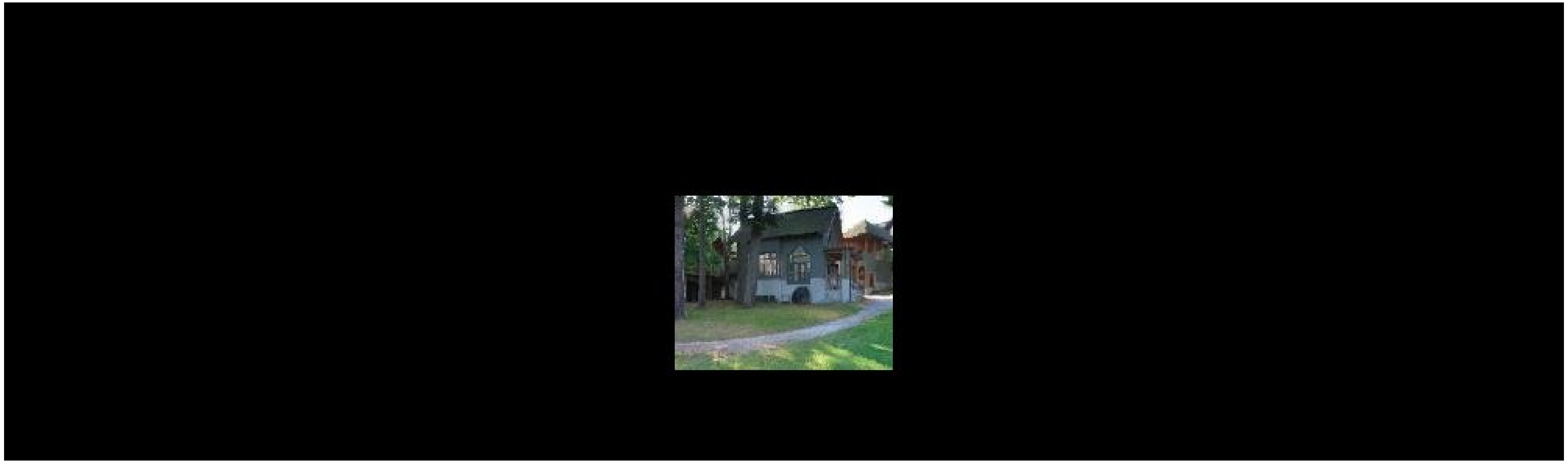
Introduction

- Are you getting the whole picture?
  - Compact Camera FOV = 50 x 35°



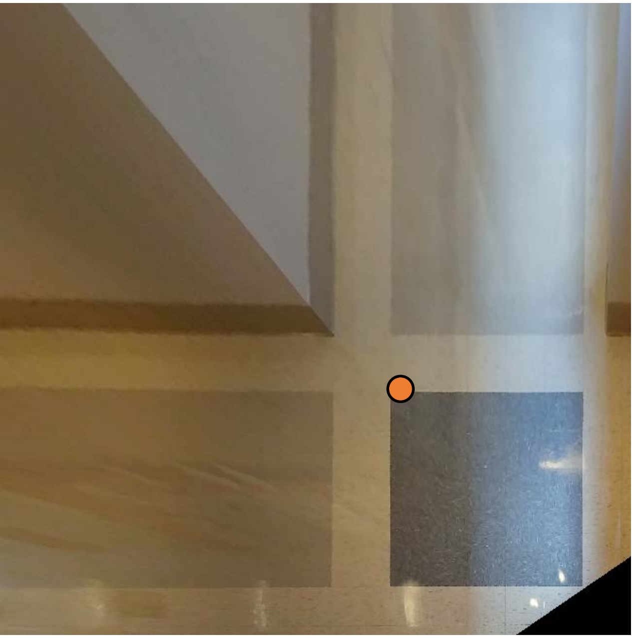
Introduction

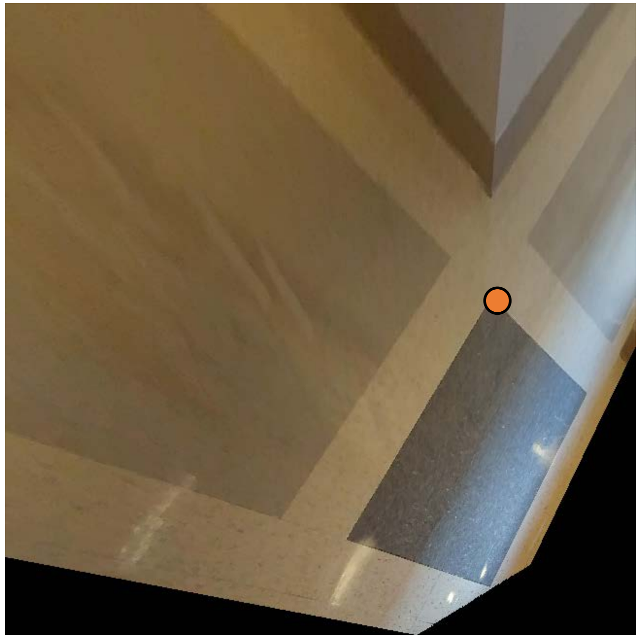
- Are you getting the whole picture?
  - Compact Camera FOV = 50 x 35°
  - Human FOV = 200 x 135°

Introduction

- Are you getting the whole picture?
  - Compact Camera FOV = 50 x 35°
  - Human FOV = 200 x 135°
  - Panoramic Mosaic = 360 x 180°

Homography Computation



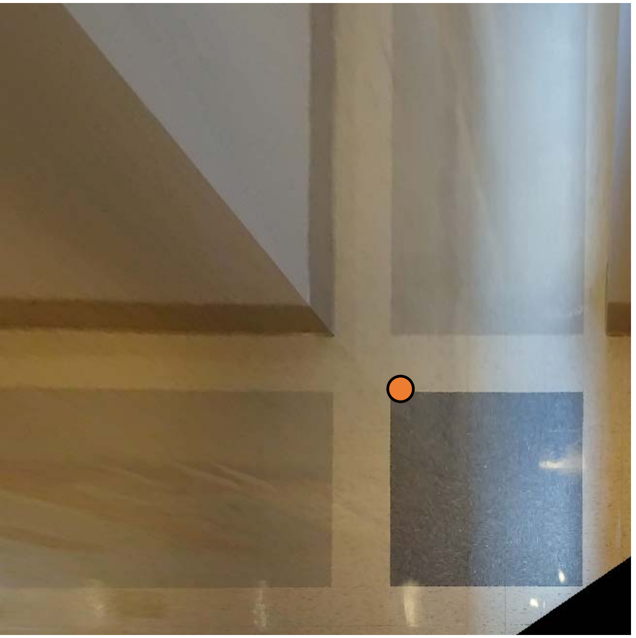


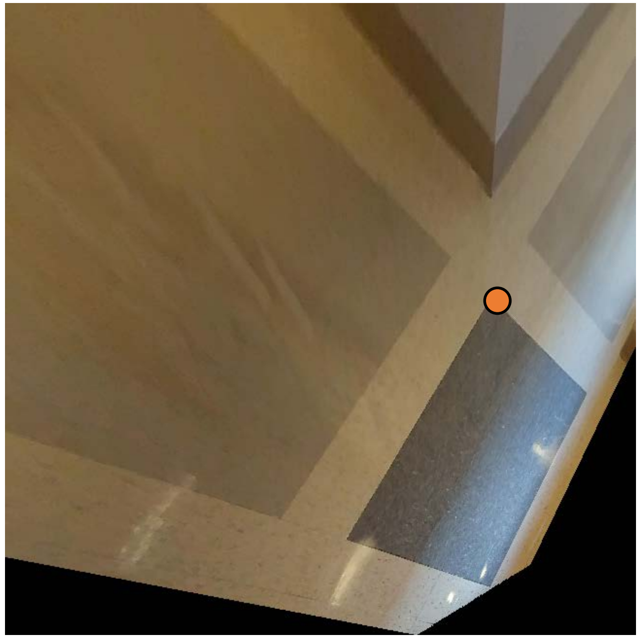
$$v_x = \frac{h_{11}u_x + h_{12}u_y + h_{13}}{h_{31}u_x + h_{32}u_y + h_{33}}$$

$$v_y = \frac{h_{21}u_x + h_{22}u_y + h_{23}}{h_{31}u_x + h_{32}u_y + h_{33}}$$

$$\lambda \begin{bmatrix} v_x \\ v_y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ 1 \end{bmatrix}$$

Homography Computation





$$\lambda \begin{bmatrix} v_x \\ v_y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ 1 \end{bmatrix}$$

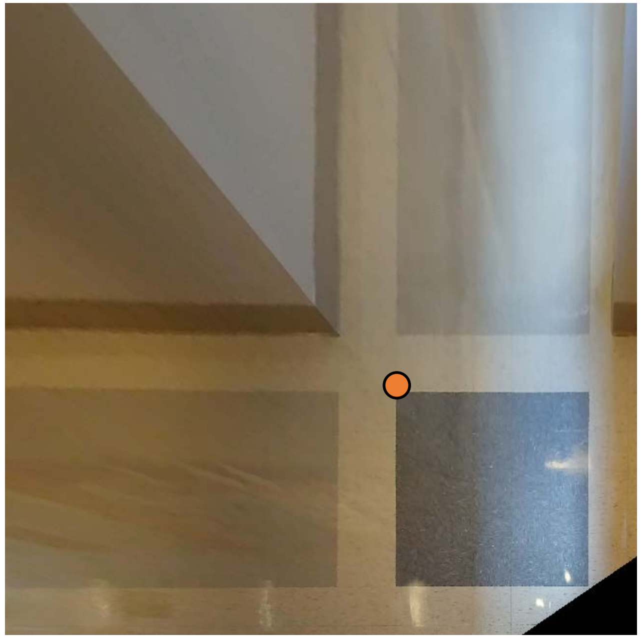
$$v_x = \frac{h_{11}u_x + h_{12}u_y + h_{13}}{h_{31}u_x + h_{32}u_y + h_{33}}$$

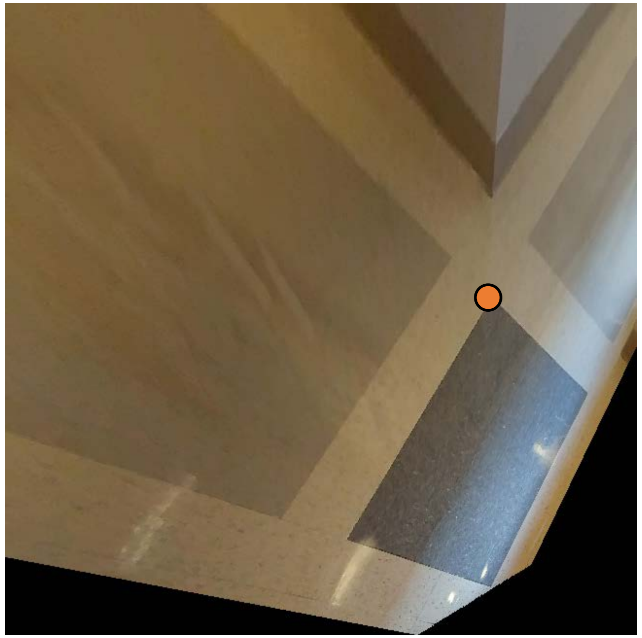
$$v_y = \frac{h_{21}u_x + h_{22}u_y + h_{23}}{h_{31}u_x + h_{32}u_y + h_{33}}$$

$$\rightarrow \begin{aligned} h_{11}u_x + h_{12}u_y + h_{13} - h_{31}u_x v_x - h_{32}u_y v_x - h_{33}v_x &= 0 \\ h_{21}u_x + h_{22}u_y + h_{23} - h_{31}u_x v_y - h_{32}u_y v_y - h_{33}v_y &= 0 \end{aligned}$$

$$\rightarrow \begin{bmatrix} u_x & u_y & 1 & & & & -u_x v_x & -u_y v_x & -v_x \\ & & & u_x & u_y & 1 & -u_x v_y & -u_y v_y & -v_y \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Homography Computation





$$\lambda \begin{bmatrix} v_x \\ v_y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ 1 \end{bmatrix}$$

$$v_x = \frac{h_{11}u_x + h_{12}u_y + h_{13}}{h_{31}u_x + h_{32}u_y + h_{33}}$$

$$v_y = \frac{h_{21}u_x + h_{22}u_y + h_{23}}{h_{31}u_x + h_{32}u_y + h_{33}}$$

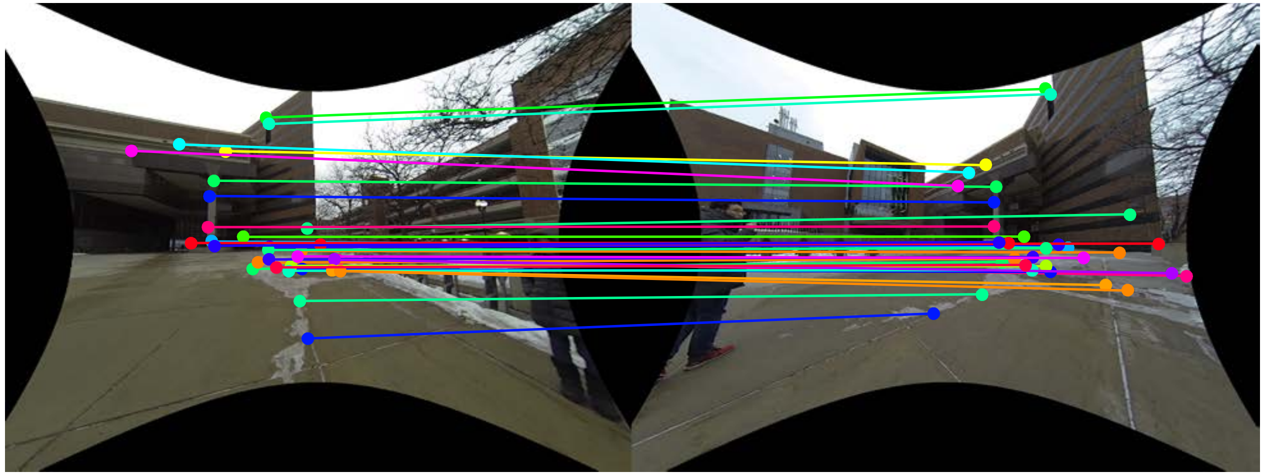
$$\rightarrow \quad \begin{aligned} h_{11}u_x + h_{12}u_y + h_{13} - h_{31}u_x v_x - h_{32}u_y v_x - h_{33}v_x &= 0 \\ h_{21}u_x + h_{22}u_y + h_{23} - h_{31}u_x v_y - h_{32}u_y v_y - h_{33}v_y &= 0 \end{aligned}$$

$$\rightarrow \quad \begin{bmatrix} u_x & u_y & 1 & & & & -u_x v_x & -u_y v_x & -v_x \\ & & & u_x & u_y & 1 & -u_x v_y & -u_y v_y & -v_y \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mathbf{A}\,\mathbf{x} = \mathbf{0}$$

2x9

Linear System for Homography Matrix

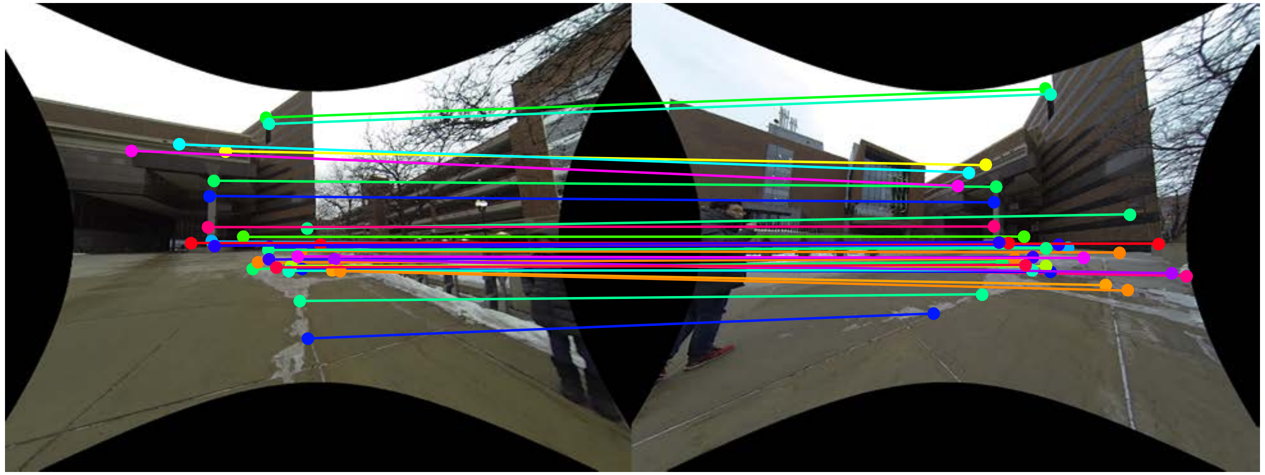


$$\mathbf{A} = \begin{bmatrix} u_x & u_y & 1 & & & & -u_x v_x & -u_y v_x & -v_x \\ & & & u_x & u_y & & -u_x v_y & -u_y v_y & -v_y \end{bmatrix}$$

$$\mathbf{A}\,\mathbf{x} = \mathbf{0}, \quad \mathbf{x} = \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix}, \quad \mathbf{0} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

2x9

How Many Correspondences?



$$\begin{bmatrix} u_x & u_y & 1 & & & & -u_x v_x & -u_y v_x & -v_x \\ & & & u_x & u_y & & -u_x v_y & -u_y v_y & -v_y \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mathbf{A}\,\mathbf{x} = \mathbf{0}$$

2x9

What is minimum m?

$\mathbf{I}_1$ $\mathbf{I}_2$

$$\left\{\begin{matrix} \mathbf{v}_1 \leftrightarrow \mathbf{u}_1 \\ \mathbf{v}_2 \leftrightarrow \mathbf{u}_2 \\ \mathbf{v}_3 \leftrightarrow \mathbf{u}_3 \\ \mathbf{v}_4 \leftrightarrow \mathbf{u}_4 \end{matrix}\right\} \rightarrow \mathbf{H}$$

Homography computation

$$\begin{bmatrix} u^1_x & u^1_y & 1 & & & & -u^1_x v^1_x & -u^1_y v^1_x & -v^1_x \\ & & & u^1_x & u^1_y & 1 & -u^1_x v^1_y & -u^1_y v^1_y & -v^1_y \\ u^2_x & u^2_y & 1 & & & & -u^2_x v^2_x & -u^2_y v^2_x & -v^2_x \\ & & & u^2_x & u^2_y & 1 & -u^2_x v^2_y & -u^2_y v^2_y & -v^2_y \\ u^3_x & u^3_y & 1 & & & & -u^3_x v^3_x & -u^3_y v^3_x & -v^3_x \\ & & & u^3_x & u^3_y & 1 & -u^3_x v^3_y & -u^3_y v^3_y & -v^3_y \\ u^4_x & u^4_y & 1 & & & & -u^4_x v^4_x & -u^4_y v^4_x & -v^4_x \\ & & & u^4_x & u^4_y & 1 & -u^4_x v^4_y & -u^4_y v^4_y & -v^4_y \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$\mathbf{I}_1$ $\mathbf{I}_2$

$$\left.\begin{Bmatrix} \mathbf{v}_1 \leftrightarrow \mathbf{u}_1 \\ \mathbf{v}_2 \leftrightarrow \mathbf{u}_2 \\ \mathbf{v}_3 \leftrightarrow \mathbf{u}_3 \\ \mathbf{v}_4 \leftrightarrow \mathbf{u}_4 \end{Bmatrix}\right. \rightarrow \mathbf{H}$$

Homography computation

$$\underbrace{\begin{bmatrix} u^1_x & u^1_y & 1 & & & & -u^1_x v^1_x & -u^1_y v^1_x & -v^1_x \\ & & & u^1_x & u^1_y & 1 & -u^1_x v^1_y & -u^1_y v^1_y & -v^1_y \\ u^2_x & u^2_y & 1 & & & & -u^2_x v^2_x & -u^2_y v^2_x & -v^2_x \\ & & & u^2_x & u^2_y & 1 & -u^2_x v^2_y & -u^2_y v^2_y & -v^2_y \\ u^3_x & u^3_y & 1 & & & & -u^3_x v^3_x & -u^3_y v^3_x & -v^3_x \\ & & & u^3_x & u^3_y & 1 & -u^3_x v^3_y & -u^3_y v^3_y & -v^3_y \\ u^4_x & u^4_y & 1 & & & & -u^4_x v^4_x & -u^4_y v^4_x & -v^4_x \\ & & & u^4_x & u^4_y & 1 & -u^4_x v^4_y & -u^4_y v^4_y & -v^4_y \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}}_{\mathbf{0}}$$

I_1 I_2

$$\left.\begin{Bmatrix} \mathbf{v}_1 \leftrightarrow \mathbf{u}_1 \\ \mathbf{v}_2 \leftrightarrow \mathbf{u}_2 \\ \mathbf{v}_3 \leftrightarrow \mathbf{u}_3 \\ \mathbf{v}_4 \leftrightarrow \mathbf{u}_4 \end{Bmatrix}\right. \rightarrow \mathbf{H}$$

