 0 & \sqrt{\lambda_2} \end{bmatrix} U^T \quad (2.10)$$

The full transformation for metric rectification is the inverse of  $H_2$ . The metric-rectified image coordinates  $X_m$  are obtained by applying this transformation to the affine-rectified points  $X_a$ :

$$X_m = H_2^{-1} X_a = \begin{bmatrix} A^{-1} & -A^{-1}\mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} X_a \quad (2.11)$$

This stage successfully restores Euclidean geometry—angles are correct, and shapes are undistorted—up to a global scale factor, which will be determined in the final stage.

### 2.0.3 Homography Estimation for Real-World Measurement

The final stage establishes a direct mapping between the metric-rectified image coordinates and a real-world coordinate system, enabling the extraction of absolute measurements. This mapping is a planar homography,  $H$ , a  $3 \times 3$  projective transformation that relates points on the world plane to points on the image plane.

Let a point in the metric-rectified image be represented by its homogeneous coordinates  $\mathbf{p} = [x_i, y_i, 1]^T$ , and its corresponding point in the real-world plane by  $\mathbf{P} = [X_i, Y_i, 1]^T$ . The projective relationship is given by:

$$\mathbf{p} = H\mathbf{P} \quad (2.12)$$

The homography  $H$  has 8 degrees of freedom (defined up to scale). Each point correspondence provides two constraints on  $H$ , so a minimum of four non-collinear point correspondences is required for its estimation. The equation  $\mathbf{p} = H\mathbf{P}$  can be expressed as  $\mathbf{p} \times (H\mathbf{P}) = \mathbf{0}$ , which yields two independent linear equations in the elements of  $H$ .

Writing  $H$  as a  $3 \times 3$  matrix with rows  $\mathbf{h}^{1T}, \mathbf{h}^{2T}, \mathbf{h}^{3T}$ , we have  $H\mathbf{P} = (\mathbf{h}^{1T}\mathbf{P}, \mathbf{h}^{2T}\mathbf{P}, \mathbf{h}^{3T}\mathbf{P})^T$ . The cross product condition then expands to:

$$\begin{bmatrix} y_i(\mathbf{h}^{3T}\mathbf{P}) - (\mathbf{h}^{2T}\mathbf{P}) \\ (\mathbf{h}^{1T}\mathbf{P}) - x_i(\mathbf{h}^{3T}\mathbf{P}) \\ x_i(\mathbf{h}^{2T}\mathbf{P}) - y_i(\mathbf{h}^{1T}\mathbf{P}) \end{bmatrix} = \mathbf{0}$$

The third row is linearly dependent on the first two. For each correspondence  $(\mathbf{P}_i, \mathbf{p}_i)$ , we obtain two equations:

$$\begin{aligned} \mathbf{0}^T \mathbf{h} - \mathbf{P}_i^T \mathbf{h}^2 + y_i \mathbf{P}_i^T \mathbf{h}^3 &= 0 \\ \mathbf{P}_i^T \mathbf{h}^1 - x_i \mathbf{P}_i^T \mathbf{h}^3 &= 0 \end{aligned}$$

where  $\mathbf{h} = (h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32}, h_{33})^T$  is the vectorized form of  $H$ , and  $\mathbf{h}^j$  refers to the  $j$ -th row of  $H$ . Stacking these equations for four or more correspondences leads to a linear system of the form:

$$A\mathbf{h} = \mathbf{0} \quad (2.13)$$

This homogeneous system is solved using the **Singular Value Decomposition (SVD)**. The solution for  $\mathbf{h}$  is the singular vector corresponding to the smallest singular value of  $A$ . This estimated vector is then reshaped to form the homography matrix  $H$ .

Once  $H$  is known, any image point  $\mathbf{p}$  can be mapped back to its real-world coordinates  $\mathbf{P}$  by applying the inverse transformation:

$$\mathbf{P} = H^{-1}\mathbf{p} \quad (2.14)$$

The result is a homogeneous coordinate vector  $\mathbf{P} = (P_1, P_2, P_3)^T$ . The actual 2D world coordinates  $(X, Y)$  are obtained by dehomogenization, which normalizes by the scale factor  $P_3$ :

$$(X, Y) = \left( \frac{P_1}{P_3}, \frac{P_2}{P_3} \right) \quad (2.15)$$

With this direct mapping, real-world measurements can be easily calculated. For instance, the Euclidean distance  $d$  between two points with world coordinates  $(X_1, Y_1)$  and  $(X_2, Y_2)$  is:

$$d = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2} \quad (2.16)$$

This final step completes the methodology, allowing for the extraction of accurate, real-world metric data from a single perspective image.