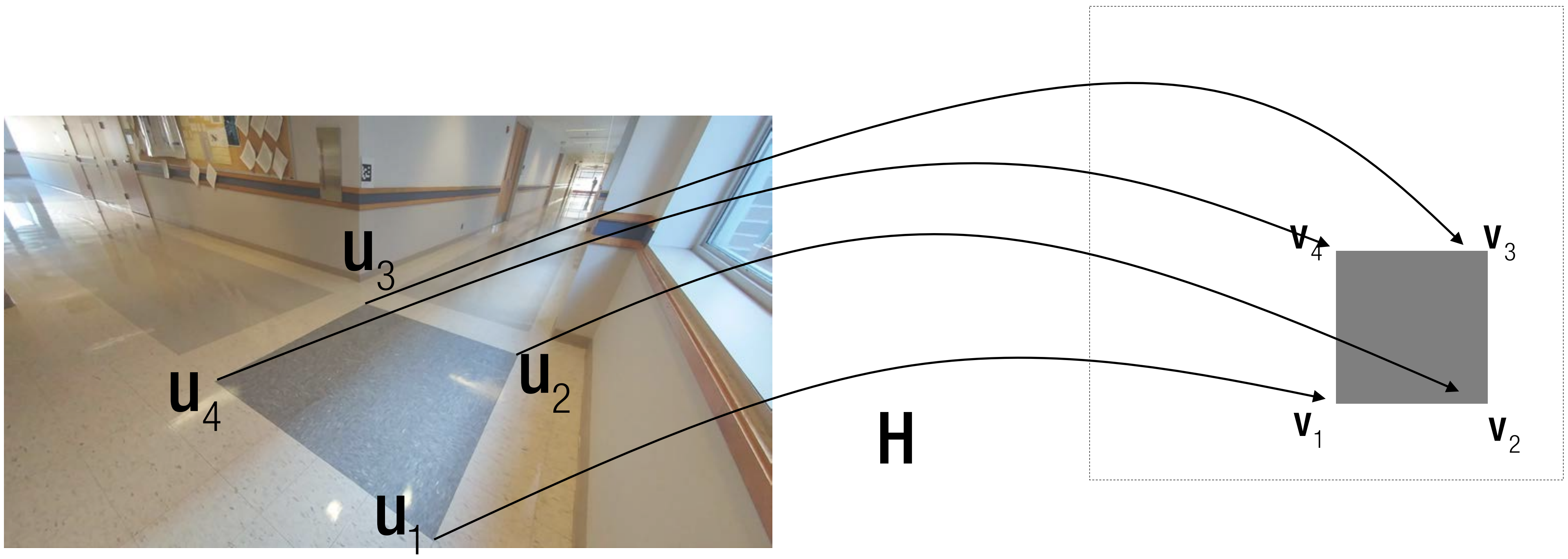
Homography computation

$$\mathbf{A}\mathbf{X} = \mathbf{0}$$

$$\begin{bmatrix} u^1_x & u^1_y & 1 & & & & -u^1_x v^1_x & -u^1_y v^1_x & -v^1_x \\ & & & u^1_x & u^1_y & 1 & -u^1_x v^1_y & -u^1_y v^1_y & -v^1_y \\ u^2_x & u^2_y & 1 & & & & -u^2_x v^2_x & -u^2_y v^2_x & -v^2_x \\ & & & u^2_x & u^2_y & 1 & -u^2_x v^2_y & -u^2_y v^2_y & -v^2_y \\ u^3_x & u^3_y & 1 & & & & -u^3_x v^3_x & -u^3_y v^3_x & -v^3_x \\ & & & u^3_x & u^3_y & 1 & -u^3_x v^3_y & -u^3_y v^3_y & -v^3_y \\ u^4_x & u^4_y & 1 & & & & -u^4_x v^4_x & -u^4_y v^4_x & -v^4_x \\ & & & u^4_x & u^4_y & 1 & -u^4_x v^4_y & -u^4_y v^4_y & -v^4_y \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

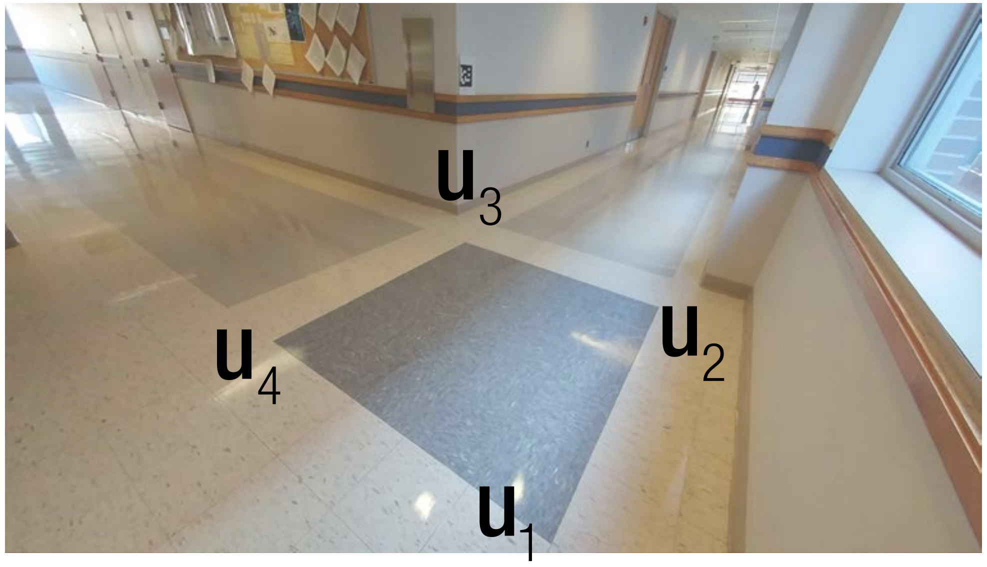
```
[u,d,v] = svd(A);
X = v(:,end)/v(end,end);
H = reshape(X,3,3)';
```

Fun with Homography



The image can be rectified as if it is seen from top view.

Fun with Homography



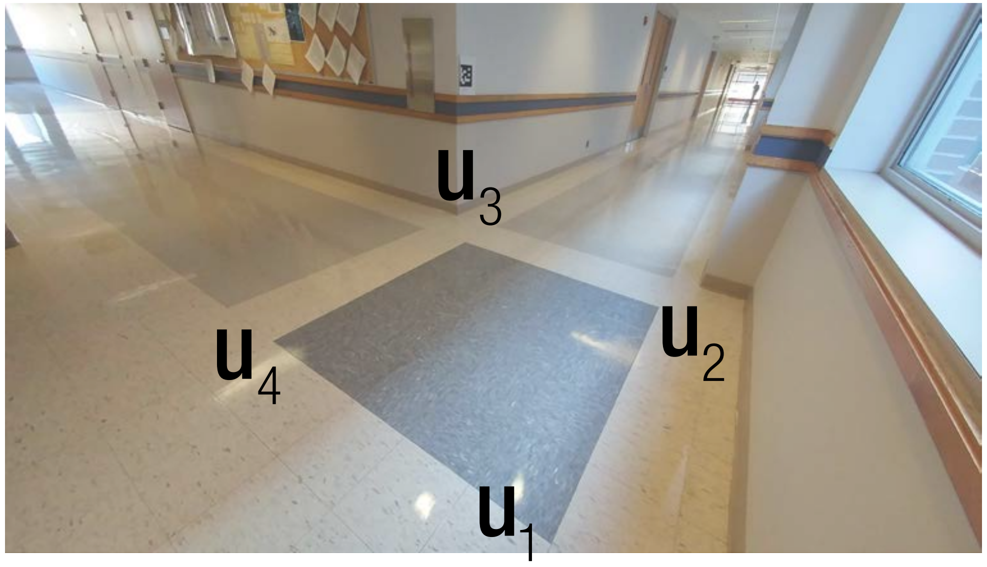
RectificationViaHomography.m

```
u = [u1'; u2'; u3'; u4'];
v = [v1'; v2'; v3'; v4'];

% Need at least non-colinear four points
H = ComputeHomography(v, u);

im_warped = ImageWarping(im, inv(H));
```

Fun with Homography



RectificationViaHomography.m

```
u = [u1'; u2'; u3'; u4'];
v = [v1'; v2'; v3'; v4'];

% Need at least non-colinear four points
H = ComputeHomography(v, u);

im_warped = ImageWarping(im, inv(H));
```

ImageWarping.m

```
u_x = H(1,1)*v_x + H(1,2)*v_y + H(1,3);
u_y = H(2,1)*v_x + H(2,2)*v_y + H(2,3);
u_z = H(3,1)*v_x + H(3,2)*v_y + H(3,3);

u_x = u_x./u_z;
u_y = u_y./u_z;
```

$$\lambda \begin{bmatrix} v_x \\ v_y \\ 1 \end{bmatrix} = \mathbf{H}^{-1} \begin{bmatrix} u_x \\ u_y \\ 1 \end{bmatrix}$$

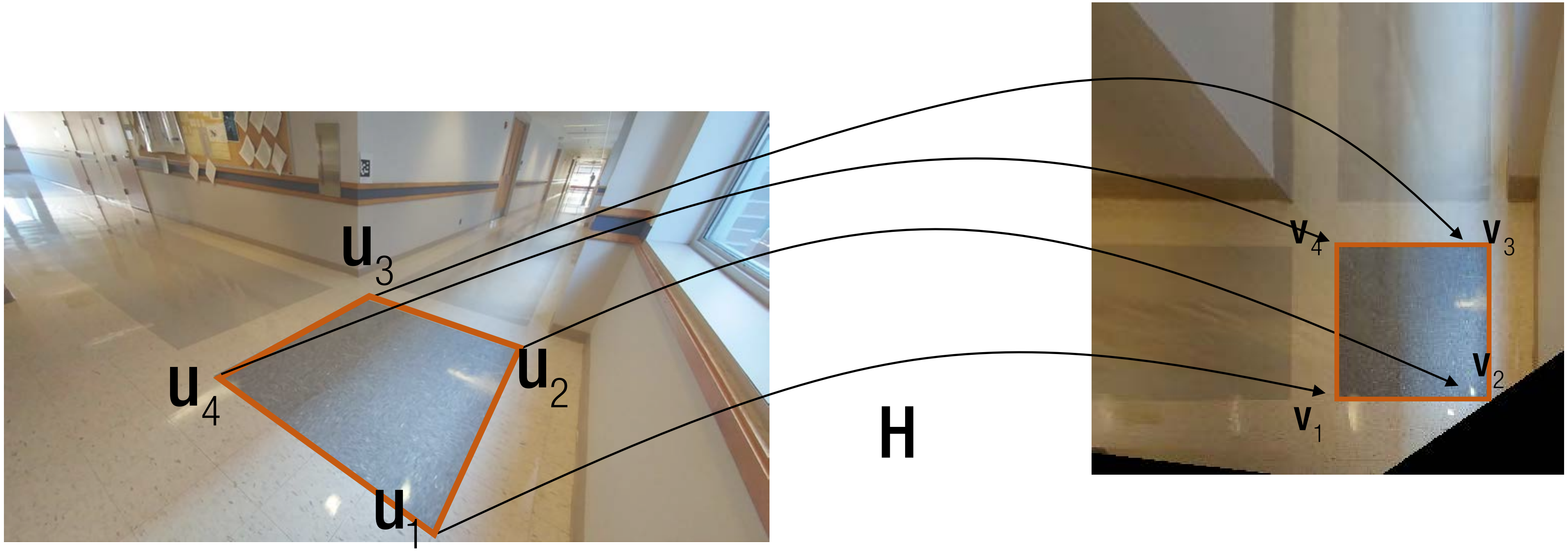
```
im_warped(:,:,1) = reshape(interp2(im(:,:,1), u_x(:), u_y(:)), [h, w]);
im_warped(:,:,2) = reshape(interp2(im(:,:,2), u_x(:), u_y(:)), [h, w]);
im_warped(:,:,3) = reshape(interp2(im(:,:,3), u_x(:), u_y(:)), [h, w]);

im_warped = uint8(im_warped);
```

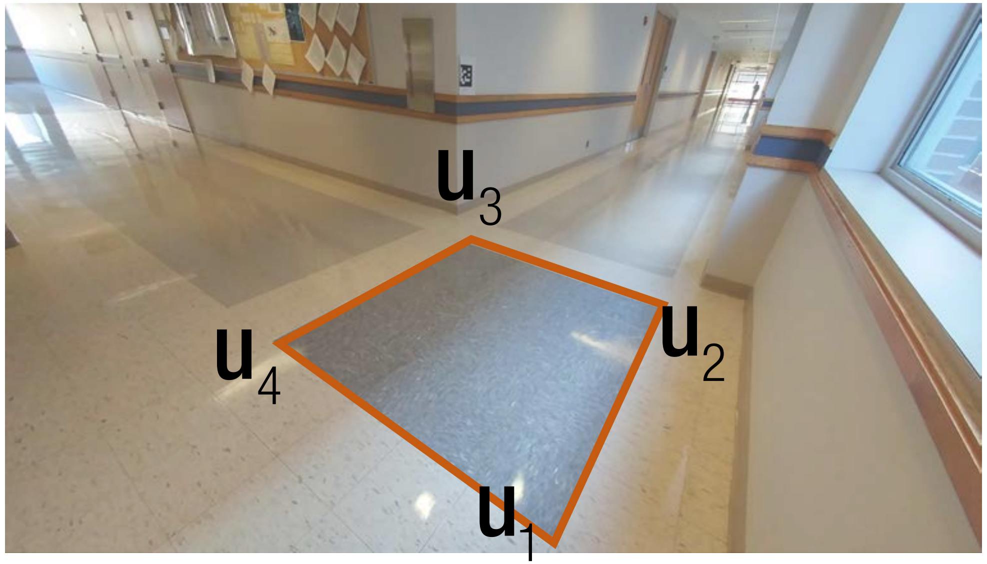
Cf) ImageWarpingEuclidean.m

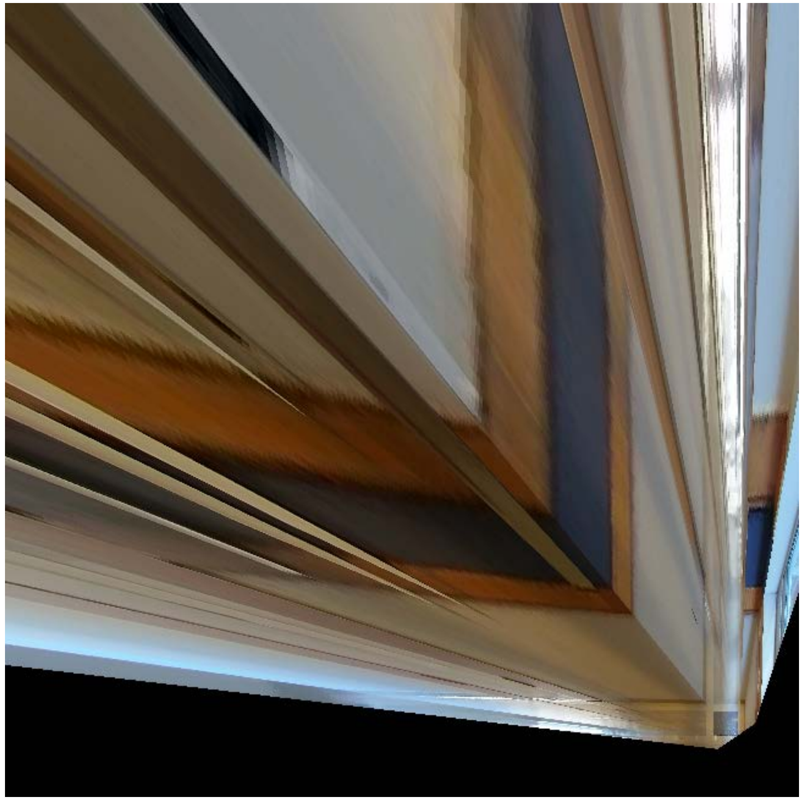
```
u_x = H(1,1)*v_x + H(1,2)*v_y + H(1,3);
u_y = H(2,1)*v_x + H(2,2)*v_y + H(2,3);
```

Fun with Homography



Fun with Homography

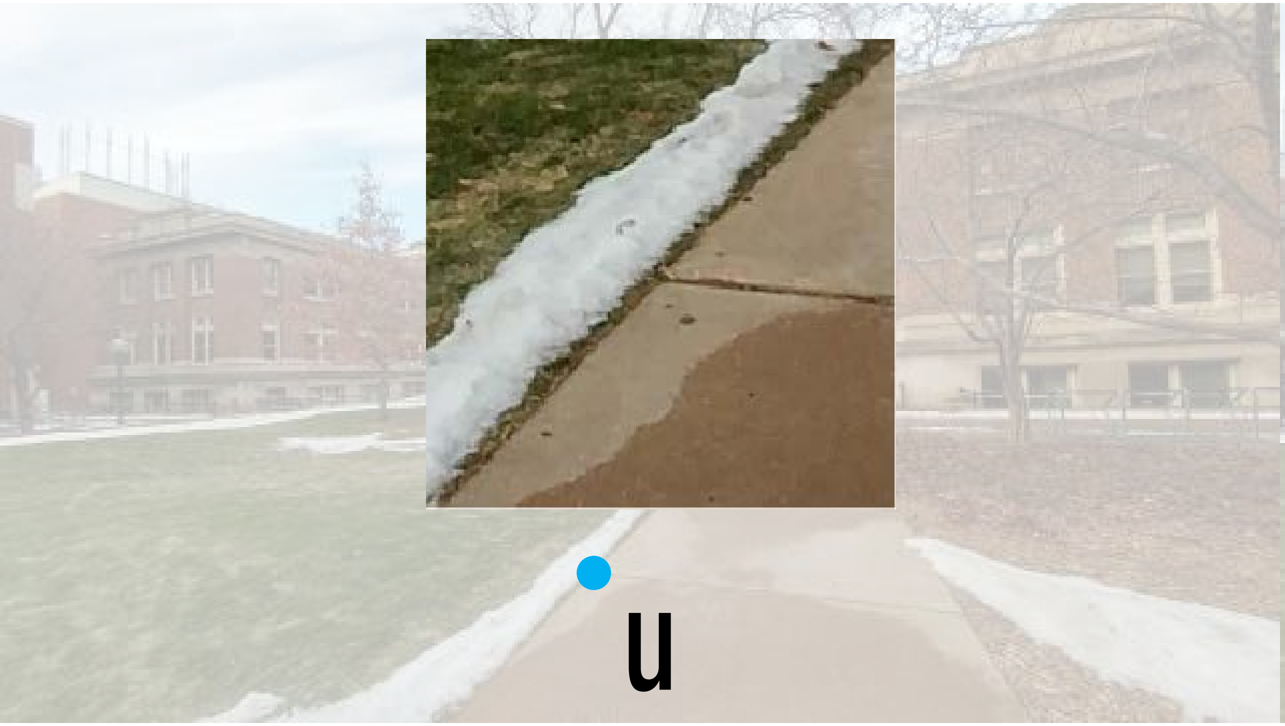




Feature Matching


u
v

Local Patch



Local Patch (Orientation)

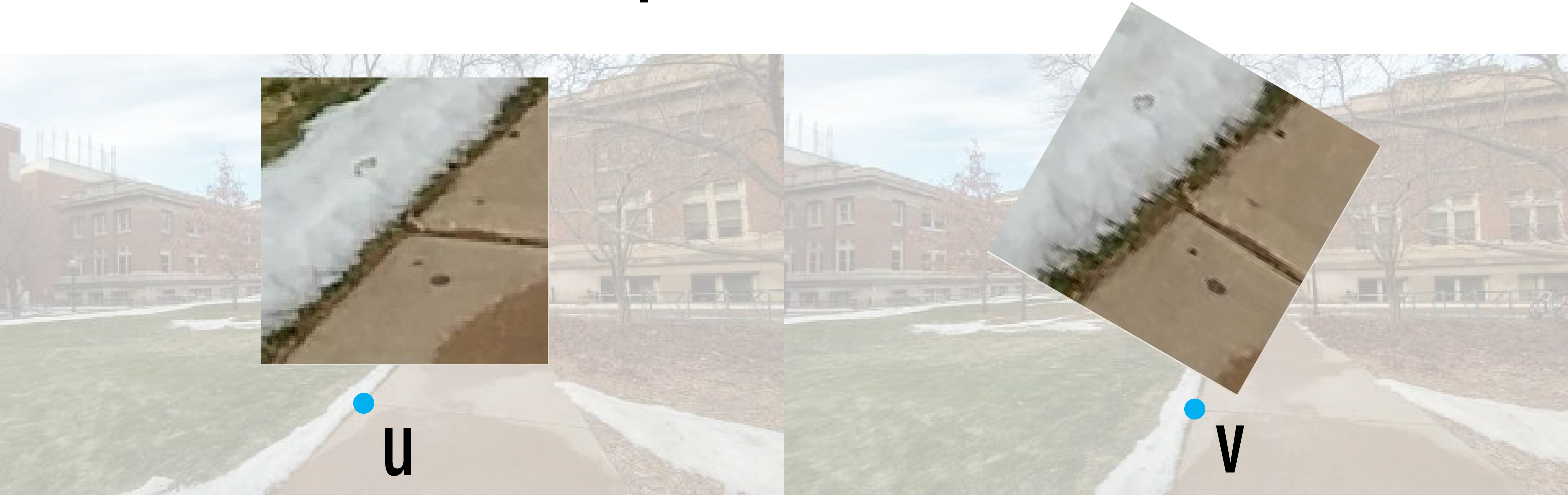
Local Patch (Scale)

Local Visual Descriptor

Desired properties:

- Repeatability: the same point is repeatedly detected.
- Discriminativity: the point is unique.

Local Visual Descriptor



Desired properties:

- Repeatability: the same point is repeatedly detected.
- Discriminativity: the point is unique.
- Orientation aware

Image Features

slides from
A. Efros, Steve Seitz and Rick Szeliski

Today's lecture

- Feature detectors
    - scale invariant Harris corners
- Feature descriptors
    - patches, oriented patches

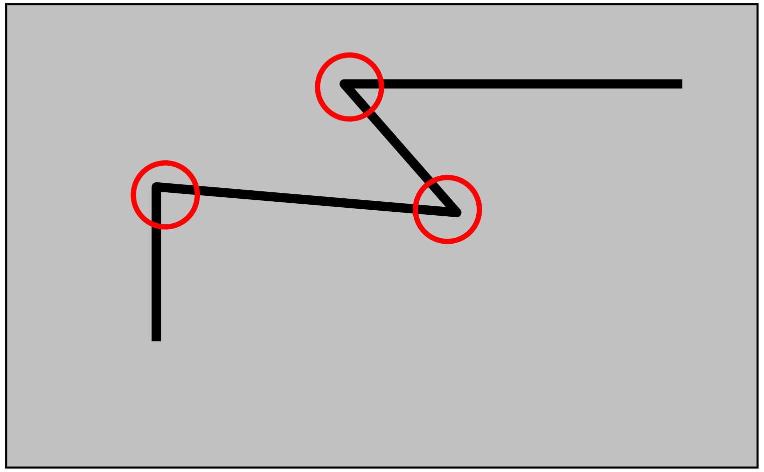
- Reading :
- Multi-image Matching using Multi-scale image patches, CVPR 2005

More motivation…

- Feature points are used for:
  - Image alignment (homography, fundamental matrix)
  - 3D reconstruction
  - Motion tracking
  - Object recognition
  - Indexing and database retrieval
  - Robot navigation
  - … other

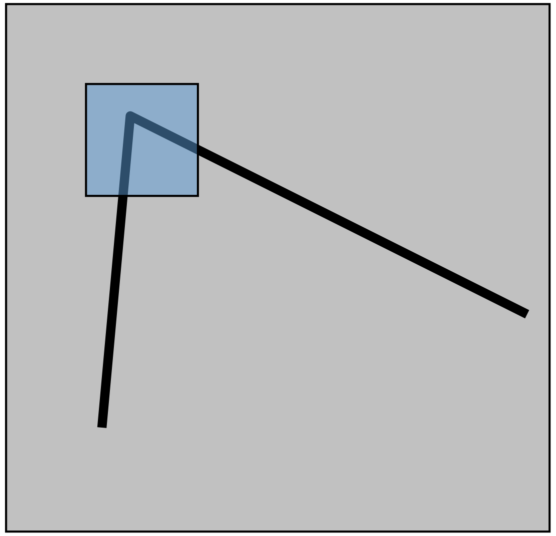
Harris corner detector

- C.Harris, M.Stephens. “A Combined Corner and Edge Detector”. 1988

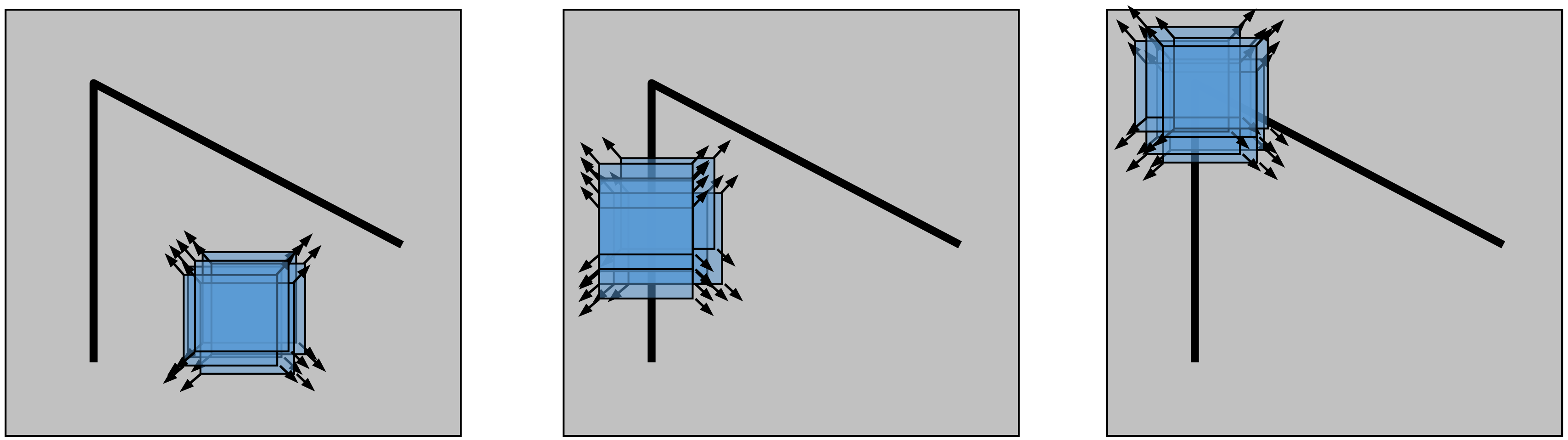


The Basic Idea

- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



Harris Detector: Basic Idea



"flat" region:
no change in
all directions

"edge":
no change along
the edge direction

"corner":
significant change
in all directions

Harris Detector: Mathematics

Change of intensity for the shift [*u*,*v*]:

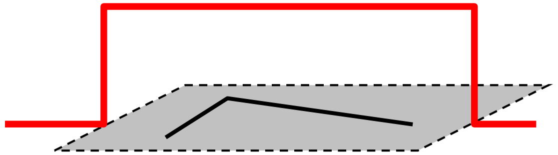
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

Window function

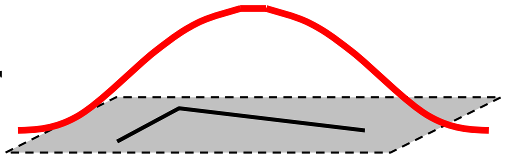
Shifted intensity

Intensity

Window function $w(x,y)$ =



or



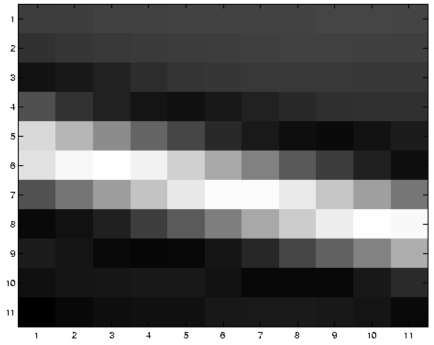
1 in window, 0 outside

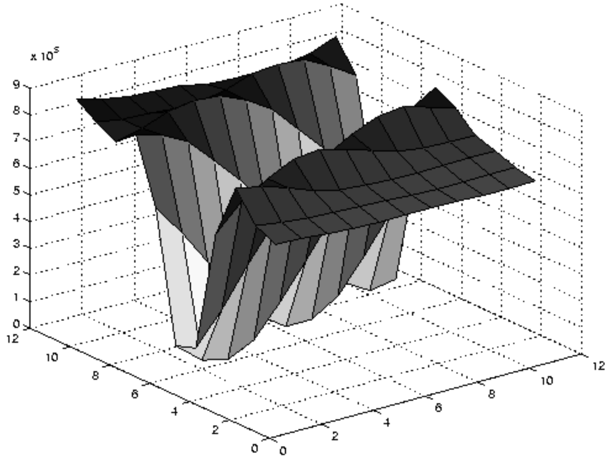
Gaussian

Edge

Sum of squared differences

Err(u,v)



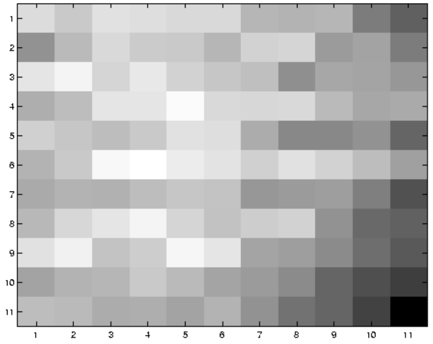


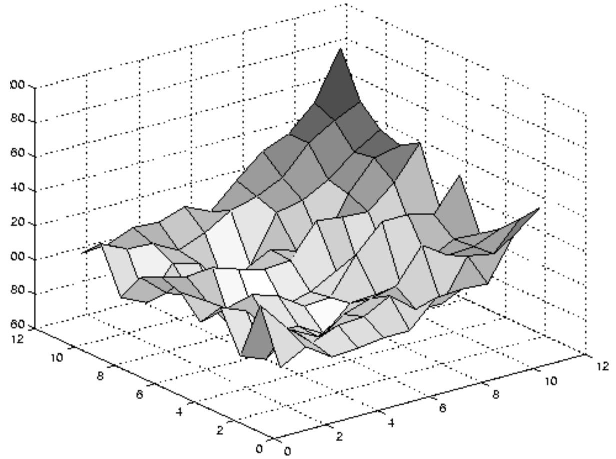
* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Low texture region

Sum of squared differences

Err(x,y)



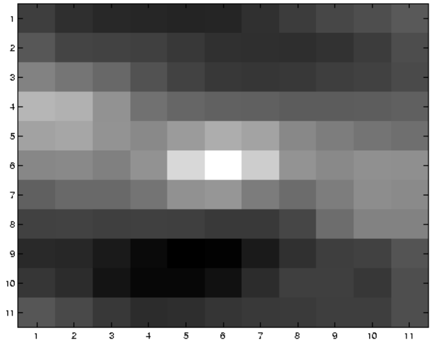


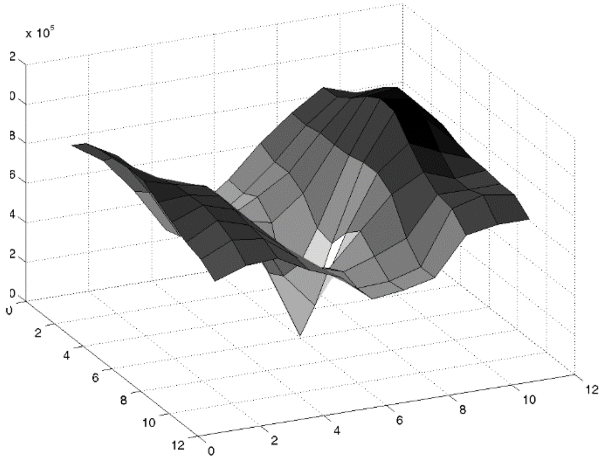
* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

High textured region

Sum of squared differences

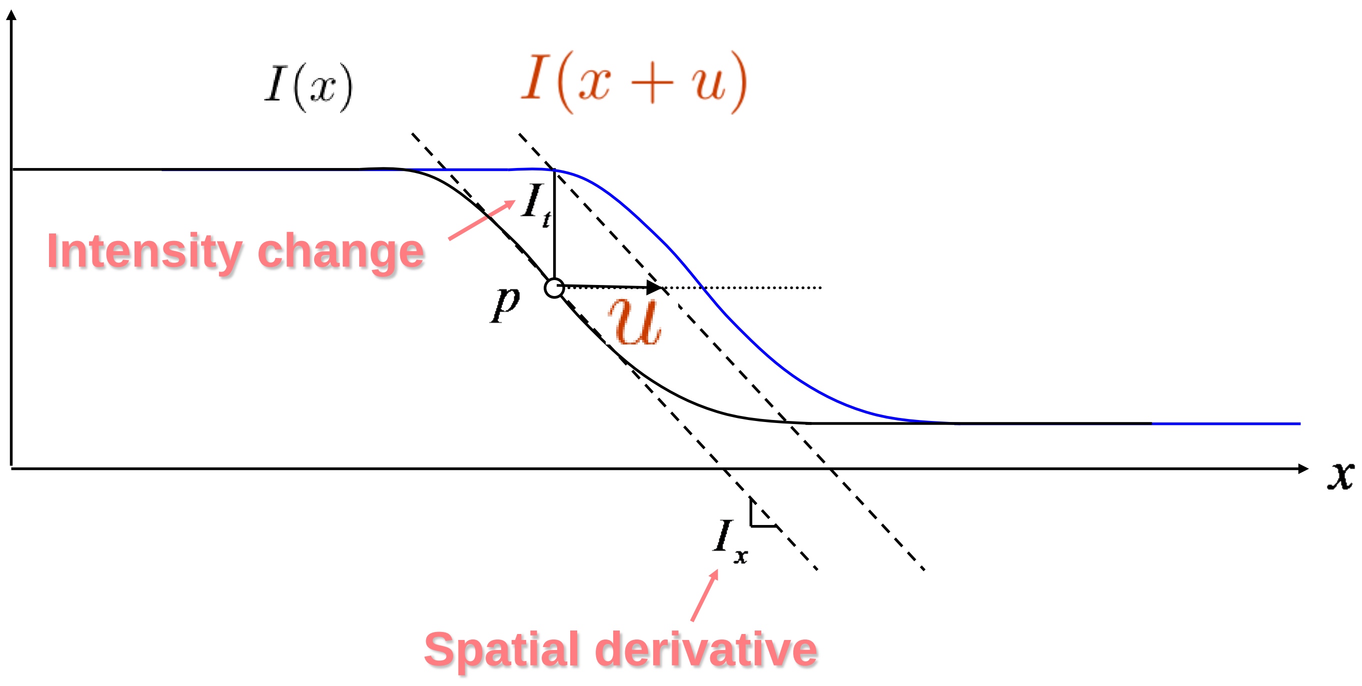
Err(x,y)





*** From Khurram Hassan-Shafique** CAP5415 Computer Vision 2003

We can treat I(x+u,y+v) as image moved slightly.
The change in intensity can be predicted:

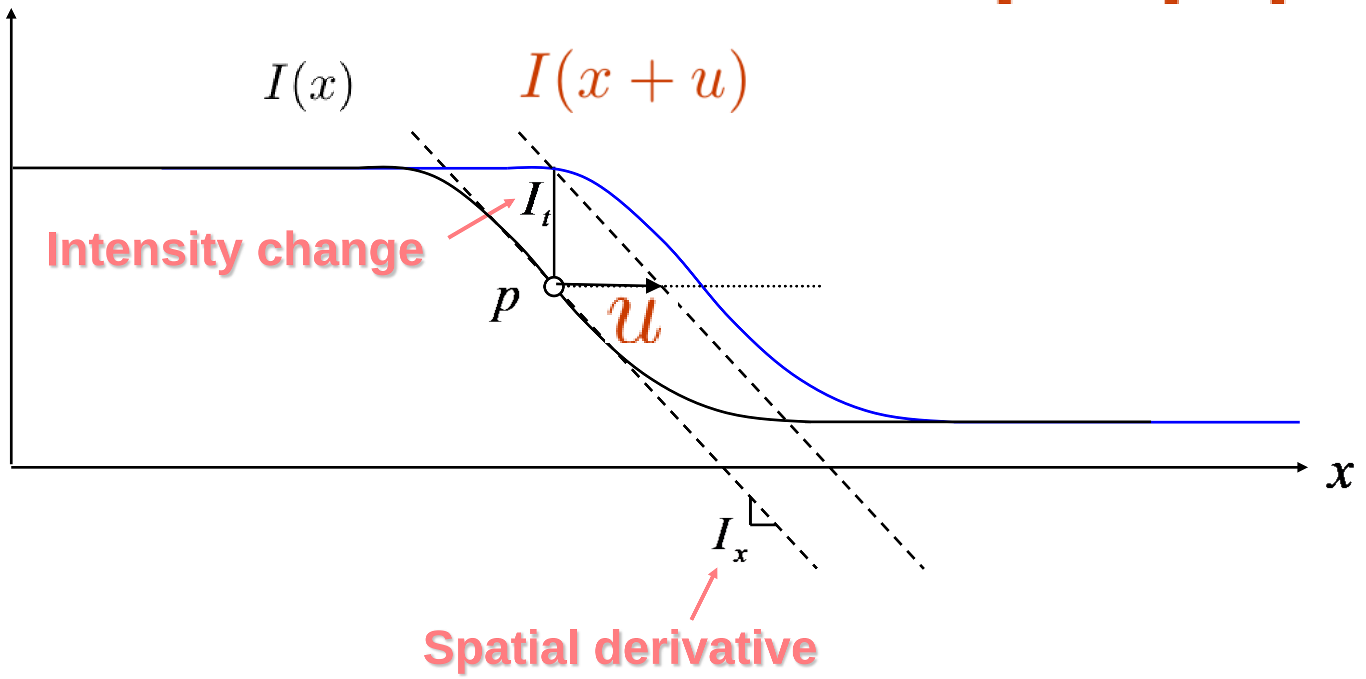


$$I(x+u) - I(x) = u \times I_x$$

intensity change in 1D: $$I(x+u) - I(x) = u \times I_x$$

intensity change in 2D:

$$I(x+u, y+v) - I(x,y) = u \times I_x + v \times I_y$$
$$= [u, v] \cdot [I_x; I_y]$$



Harris Detector: Mathematics

For small shifts $[u,v]$ we have a *bilinear* approximation:

$$E(u,v) \cong [u,v] \; M \begin{bmatrix} u \\ v \end{bmatrix}$$

where M is a 2×2 matrix computed from image derivatives:

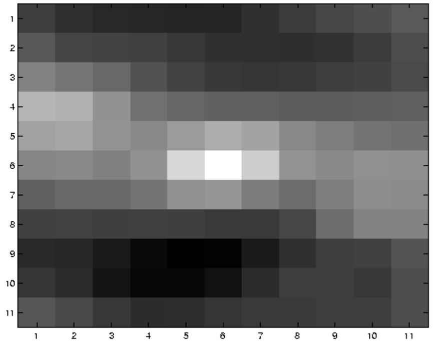
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

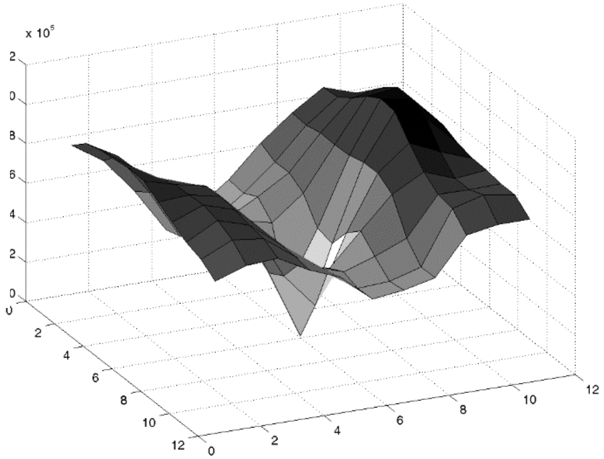
$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y] = \sum \nabla I (\nabla I)^T$$

High textured region

Sum of squared differences

Err(x,y)

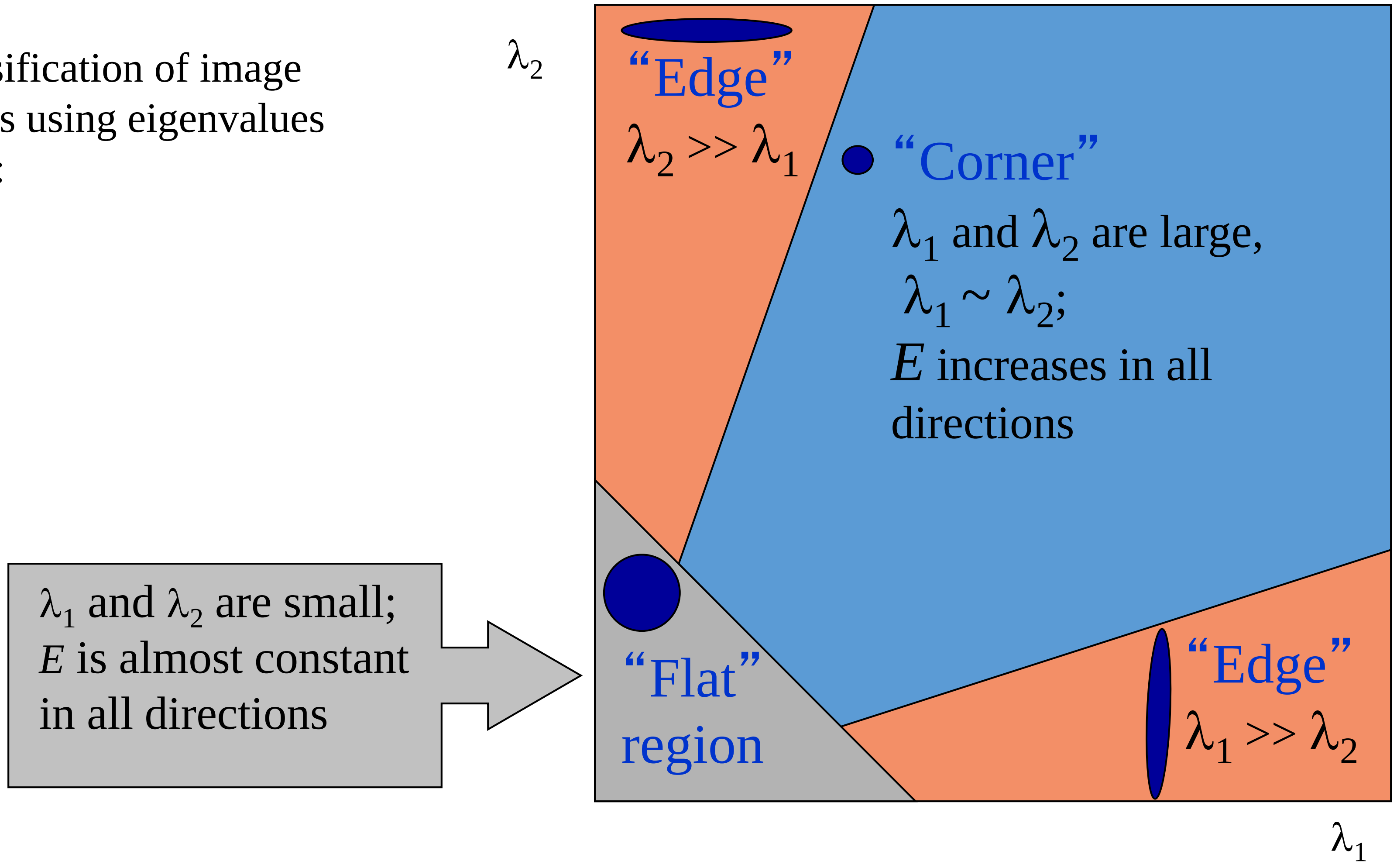




* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Harris Detector: Mathematics

Classification of image points using eigenvalues of *M*:



Harris Detector: Mathematics

Measure of corner response:

$$R = \frac{\det M}{\operatorname{Trace} M}$$

$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

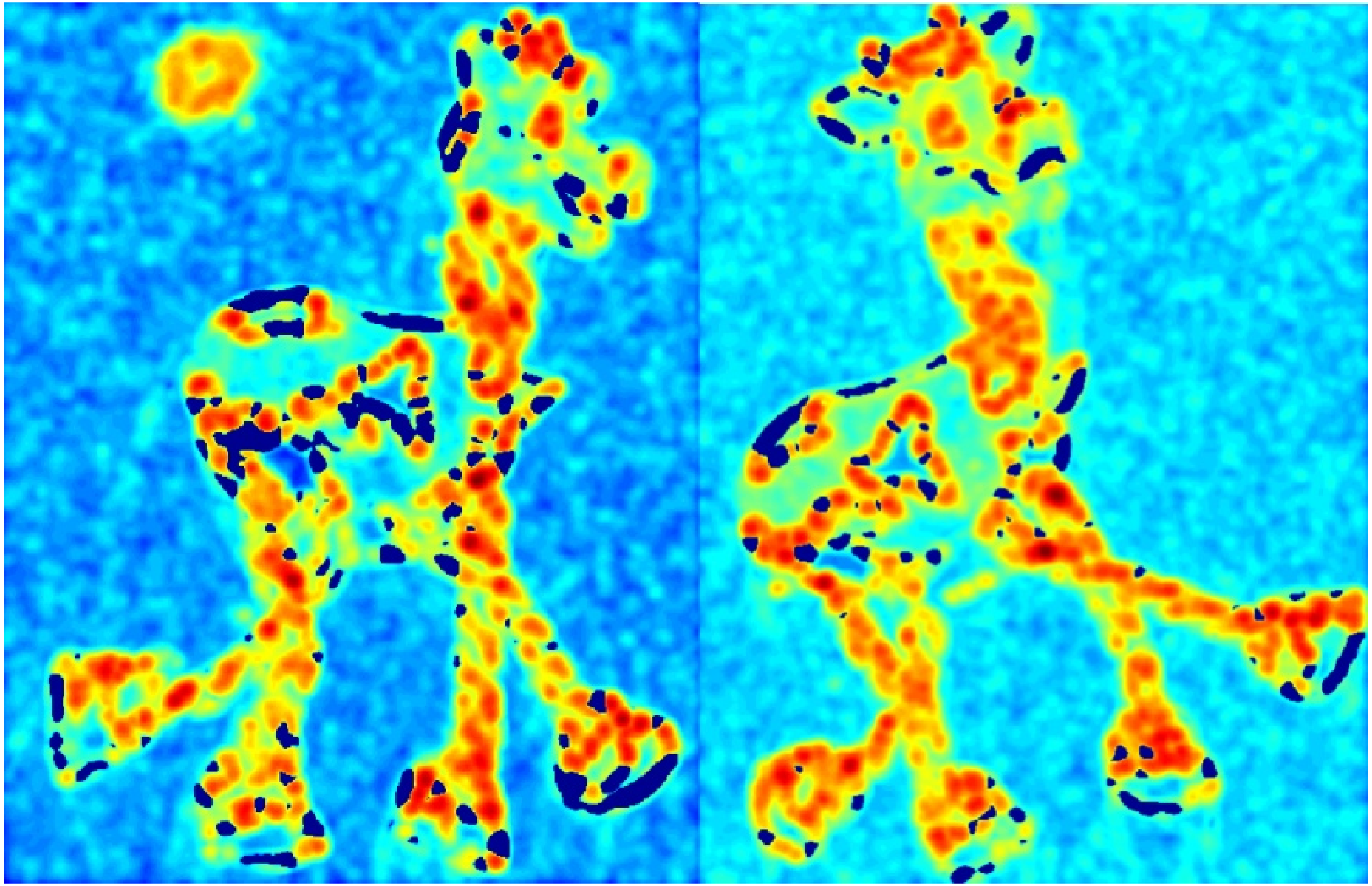
Harris Detector

- The Algorithm:
  - Find points with large corner response function R ($R >$ threshold)
  - Take the points of local maxima of R

Harris Detector: Workflow

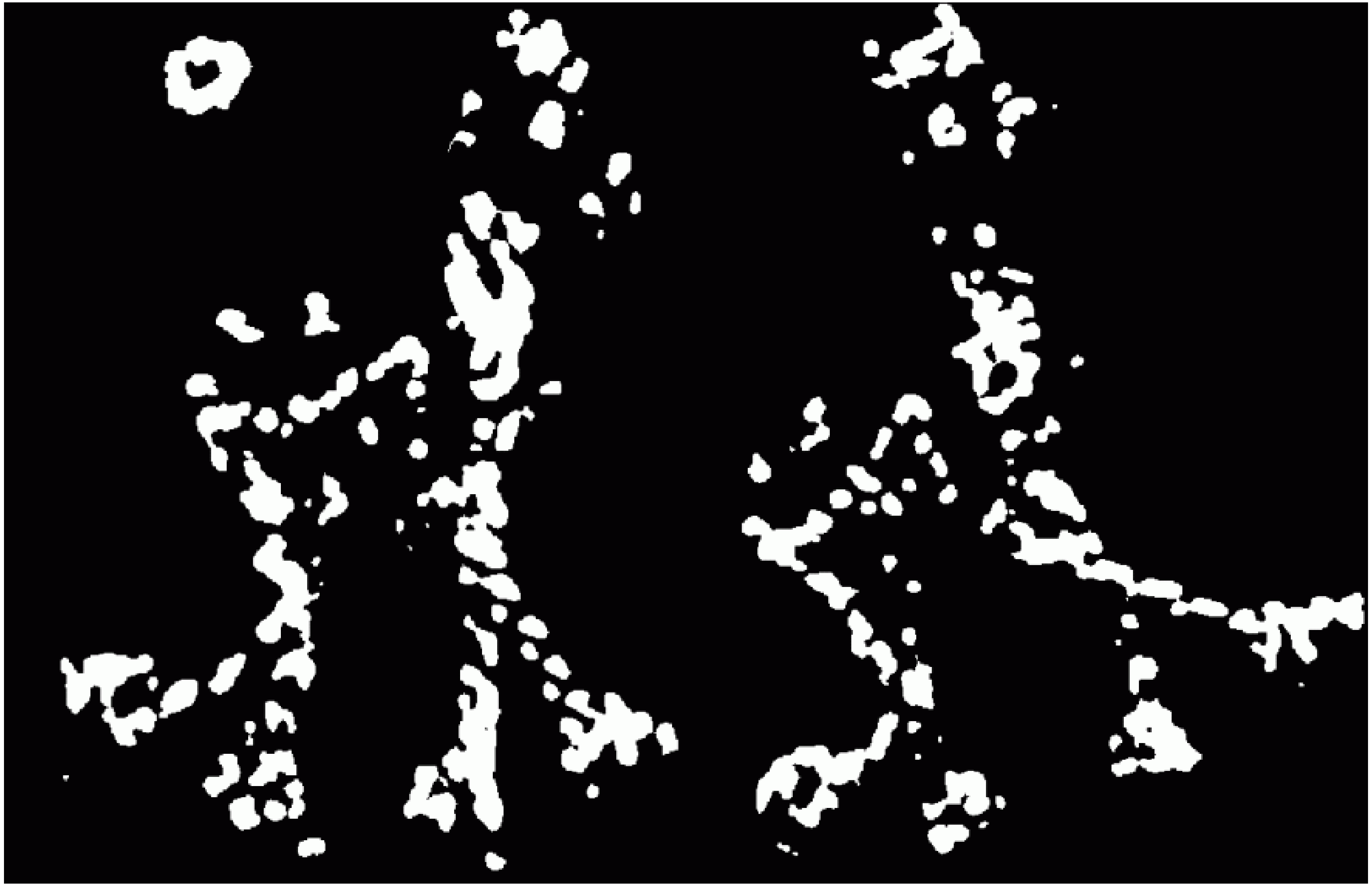
Harris Detector: Workflow

Compute corner response R



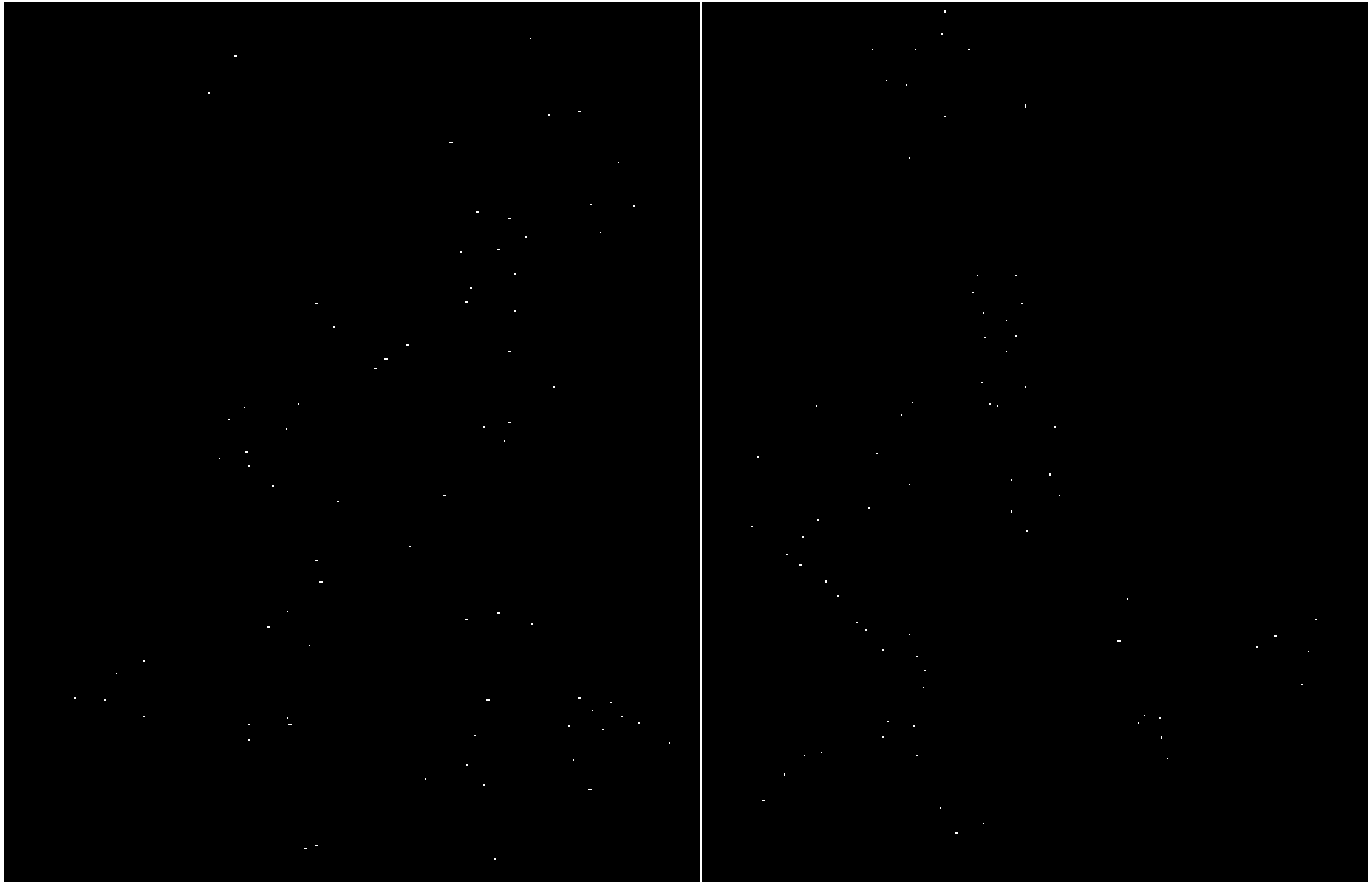
Harris Detector: Workflow

Find points with large corner response: $R>$threshold



Harris Detector: Workflow

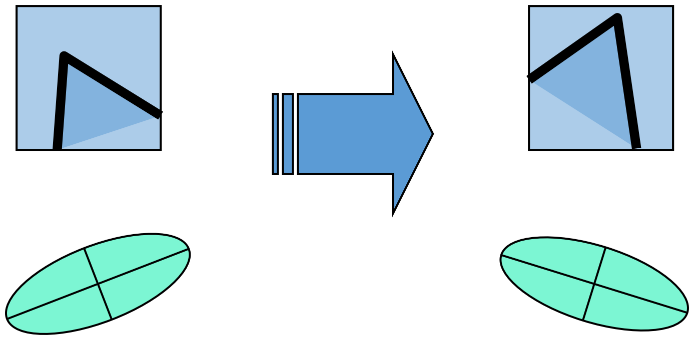
Take only the points of local maxima of R



Harris Detector: Workflow

Harris Detector: Some Properties

- Rotation invariance



Ellipse rotates but its shape (i.e. eigenvalues) remains the same

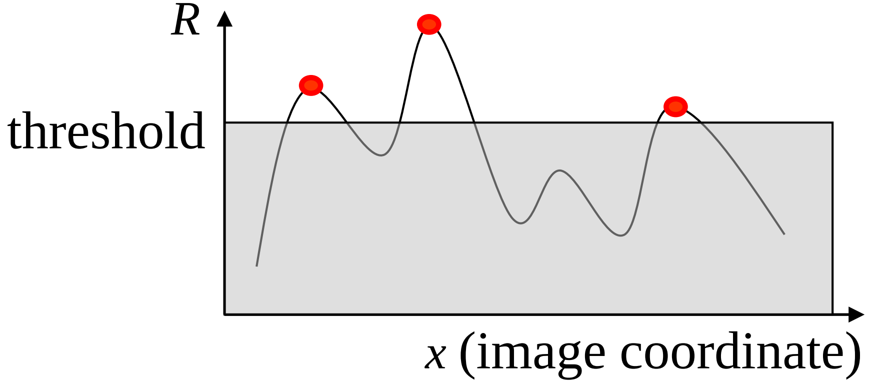
Corner response R is invariant to image rotation

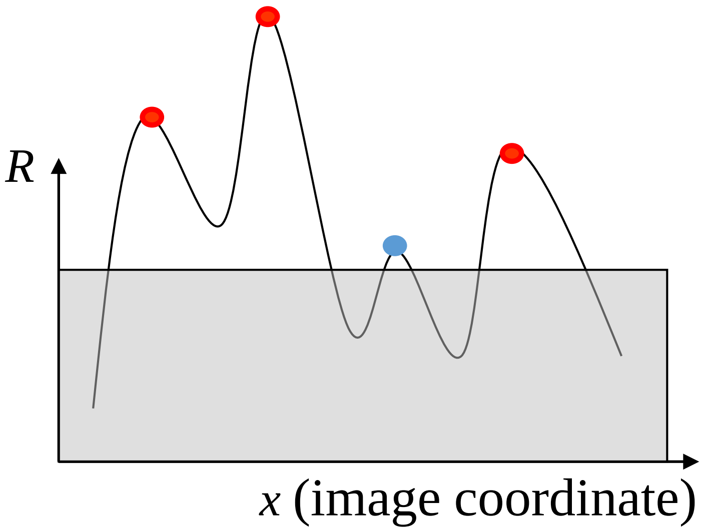
Harris Detector: Some Properties

- Partial invariance to *affine intensity* change

✓ Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$

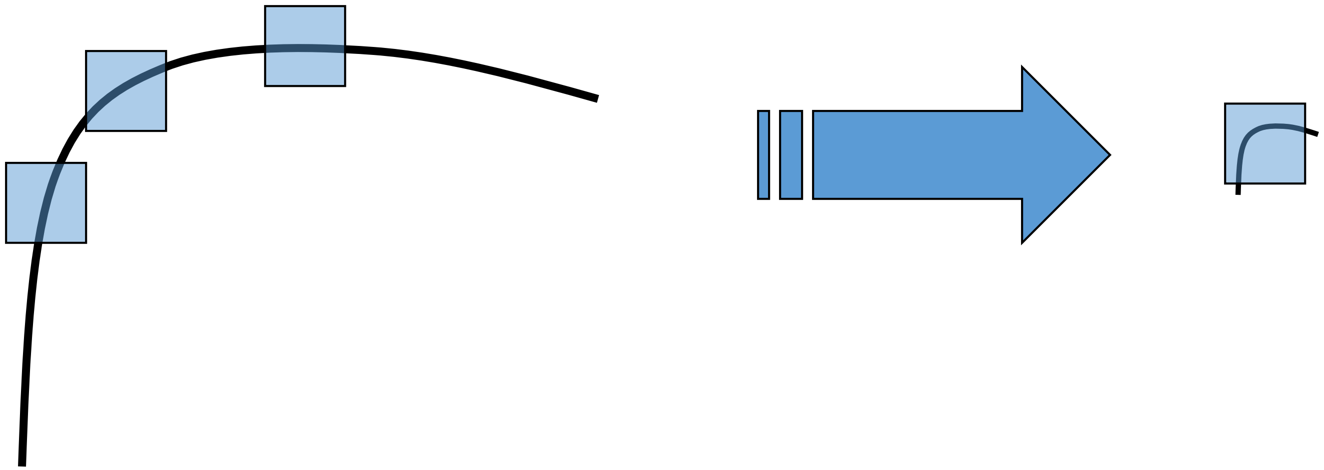
✓ Intensity scale: $I \rightarrow a\, I$





Harris Detector: Some Properties

- But: non-invariant to *image scale*!

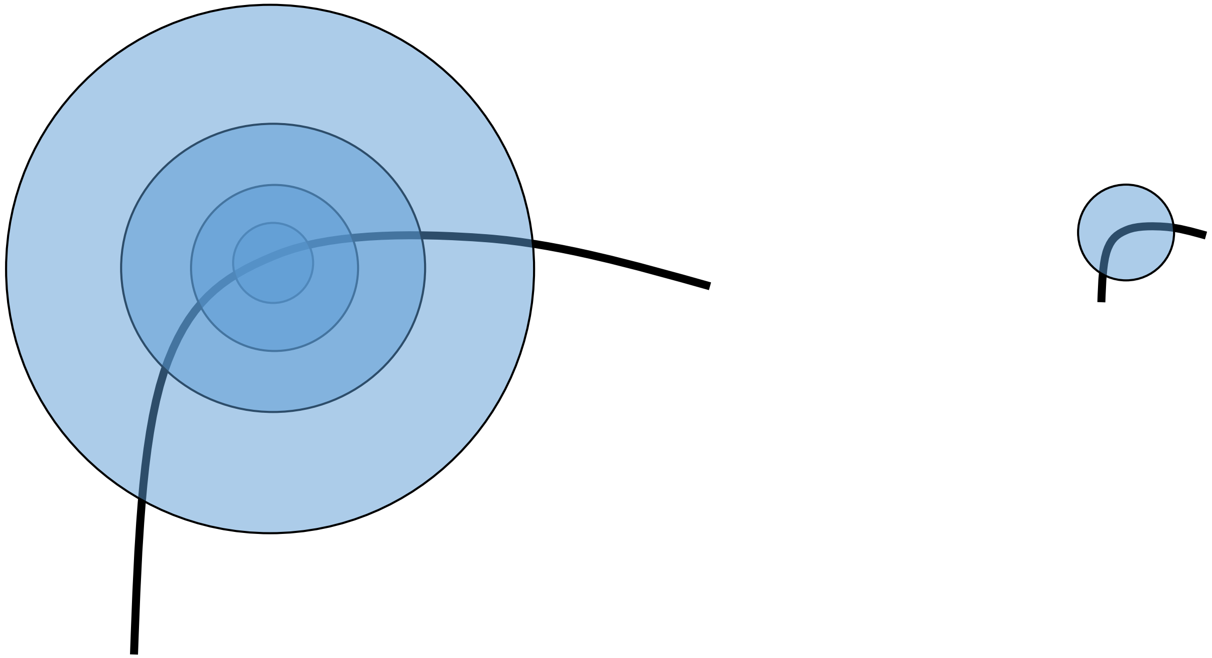


All points will be classified as edges

Corner !