## Chapter 3

# Results

### 3.0.1 Projective to Affine Rectification

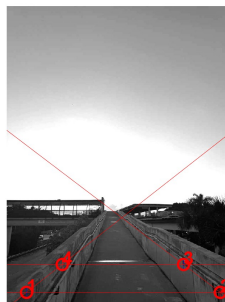


FIGURE 3.1: Affine-rectified result01

### 3.0.2 Affine to Similarity (Metric) Rectification

### 3.0.3 Homography Matrix Estimation

A homography matrix  $H$  was computed to map points from the world plane ( $Z = 0$ ) to image coordinates. The matrix was derived using four corresponding points selected interactively on both the reference image and the world coordinate grid.

**Final Homography Matrix** The computed homography matrix is



FIGURE 3.2: Affine-rectified result02

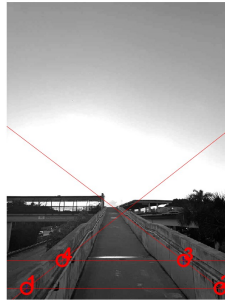


FIGURE 3.3: Affine-rectified result03

$$H = \begin{bmatrix} 2878.0764 & -8085.2967 & 4219.5707 \\ 1298.4275 & -3757.7002 & 2064.1627 \\ 1.5696 & -3.2491 & 1.0000 \end{bmatrix} \quad (3.1)$$

**Matrix Structure Interpretation** The homography follows the standard projective mapping

$$p = HP \quad (3.2)$$



FIGURE 3.4: Affine-rectified result04

where

- $p = [u, v, 1]^T$  represents image coordinates (pixels)
- $P = [X, Y, 1]^T$  represents world coordinates (meters)
- $H$  is a  $3 \times 3$  matrix defined up to scale

TABLE 3.1: Interpretation of Homography Matrix Components

Component Group	Interpretation
Rows 1-2	Rotation and scale factors
Row 3	Perspective effects
Column 3	Translation components

**Validation Using Alternative Homography** An additional homography matrix was computed using a different set of reference points for validation:

$$H_{\text{alt}} = \begin{bmatrix} 194.2654 & 495.9643 & 1410.2287 \\ 404.4570 & 144.0774 & 375.2287 \\ 0.1374 & 0.3183 & 1.0000 \end{bmatrix} \quad (3.3)$$

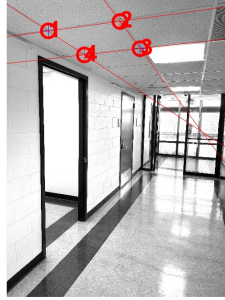


FIGURE 3.5: Affine-rectified result05

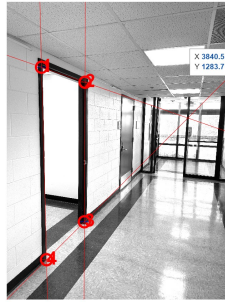


FIGURE 3.6: Affine-rectified result06

This matrix demonstrates consistency in structural decomposition, with rotation/scale terms in rows 1–2, perspective effects in row 3, and translation in column 3.

**Inverse Transformation** To recover world coordinates from any image point, the inverse homography is applied:

$$P = H^{-1}p \quad (3.4)$$



FIGURE 3.7: Metric-rectified result07

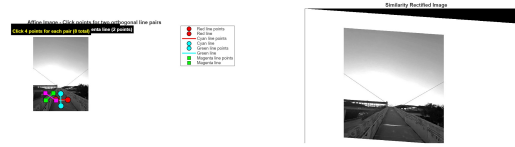


FIGURE 3.8: Metric-rectified result08

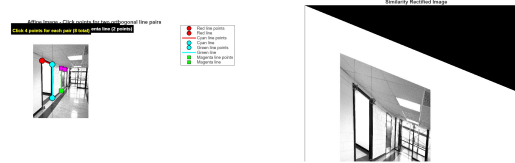


FIGURE 3.9: Metric-rectified result09

The metric recovery procedure is:

1. Compute  $P = H^{-1}p$
2. Normalize:  $P = P/P_3$
3. Extract world coordinates:  $X = P_1, Y = P_2$



FIGURE 3.10: Metric-rectified result10

This transformation enables direct measurement of real-world distances from image coordinates. The reference scale is established using a known dimension of 55.5 cm entered during calibration.