Scale Invariant Detection

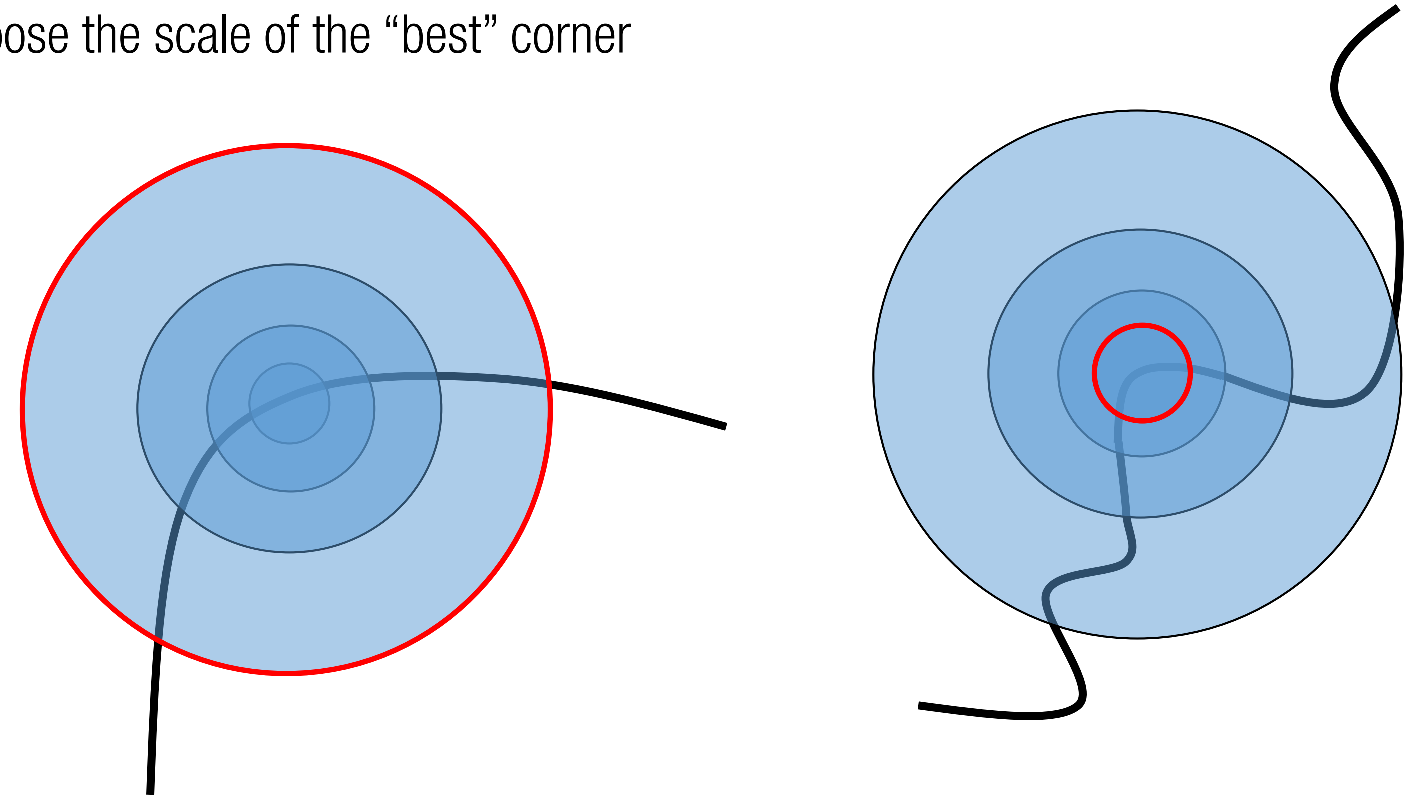
- Consider regions (e.g. circles) of different sizes around a point
- Regions of corresponding sizes will look the same in both images



Scale Invariant Detection

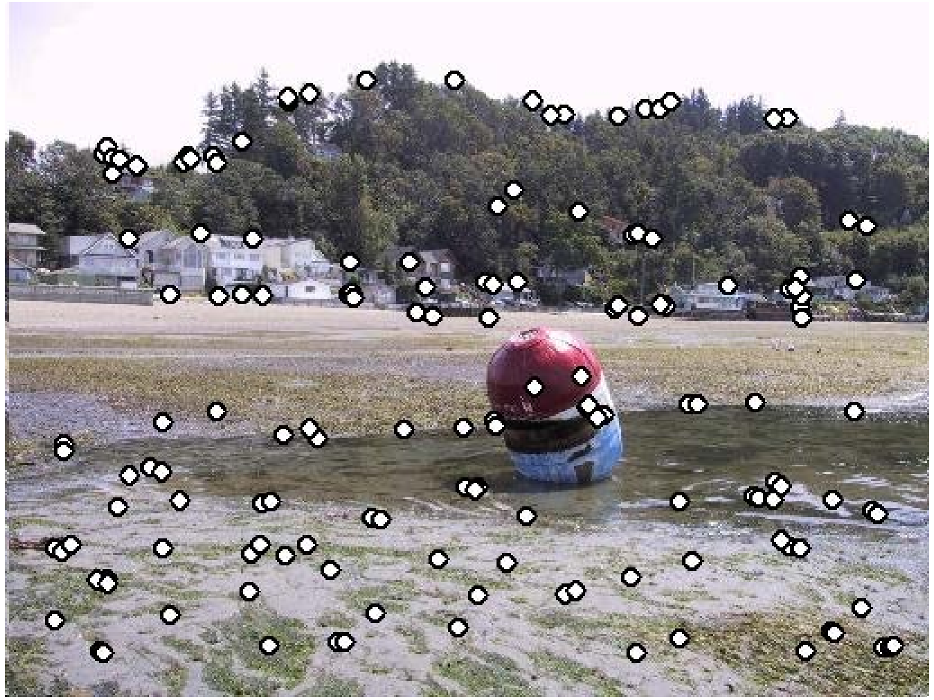
- The problem: how do we choose corresponding circles *independently* in each image?

- Choose the scale of the "best" corner



Feature selection

- Distribute points evenly over the image



Adaptive Non-maximal Suppression

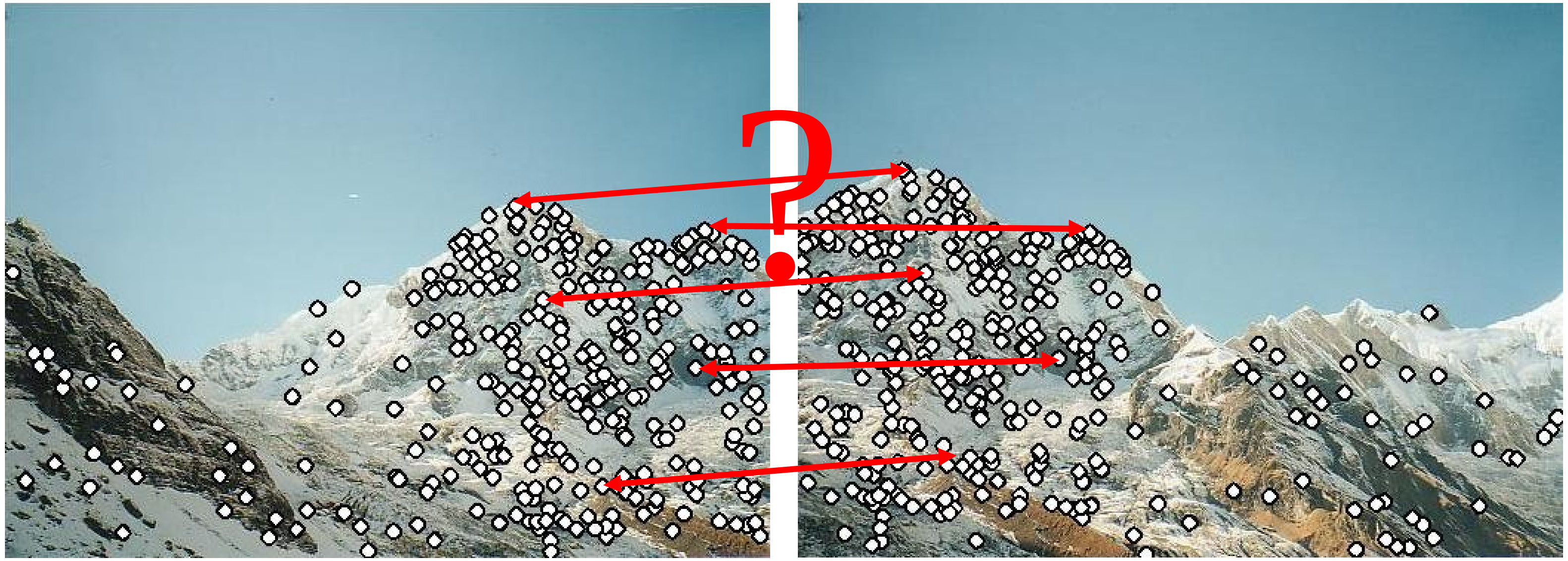
- Desired: Fixed # of features per image
  - Want evenly distributed spatially…
  - Search over non-maximal suppression radius [Brown, Szeliski, Winder, CVPR'05]

(a) Strongest 250 (b) Strongest 500

(c) ANMS 250, $r = 24$ (d) ANMS 500, $r = 16$

Feature descriptors

- We know how to detect points
- Next question: How to match them?



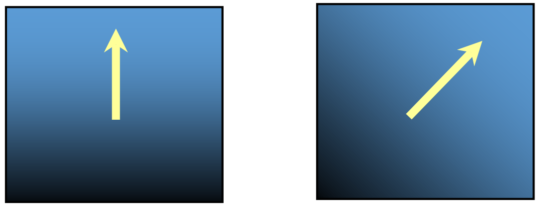
Point descriptor should be:

1. Invariant
2. Distinctive

Descriptors Invariant to Rotation

- Find local orientation

Dominant direction of gradient



- Extract image patches relative to this orientation

Multi-Scale Oriented Patches

- Interest points
  - Multi-scale Harris corners
  - Orientation from blurred gradient
  - Geometrically invariant to rotation
- Descriptor vector
  - Bias/gain normalized sampling of local patch (8x8)
  - Photometrically invariant to affine changes in intensity
- [Brown, Szeliski, Winder, CVPR'2005]

Descriptor Vector

- Orientation = blurred gradient
- Rotation Invariant Frame
    - Scale-space position (x, y, s) + orientation (θ)

Detections at multiple scales

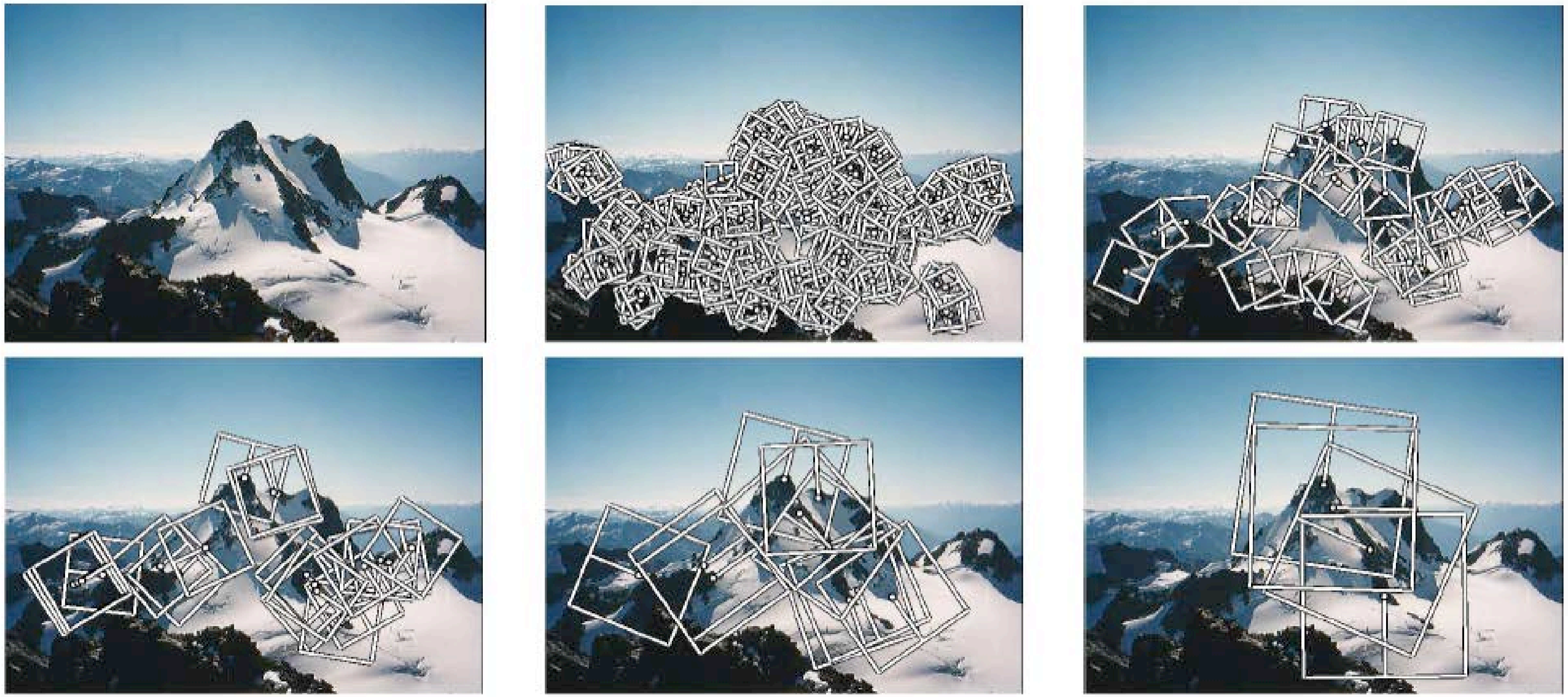


Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.

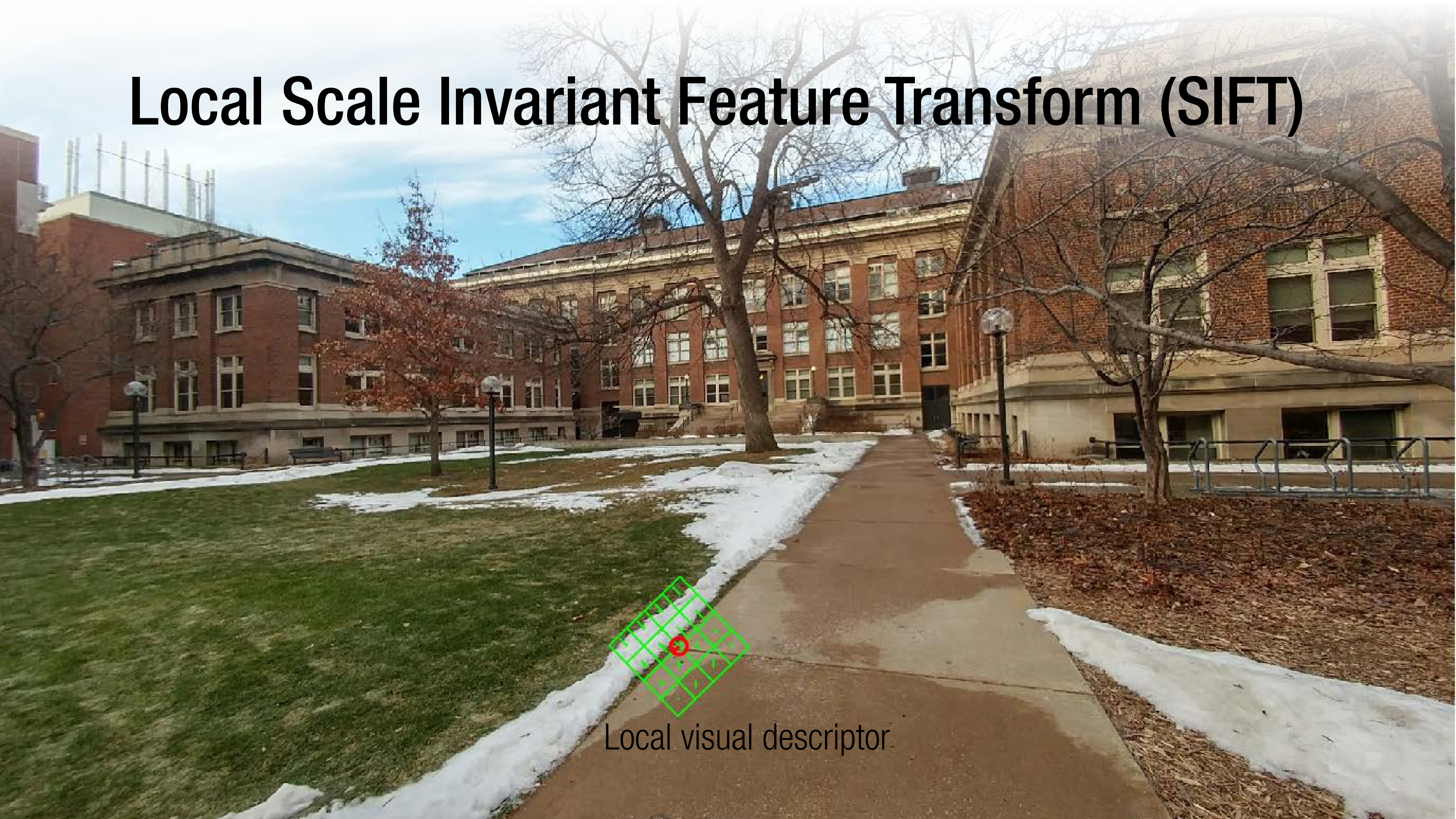
Local Scale Invariant Feature Transform (SIFT)

SIFT automatically finds the optimal scale of feature point and its orientation.

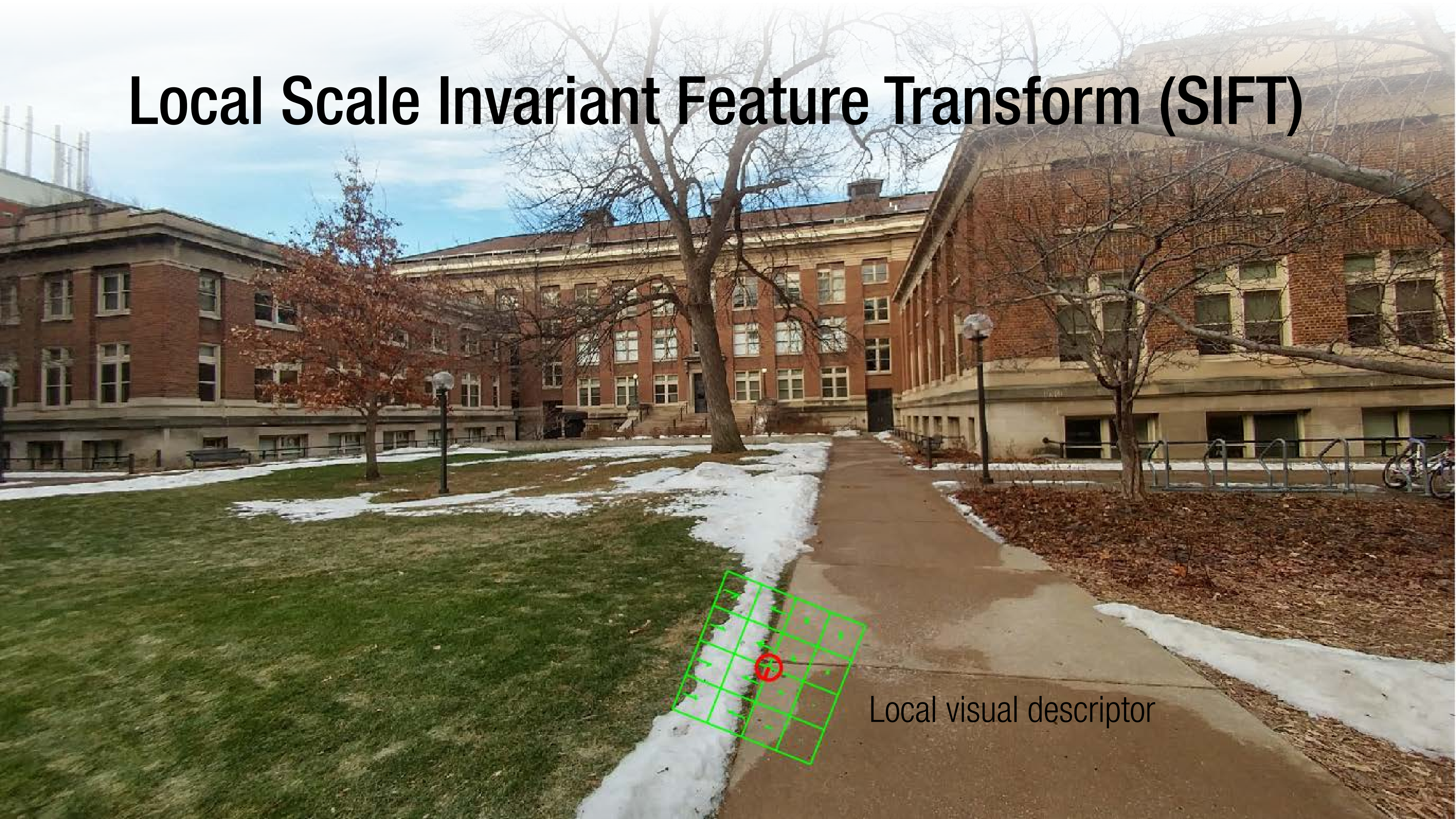
Desired properties:

- Repeatability: the same point is repeatedly detected.
- Discriminativity: the point is unique.
- Orientation aware

Local Scale Invariant Feature Transform (SIFT)

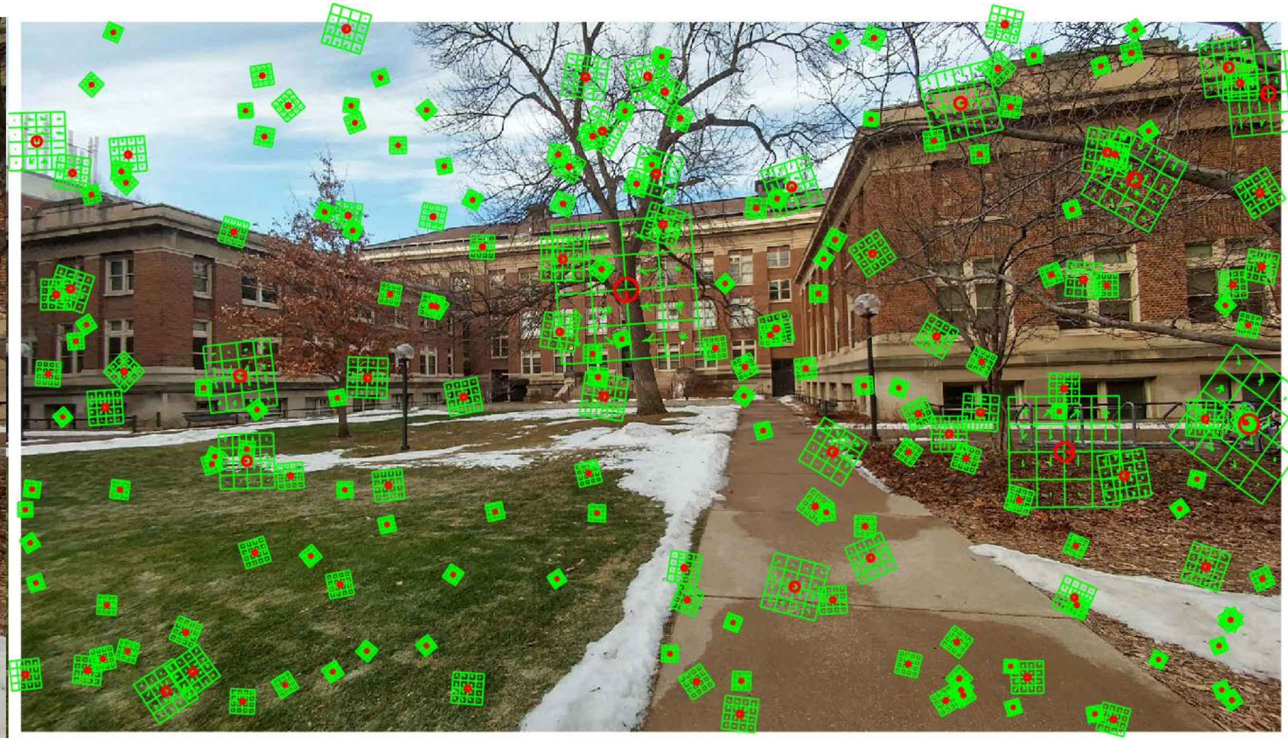


Local Scale Invariant Feature Transform (SIFT)

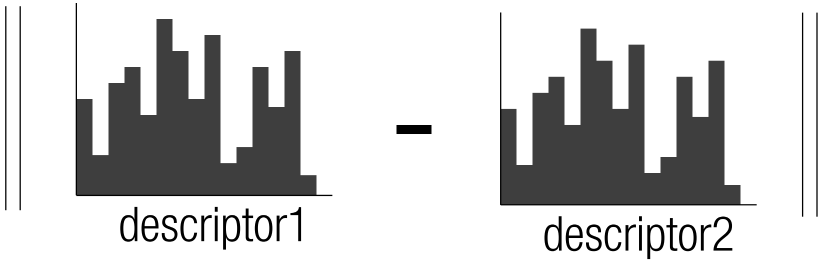


Local Scale Invariant Feature Transform (SIFT)

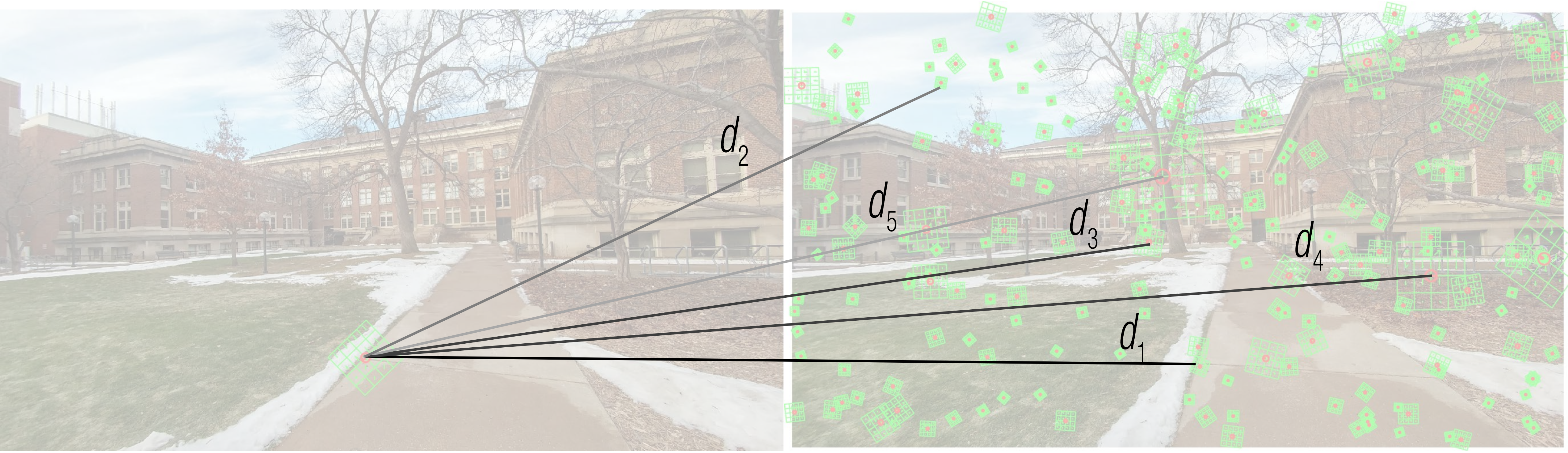
Local Scale Invariant Feature Transform (SIFT)



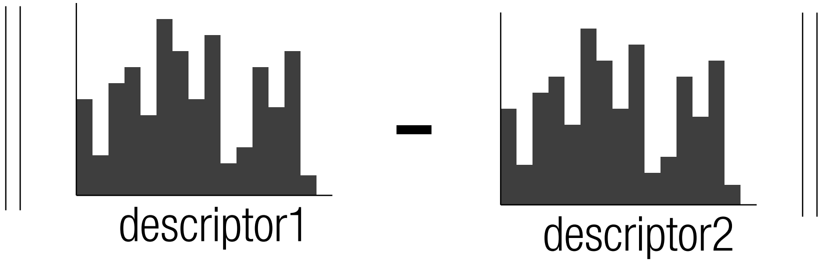
Feature match candidates



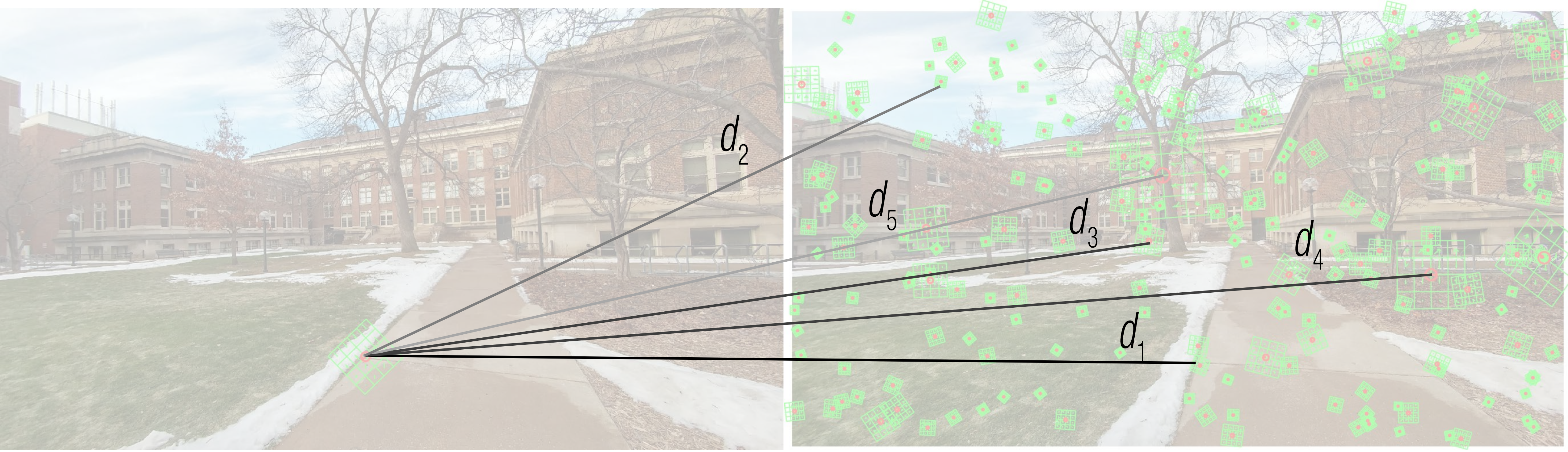
Nearest Neighbor Search



Feature match candidates



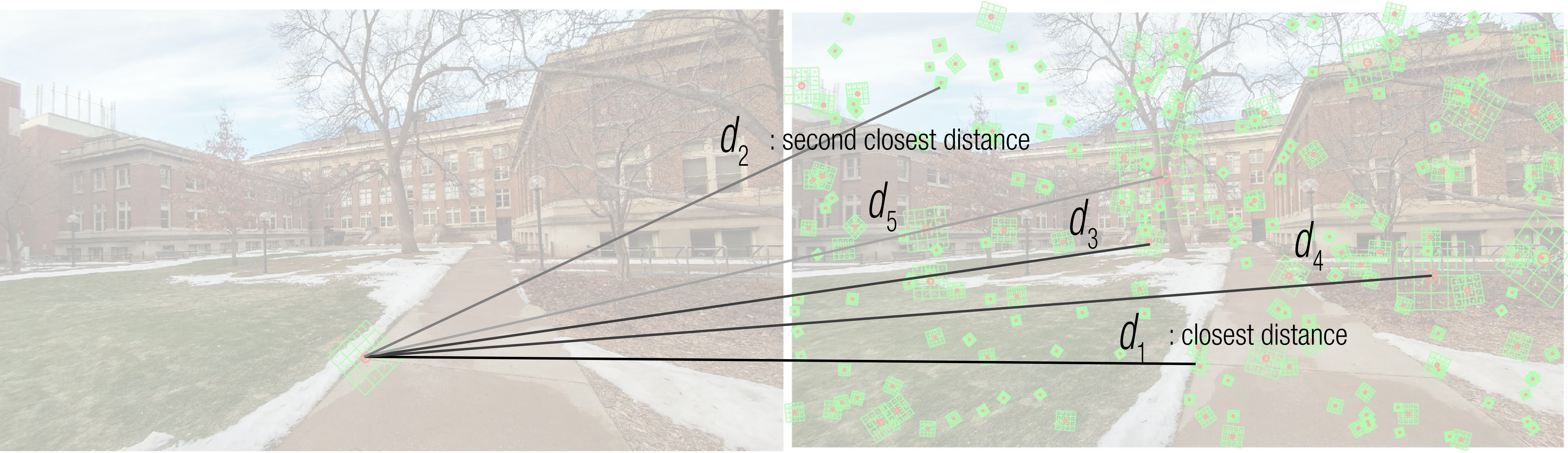
Nearest Neighbor Search



Feature match candidates

Discriminativity: how is the feature point unique?

Nearest Neighbor Search w/ Ratio Test

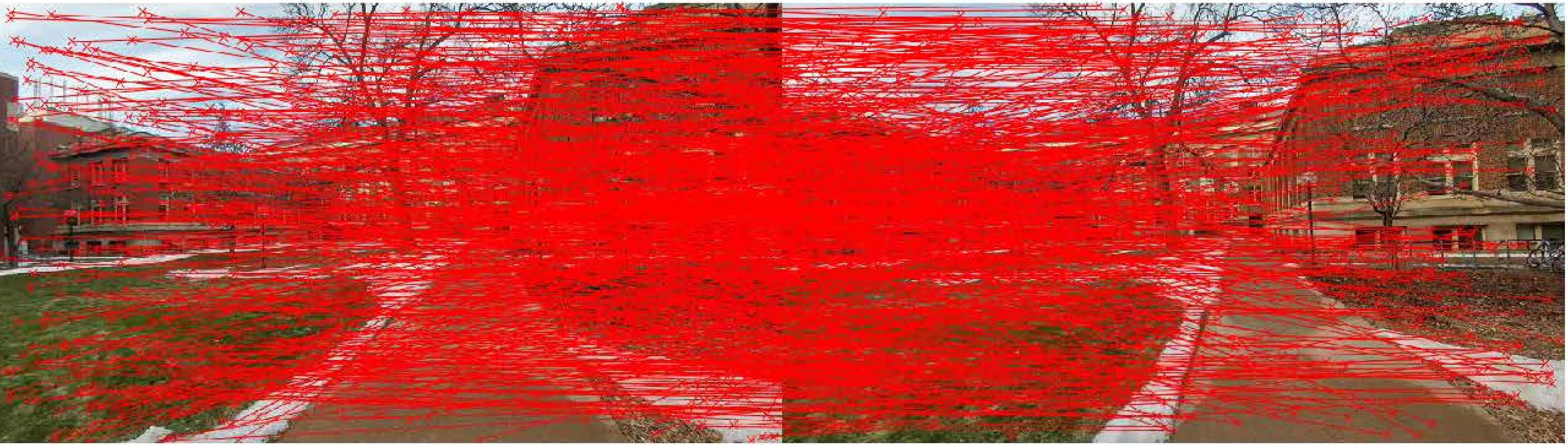


Feature match candidates

Discriminativity: how is the feature point unique?

$$\frac{d_1}{d_2} < 0.7$$

Nearest Neighbor Search w/o Ratio Test

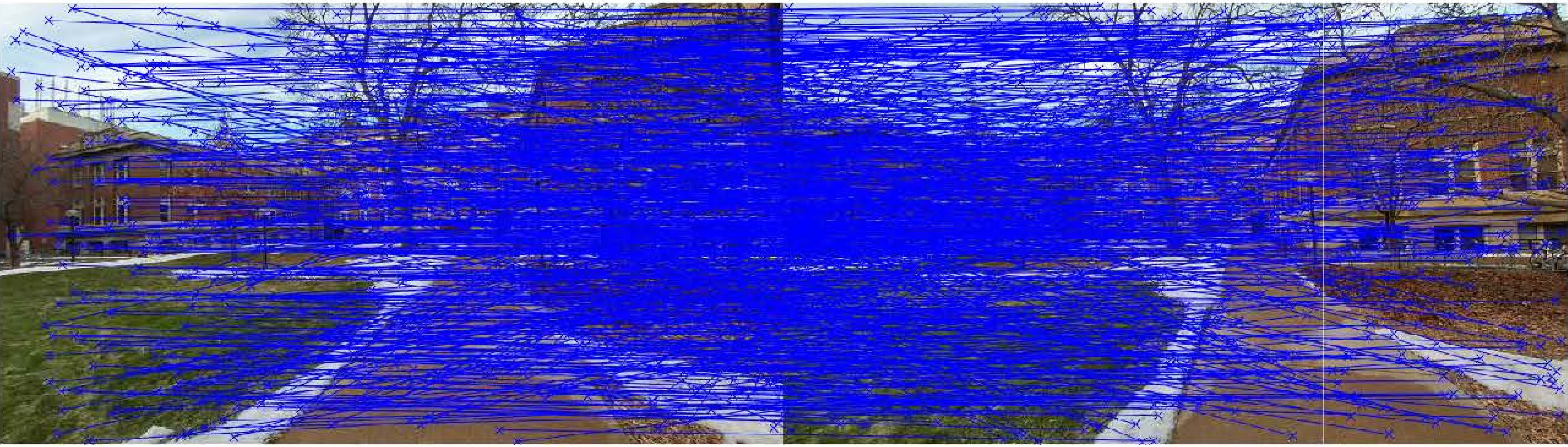


Left image→right image

Nearest Neighbor Search w/ Ratio Test

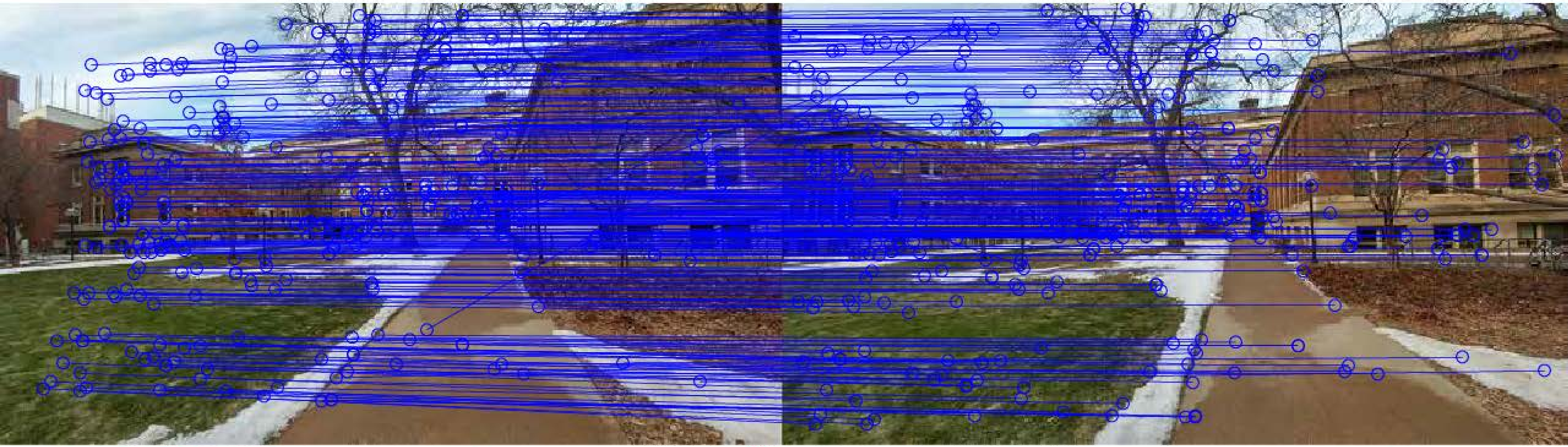
Left image→right image

Nearest Neighbor Search w/o Ratio Test



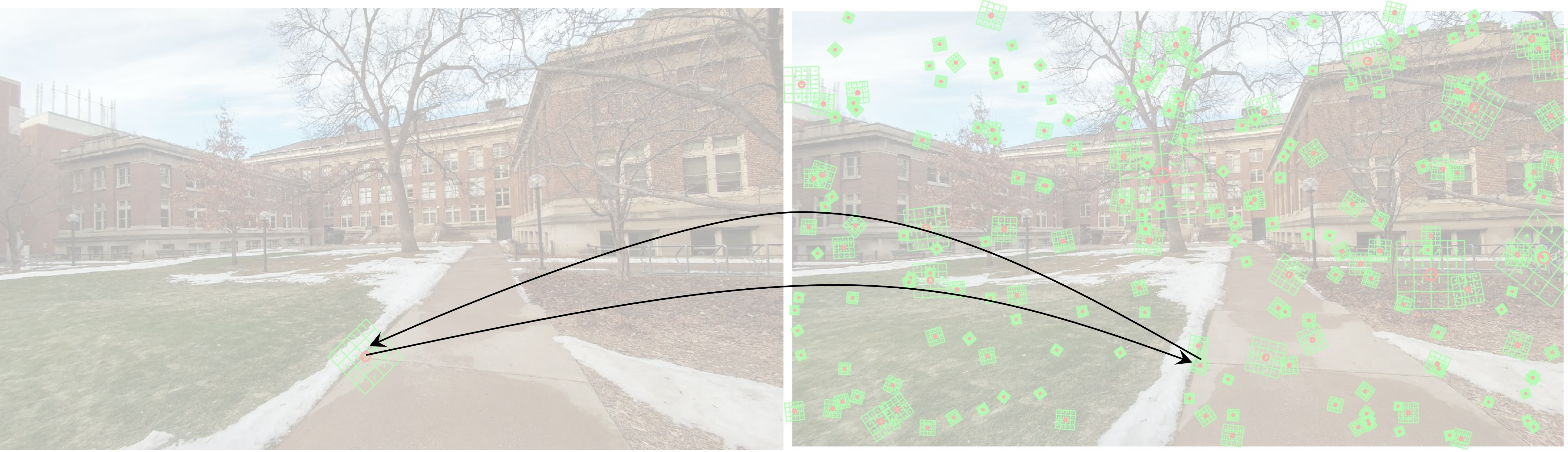
Left image←right image

Nearest Neighbor Search w/ Ratio Test



Left image←right image

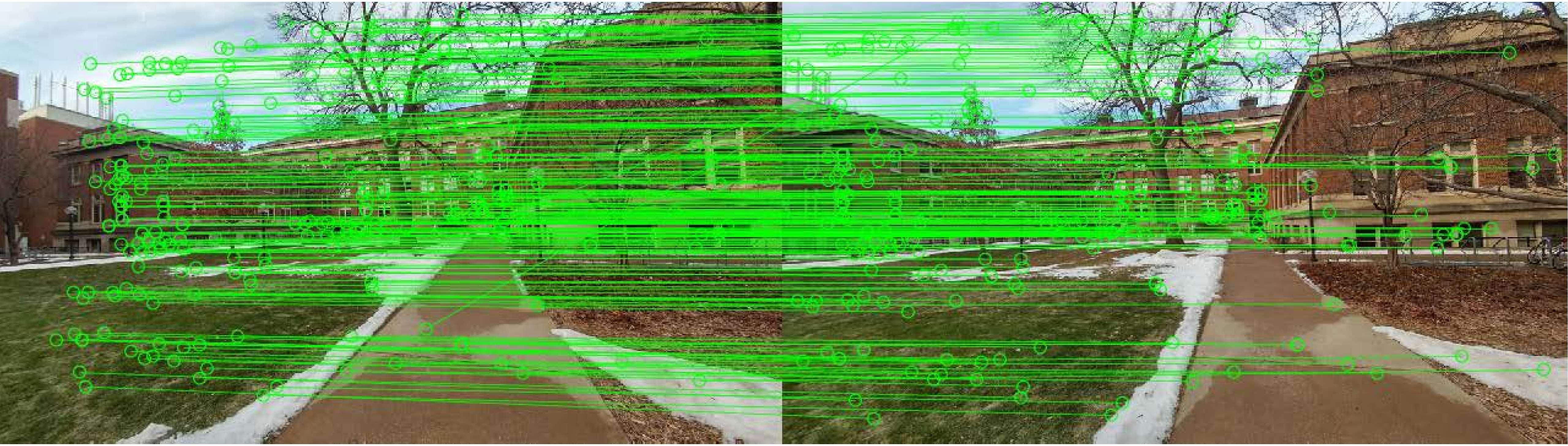
Bi-directional Consistency Check



Feature match candidates

Consistency: would a feature match correspond to each other?