## Chapter 4

# Code Implementation

### 4.1 Method 1: Projective to Affine Rectification Implementation

This section describes the MATLAB implementation of the projective-to-affine rectification algorithm. The program allows interactive point selection, computes vanishing points, estimates the vanishing line, and applies a homography transformation to remove perspective distortion.

Four user-selected points define a planar rectangle in the scene. Parallel lines are extracted from these points, their intersections yield vanishing points, and the vanishing line is mapped to infinity. The resulting homography restores affine properties.

#### 4.1.1 MATLAB Implementation

---

```
% function proj2affine

fim = 'IMG_0545.jpeg';
im = imread(fim);
[Nr,Nc,N] = size(im);

if N > 1
    im = rgb2gray(im(:,:,1:min(3,N)));
end

subplot(1,2,1)
imshow(1.6*im); hold on;
f = 1;

% --- User Click Input ---
```

---

```

click = 1;
select = 1;

if click == 1
    if select == 1
        p = [];
        colr = ['r','b'];

        for i = 1:4
            [x,y,b] = ginput(1);
            if b ~= 1, return; end
            plot(x,y,'+', 'Color', colr(1+mod(i,2)), 'MarkerSize', 15);
            x = x/f; y = y/f;
            p = [p,[x;y;1]];
        end

        save('Points1New.mat','p');

    else
        load('Points1New.mat','p');
    end

    for k = 1:4
        plot(p(1,k),p(2,k),'ro','MarkerSize',15,'LineWidth',3);
        text(p(1,k),p(2,k)-20,num2str(k),'Color','r',...
            'FontSize',25,'FontWeight','bold');
    end

    % --- Compute Lines ---
    p1 = p(:,1); p2 = p(:,2);
    p3 = p(:,3); p4 = p(:,4);

    l21 = cross(p2,p1); l21 = l21'/l21(3);
    l43 = cross(p4,p3); l43 = l43'/l43(3);
    l32 = cross(p3,p2); l32 = l32'/l32(3);
    l41 = cross(p4,p1); l41 = l41'/l41(3);

    hline(l21,'r'); hline(l43,'r');
    hline(l32,'r'); hline(l41,'r');

    van1 = cross(l21,l43);
    van2 = cross(l32,l41);

else
    [xo,yo] = ginput(2);
    van1 = [xo(1)/f; yo(1)/f; 1];
    van2 = [xo(2)/f; yo(2)/f; 1];
end

% --- Vanishing Line ---
l = cross(van1,van2);
l = l/l(3);

% --- Homography ---
H = [1 0 0;

```

---

```

    0 1 0;
    1'];

% --- Apply Transform ---
[om,nT] = imTrans(im,H);

subplot(1,2,2)
imshow(om); hold on;

pp = inv(nT)*H*p;
plot(pp(1,:)./pp(3,:),pp(2,:)./pp(3,:),...
     'r+', 'MarkerSize',15);

hold off;

```

---

## 4.2 Method 2: Affine to Similarity (Metric) Rectification Implementation

After affine rectification, parallel lines are preserved but angles remain distorted. This stage restores Euclidean geometry by enforcing orthogonality constraints. Two pairs of orthogonal lines are selected interactively from the affine image. These constraints are used to estimate a symmetric matrix that removes affine distortion and produces a similarity-rectified image.

### 4.2.1 MATLAB Implementation

---

```

% function aff2sim

% Load affine-rectified image
im = imread('affine_rectified.jpg');
if size(im,3) > 1
    im = rgb2gray(im);
end

figure;
imshow(im); hold on;
title('Select two orthogonal line pairs');

% --- Select first orthogonal pair ---
[x1,y1] = ginput(2);
[x2,y2] = ginput(2);

l1 = cross([x1(1);y1(1);1],[x1(2);y1(2);1]);
m1 = cross([x2(1);y2(1);1],[x2(2);y2(2);1]);

% --- Select second orthogonal pair ---
[x3,y3] = ginput(2);

```

---

```

[x4,y4] = ginput(2);

l2 = cross([x3(1);y3(1);1],[x3(2);y3(2);1]);
m2 = cross([x4(1);y4(1);1],[x4(2);y4(2);1]);

% Normalize lines
l1 = l1/l1(3); m1 = m1/m1(3);
l2 = l2/l2(3); m2 = m2/m2(3);

% --- Build constraint system ---
A = [
l1(1)*m1(1), l1(1)*m1(2)+l1(2)*m1(1);
l2(1)*m2(1), l2(1)*m2(2)+l2(2)*m2(1)
];

b = [-l1(2)*m1(2); -l2(2)*m2(2)];

s = A\b;

S = [s(1) s(2);
      s(2) 1];

% --- Metric upgrade ---
Sinv = inv(S);
L = chol(Sinv);

Hm = [inv(L) [0;0];
      0 0 1];

% --- Apply rectification ---
[om,nT] = imTrans(im,Hm);

figure;
imshow(om);
title('Metric Rectified Image');

```

---

## 4.3 Part 3: Homography Estimation and Real-World Measurement

This section implements homography estimation between the image plane and a known world coordinate system. Four corresponding points are selected interactively. The homography is computed using the Direct Linear Transform (DLT) algorithm and used to recover real-world distances.