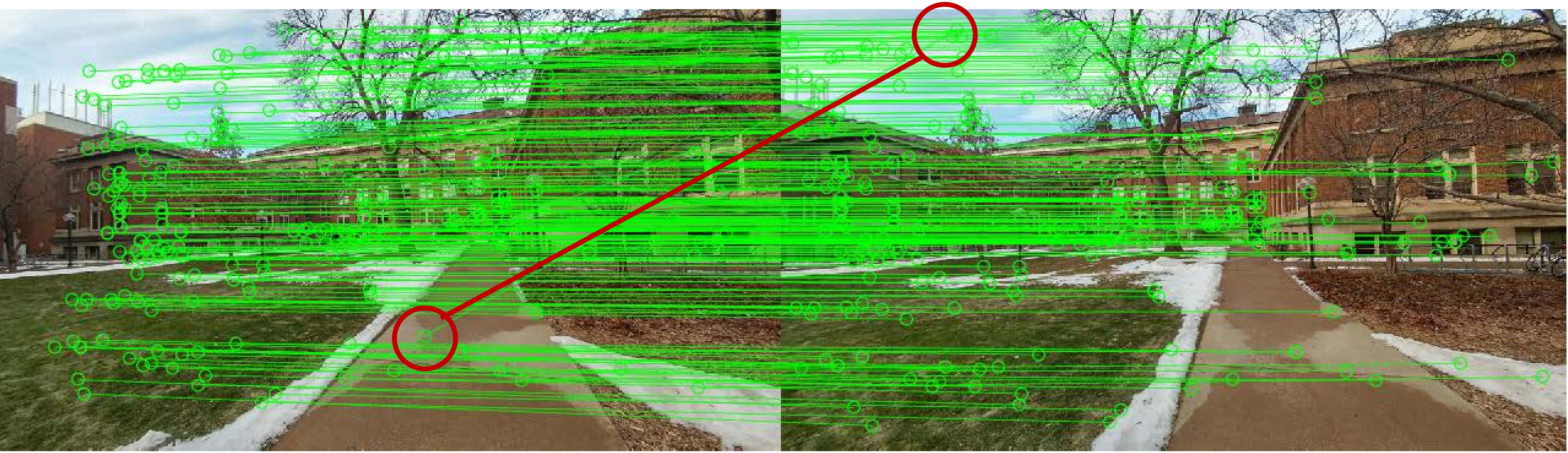
Bi-directional Consistency Check



RANSAC: Random Sample Consensus: Linear Least Squares

$$\underbrace{\begin{bmatrix} u_x & u_y & 1 & & & & -u_xv_x & -u_yv_x & -v_x \\ & & & u_x & u_y & & -u_xv_y & -u_yv_y & -v_y \end{bmatrix}}_{\mathbf{A}\ (2\text{x}9)} \underbrace{\begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ \vdots \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} 0 \\ 0 \end{bmatrix}}_{\mathbf{0}}$$

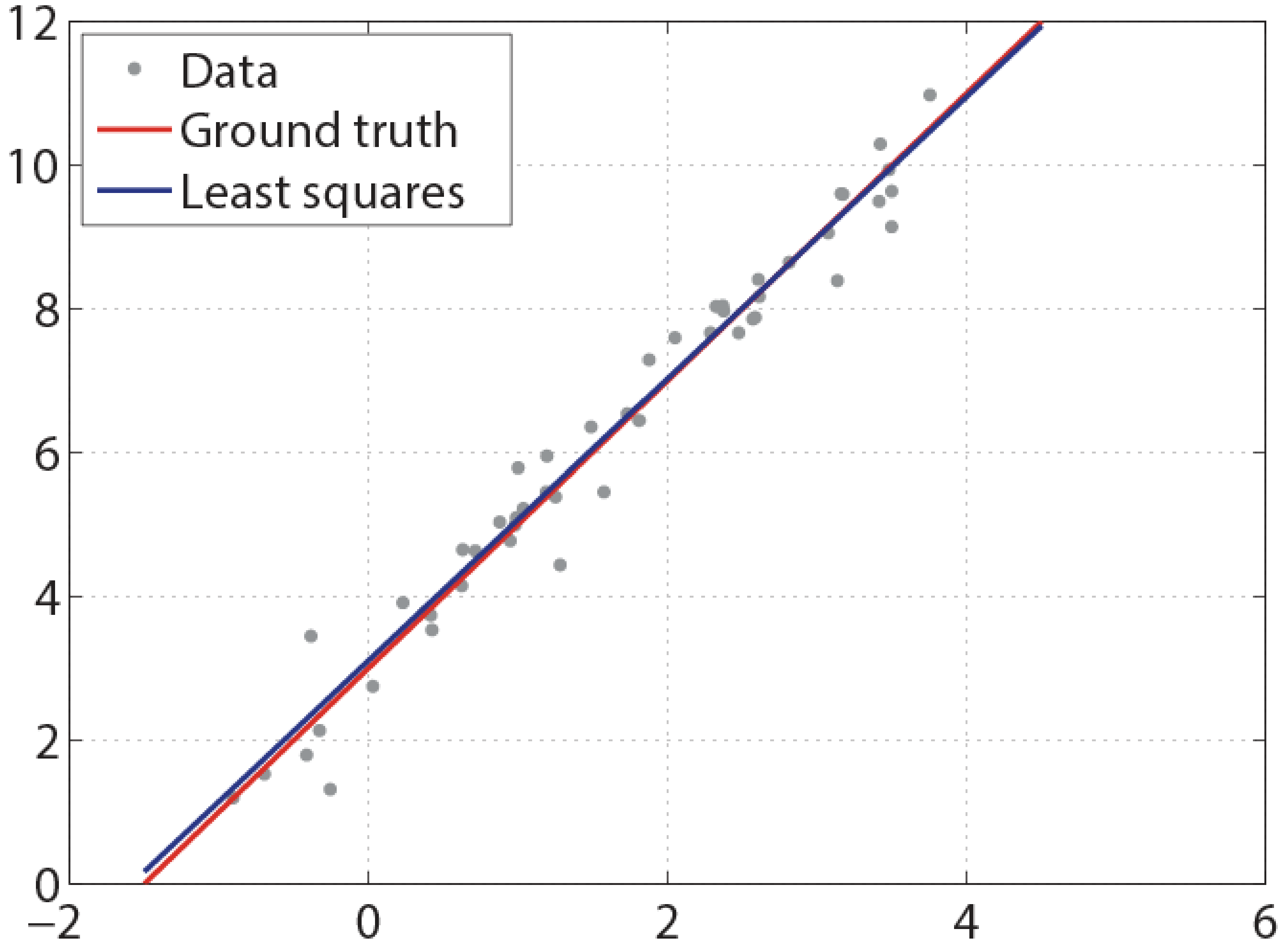
Outlier?

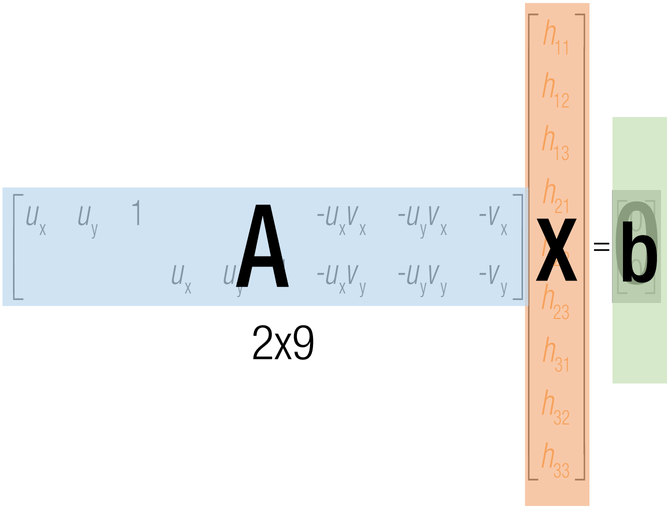


Martin A. Fischler and Robert C. Bolles (June 1981).

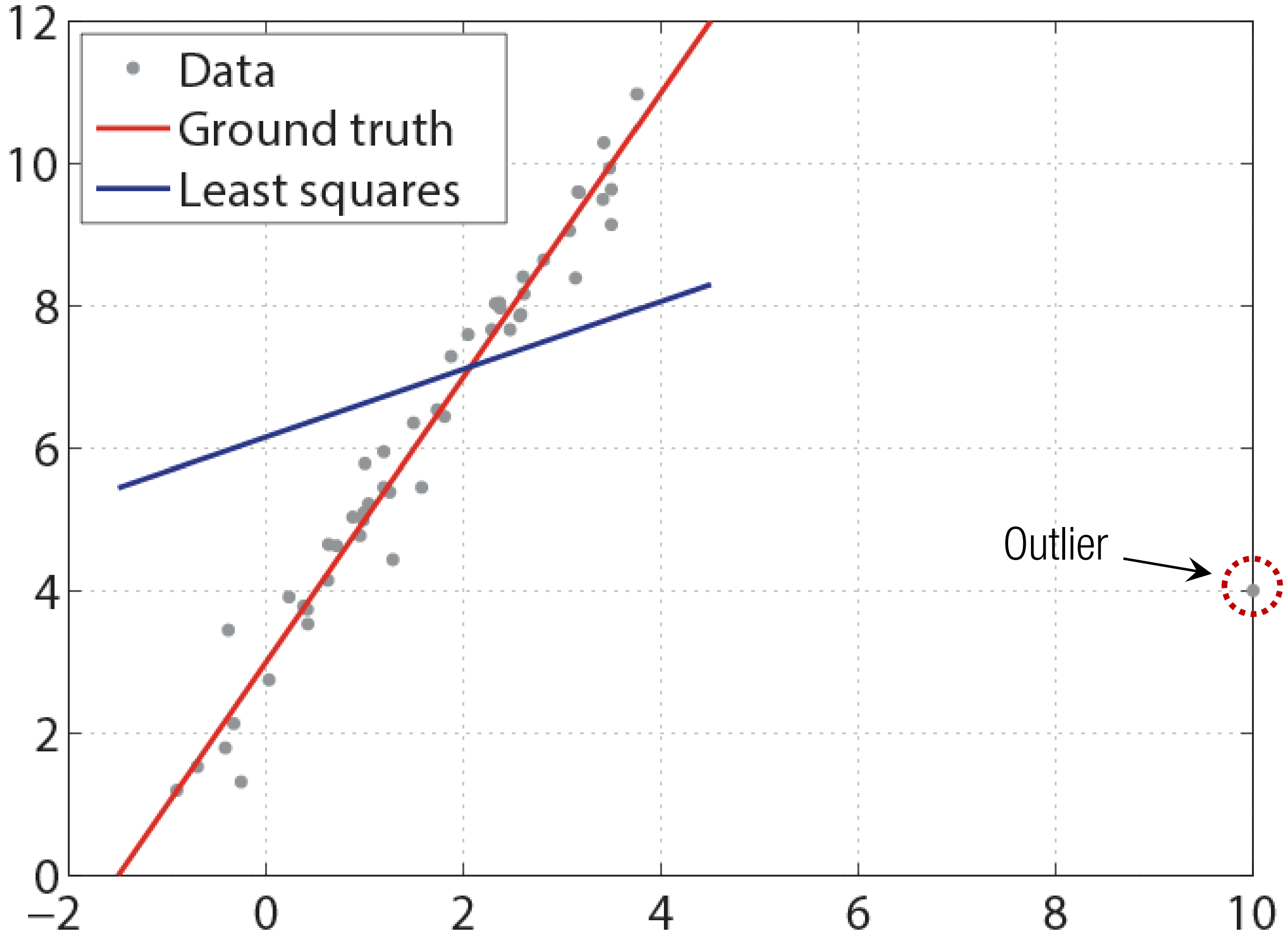
"Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography".

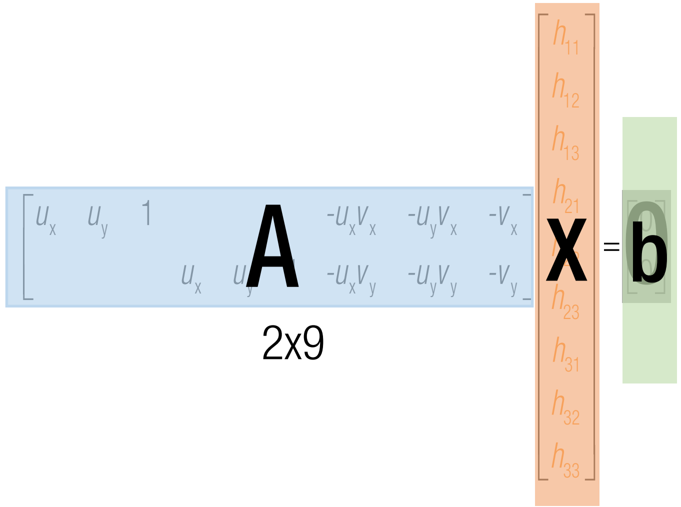
Recall: Line Fitting (Ax=b)

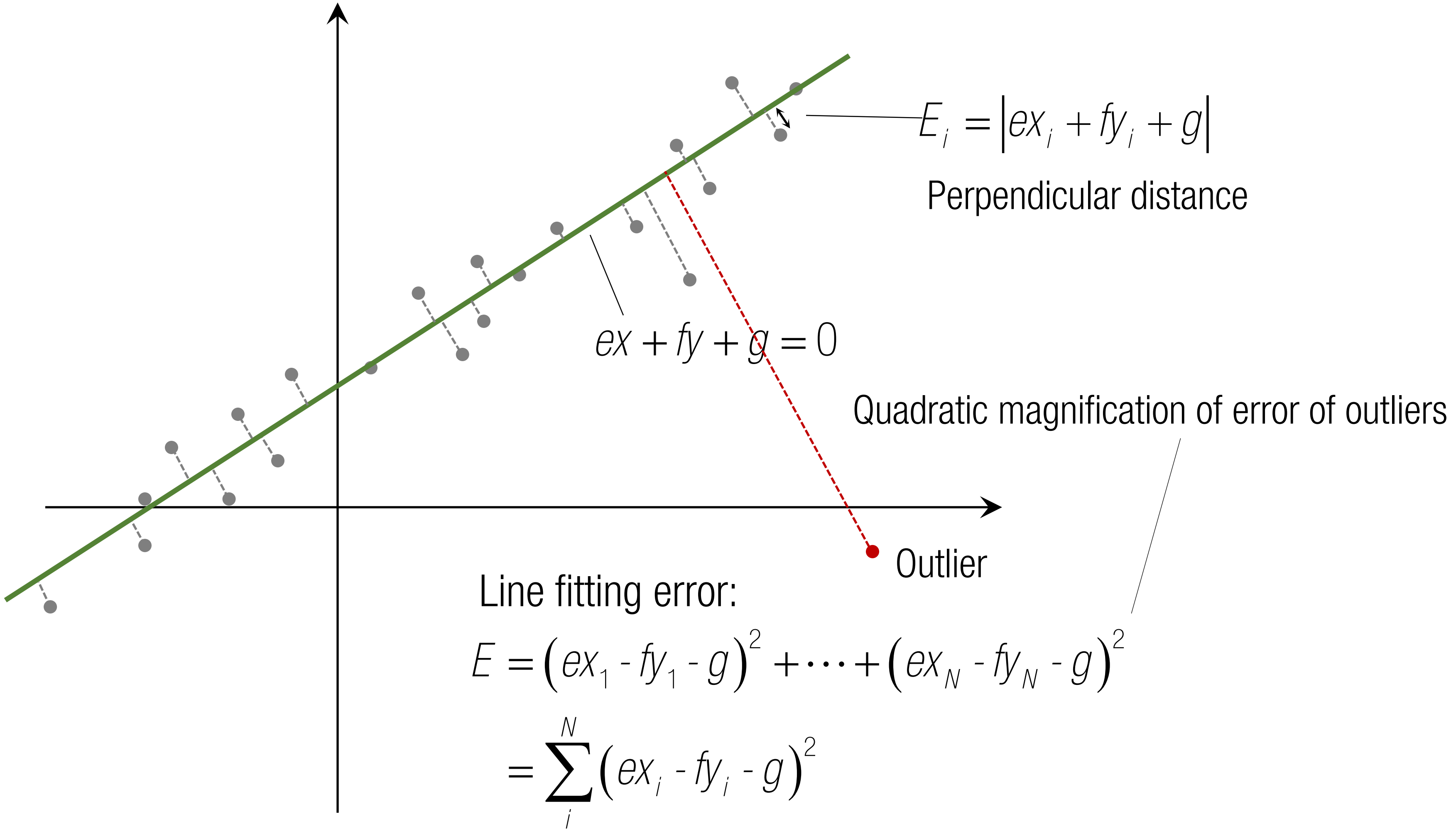




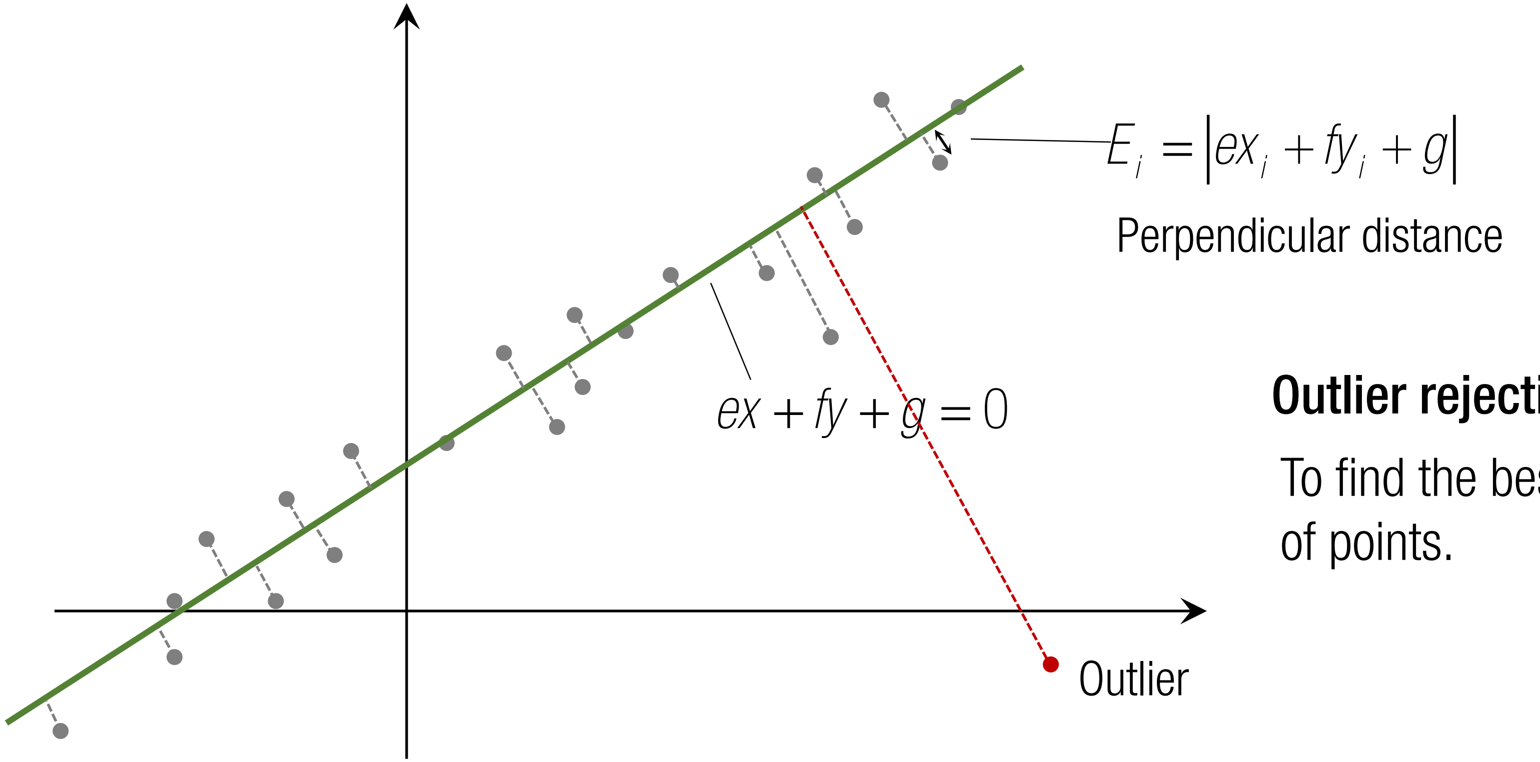
Outlier





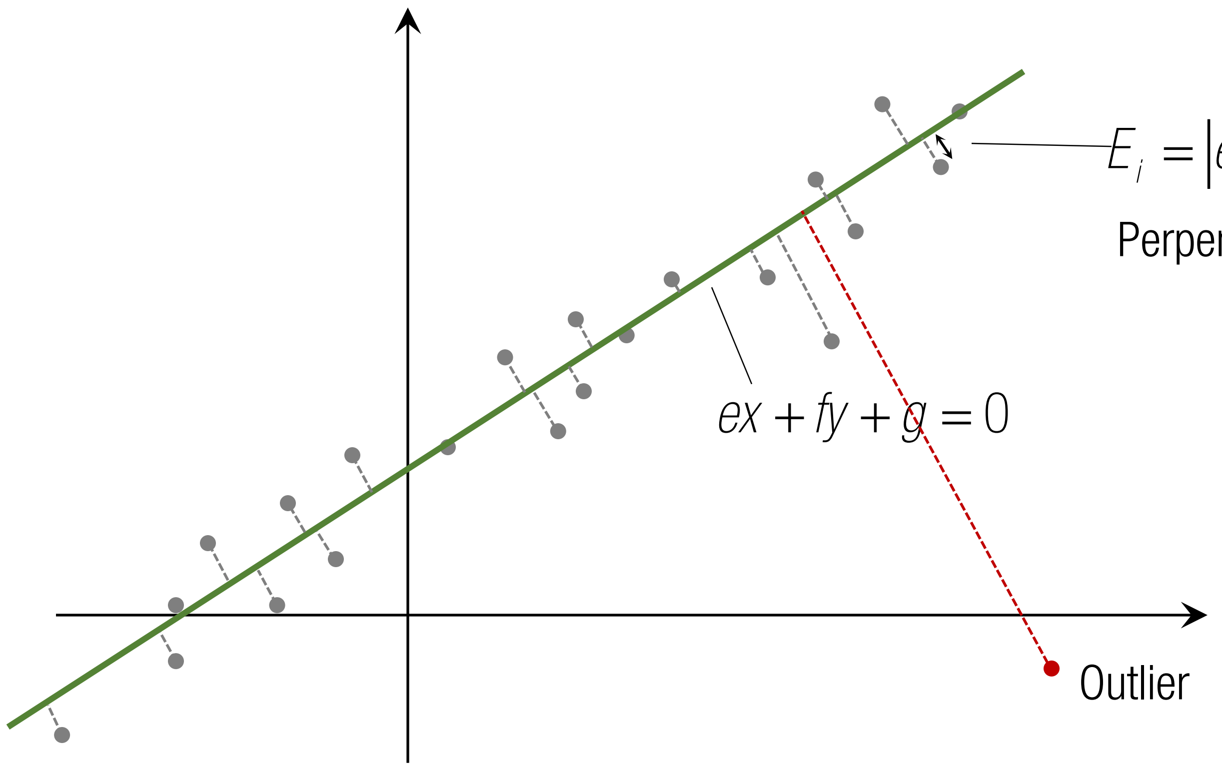


$E_i = \left| e x_i + f y_i + g \right|$
Perpendicular distance
$ex + fy + g = 0$
Quadratic magnification of error of outliers
Outlier
Line fitting error:
$E = \left(ex_1 - fy_1 - g\right)^2 + \cdots + \left(ex_N - fy_N - g\right)^2$
$= \sum_i^N \left(ex_i - fy_i - g\right)^2$



Outlier rejection strategy:

To find the best line that explanes the maximum number of points.