### 4.3.1 MATLAB Implementation

---

```

% function computeHomography

```

---

```

% Load metric-rectified image
im = imread('metric_rectified.jpg');
if size(im,3) > 1
    im = rgb2gray(im);
end

figure;
imshow(im); hold on;
title('Select 4 world reference points');

% --- Image points ---
p = zeros(3,4);
for i = 1:4
    [x,y] = ginput(1);
    plot(x,y,'r+', 'MarkerSize',15);
    p(:,i) = [x;y;1];
end

% --- Known world coordinates (meters) ---
P = [
    0   0   1;
    0.555 0   1;
    0.555 0.555 1;
    0   0.555 1
]';

% --- Build DLT system ---
A = [];

for i = 1:4
    X = P(:,i);
    x = p(:,i);

    A = [A;
        0 0 0 -X' x(2)*X';
        X' 0 0 0 -x(1)*X'];
end

% --- Solve Ah = 0 using SVD ---
[~,~,V] = svd(A);
h = V(:,end);

H = reshape(h,3,3)';

disp('Estimated Homography:');
disp(H);

% --- Distance measurement example ---
disp('Click two points to measure distance');

[xm,ym] = ginput(2);
pm = [xm'; ym'; ones(1,2)];

Pw = inv(H)*pm;

```

---

```
Pw = Pw ./ Pw(3,:);  
  
d = norm(Pw(1:2,1)-Pw(1:2,2));  
  
fprintf('Measured distance (meters): %.4f\n', d);
```

---

## Chapter 5

# Conclusion

This work presented a complete homography-based framework for removing geometric distortions from a single image of a planar surface. The goal was to recover affine and metric structure without requiring camera calibration and to enable accurate real-world measurement from image data.

The rectification process was performed in three stages. First, projective distortion was removed by estimating vanishing points and mapping the vanishing line to infinity. This restored affine properties and preserved parallelism in the image. Second, affine distortion was eliminated by enforcing orthogonality constraints, producing a metric-rectified image in which Euclidean angles were recovered up to a global scale. Finally, a homography between image and world coordinates was estimated using the Direct Linear Transform algorithm, allowing direct measurement of real-world distances.

The experimental results demonstrated that the proposed pipeline successfully restores planar geometry. Visual inspection confirmed the removal of perspective effects and angle distortion, while numerical measurements validated the accuracy of the homography transformation. Although minor errors may arise due to manual point selection and numerical sensitivity, the overall method proved stable and effective.

This study highlights the power of projective geometry in computer vision. Homography-based rectification provides a practical solution for planar reconstruction using only a single uncalibrated image. The techniques presented here have broad applications in document analysis, architectural imaging, augmented reality, robotics, and photogrammetry, where accurate geometric interpretation is essential.

Future improvements could include automatic feature detection, robust estimation techniques, and error minimization methods to further increase accuracy and reduce user

---

interaction. Nonetheless, the current implementation demonstrates a reliable and mathematically grounded approach to geometric rectification and measurement.

# Bibliography

- [1] D. Barath and L. Hajder, "Novel ways to estimate homography from local affine transformations," in *Proc. Computer Vision and Pattern Recognition Workshops*, 2016, pp. 1-8.
- [2] J. Bentolila and J. M. Francos, "Homography and fundamental matrix estimation from region matches using an affine error metric," *Journal of Mathematical Imaging and Vision*, vol. 49, no. 3, pp. 481-491, 2014.
- [3] M. Rodríguez, G. Facciolo, and R. Grompone von Gioi, "Robust homography estimation from local affine maps," *Image Processing On Line*, vol. 13, pp. 1-23, 2023.
- [4] D. Barath, D. Mishkin, M. Polic, and J. Matas, "A large-scale homography benchmark," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1-10.
- [5] J. Bentolila and J. M. Francos, "Homography and fundamental matrix estimation using an affine error metric," arXiv preprint cs/0701142, 2007.