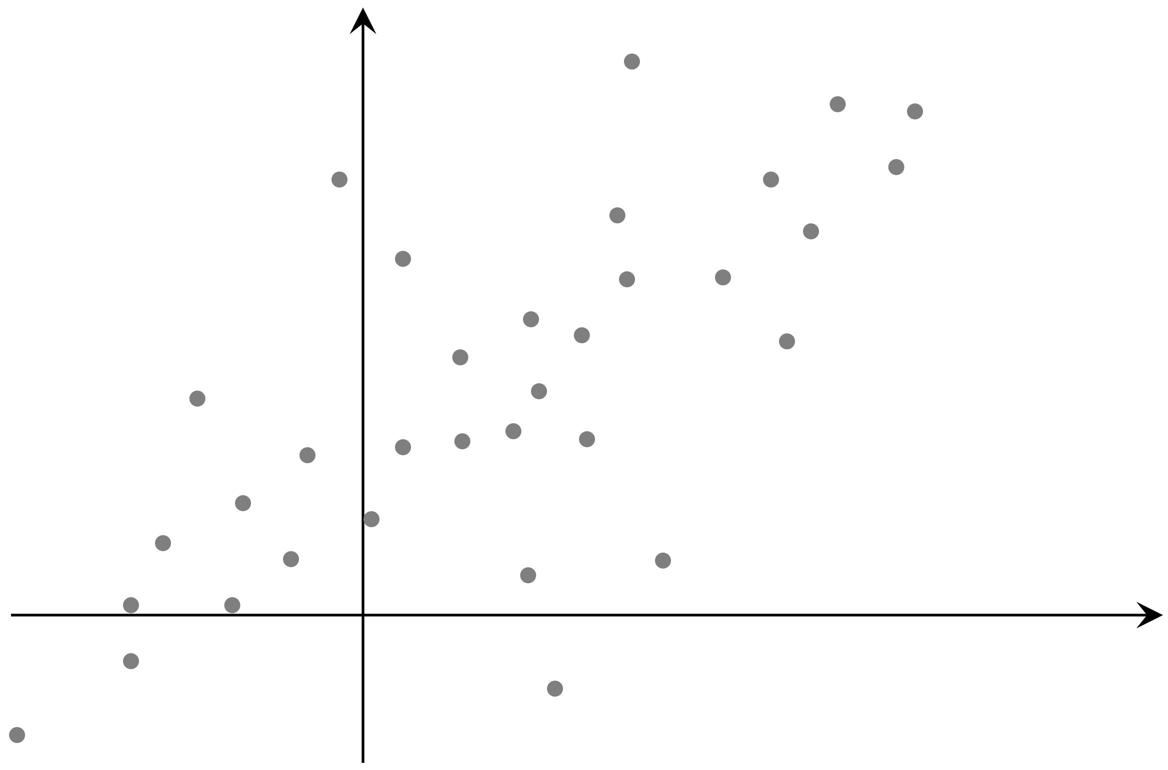
Outlier rejection strategy:

To find the best line that explanes the maximum number of points.

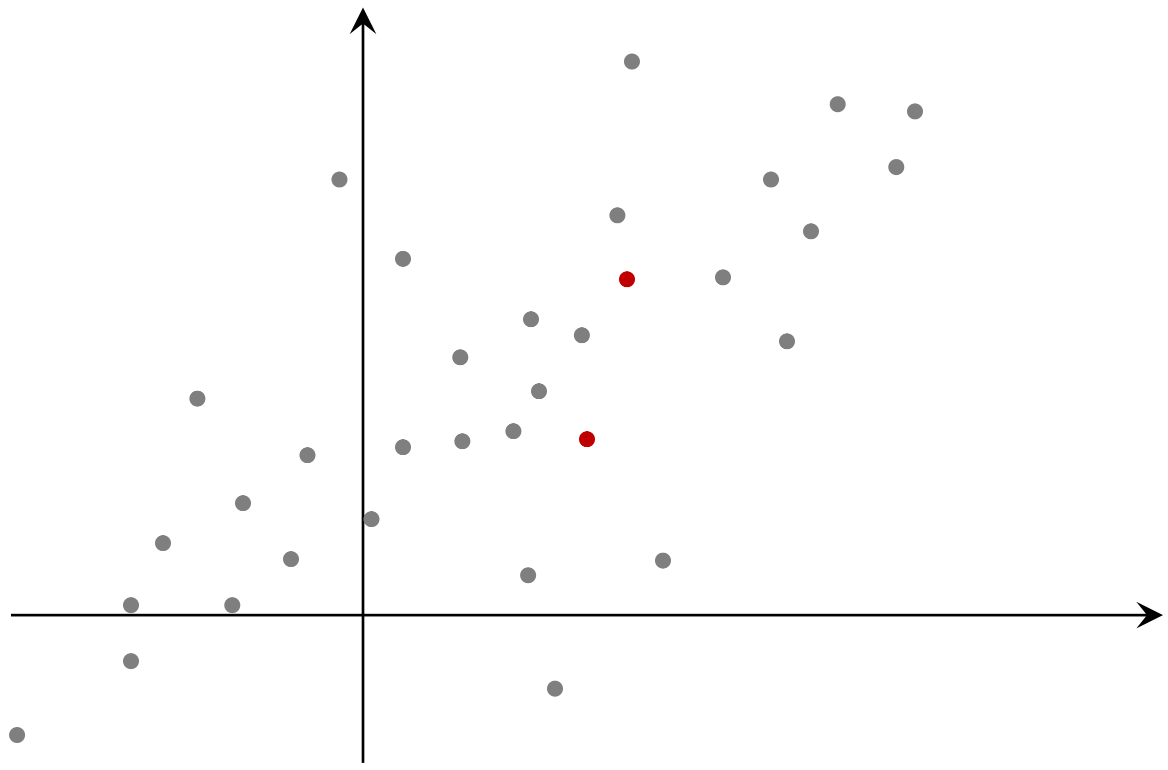
Assumptions:

1. Majority of good samples agree with the underlying model (good apples are same and simple.).

2. Bad samples does not consistently agree with a single model
(all bad apples are different and complicated.).

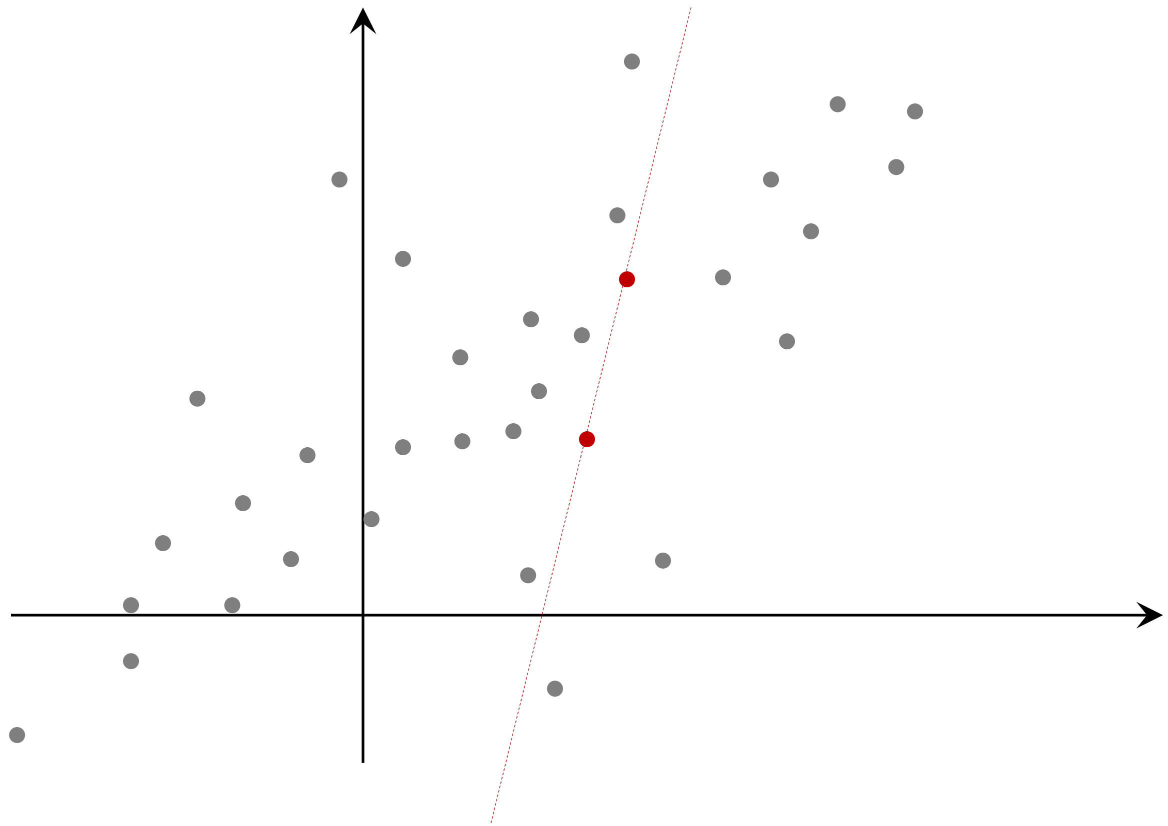


RANSAC: Random Sample Consensus



1. Random sampling

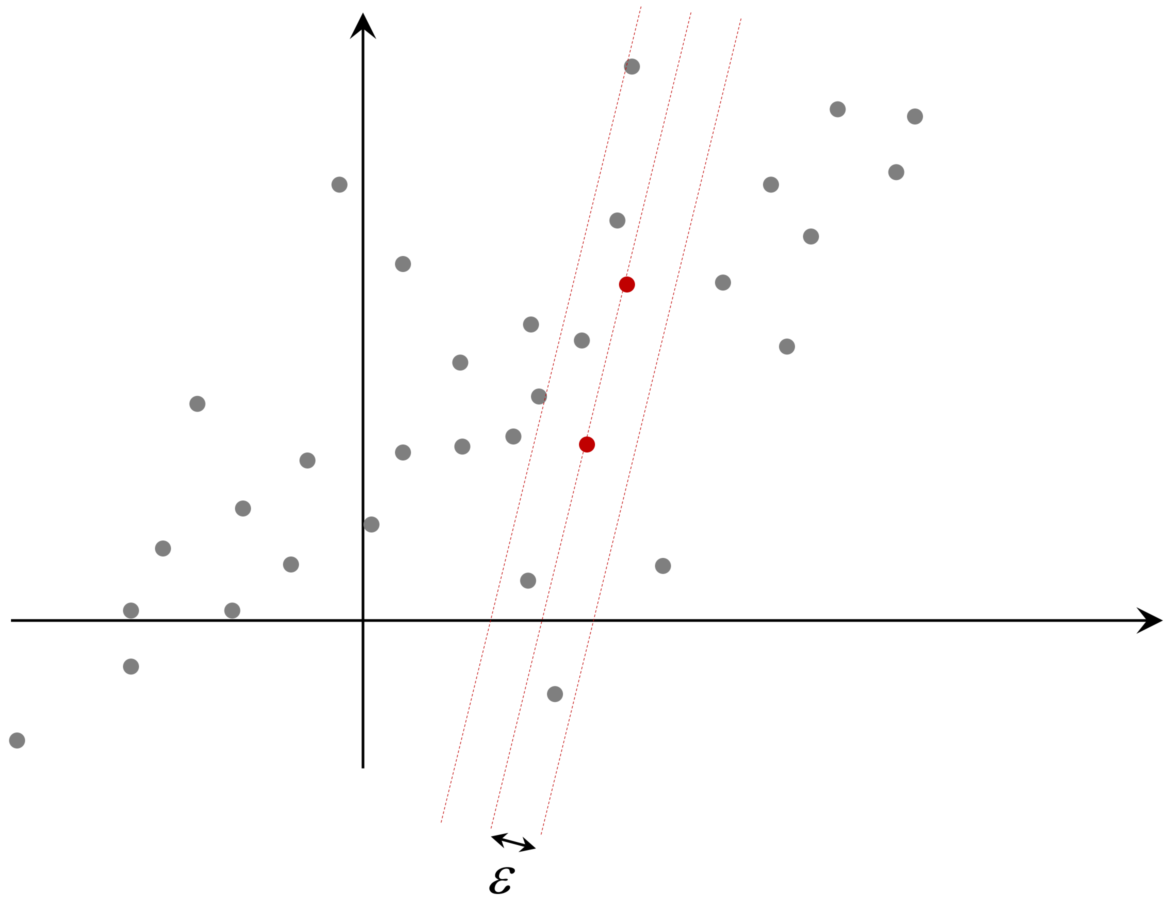
RANSAC: Random Sample Consensus



1. Random sampling

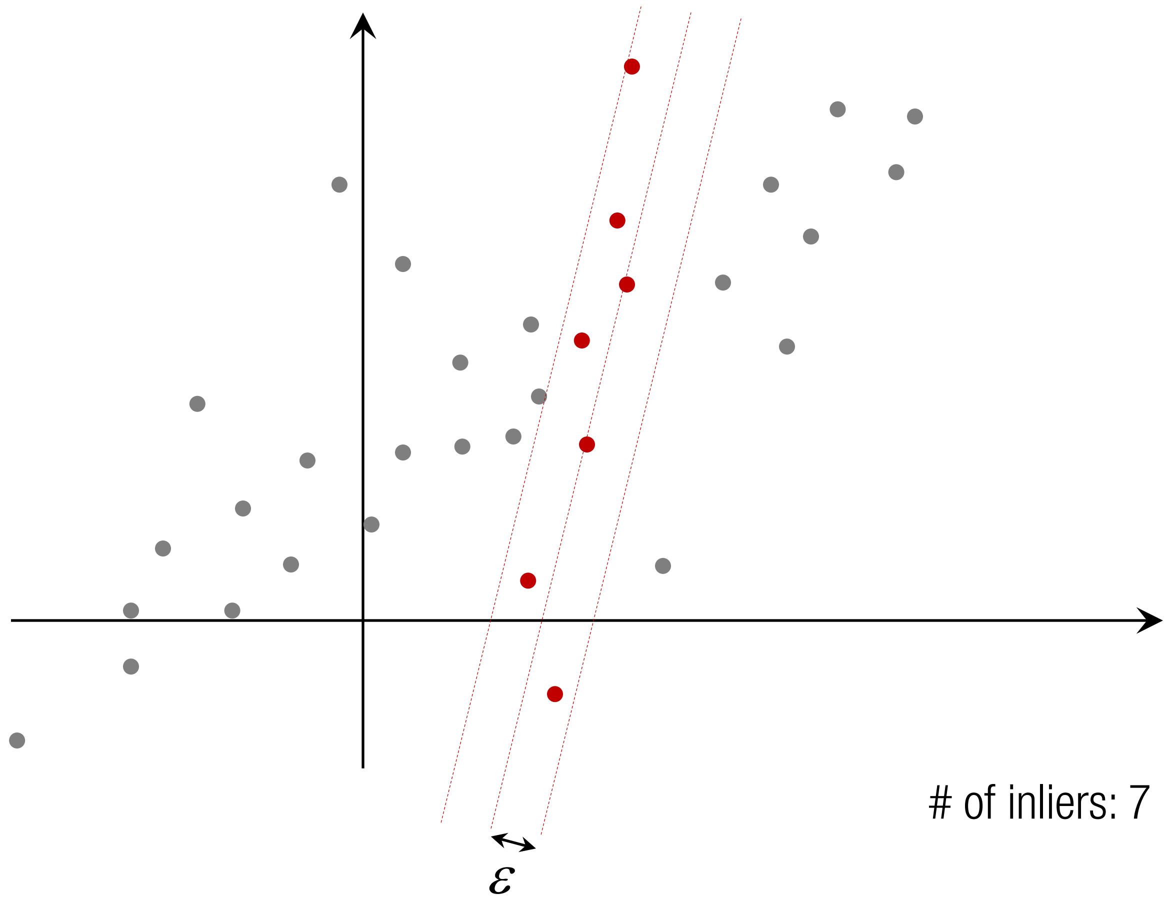
2. Model building

RANSAC: Random Sample Consensus



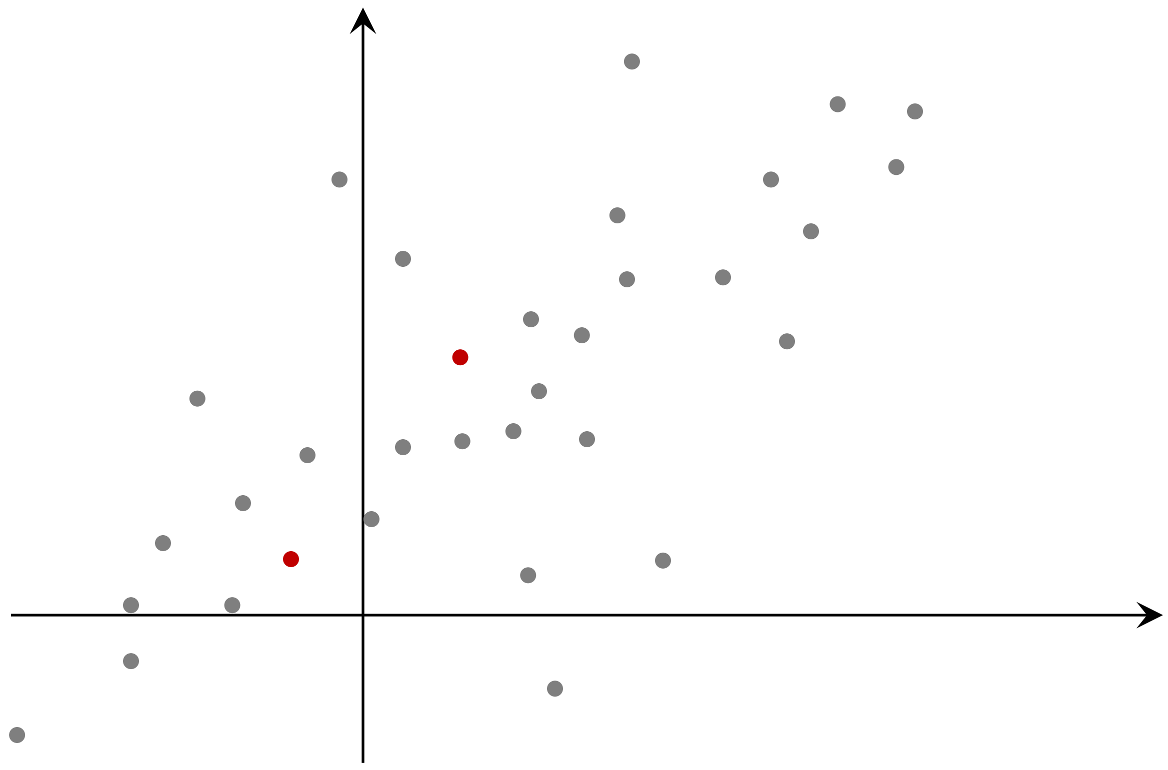
1. Random sampling
2. Model building
3. Thresholding

RANSAC: Random Sample Consensus



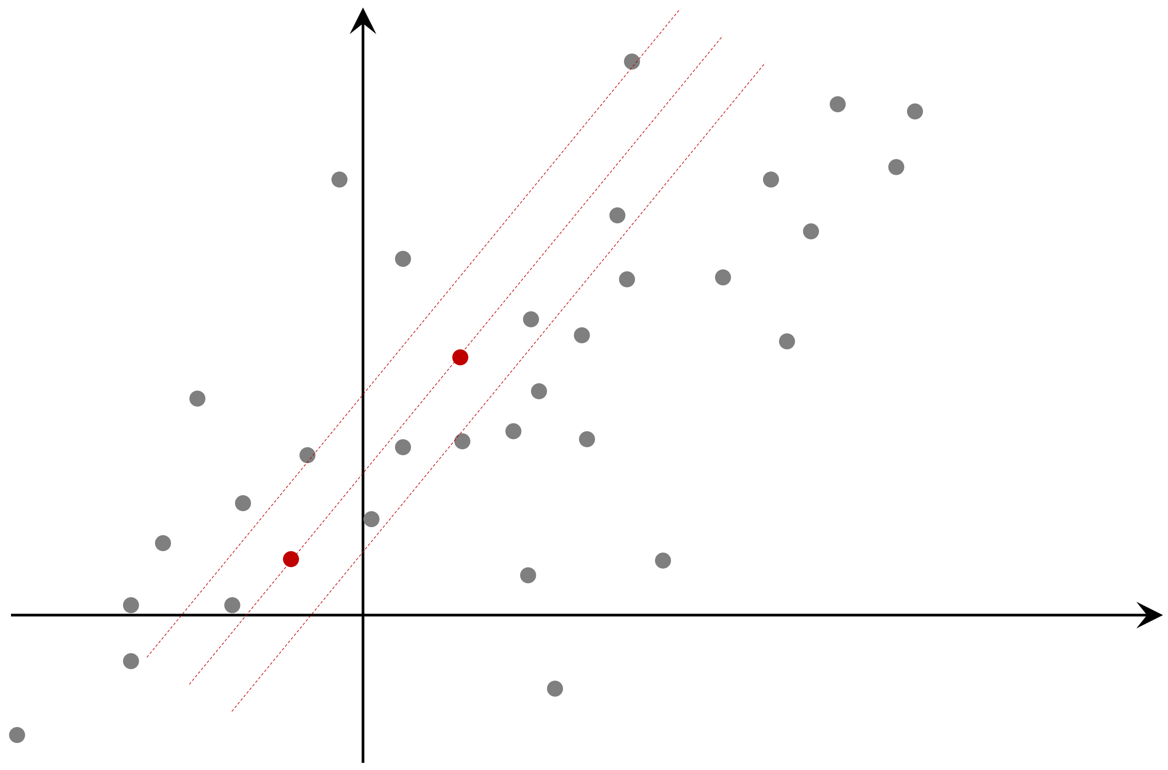
1. Random sampling
2. Model building
3. Thresholding
4. Inlier counting

RANSAC: Random Sample Consensus



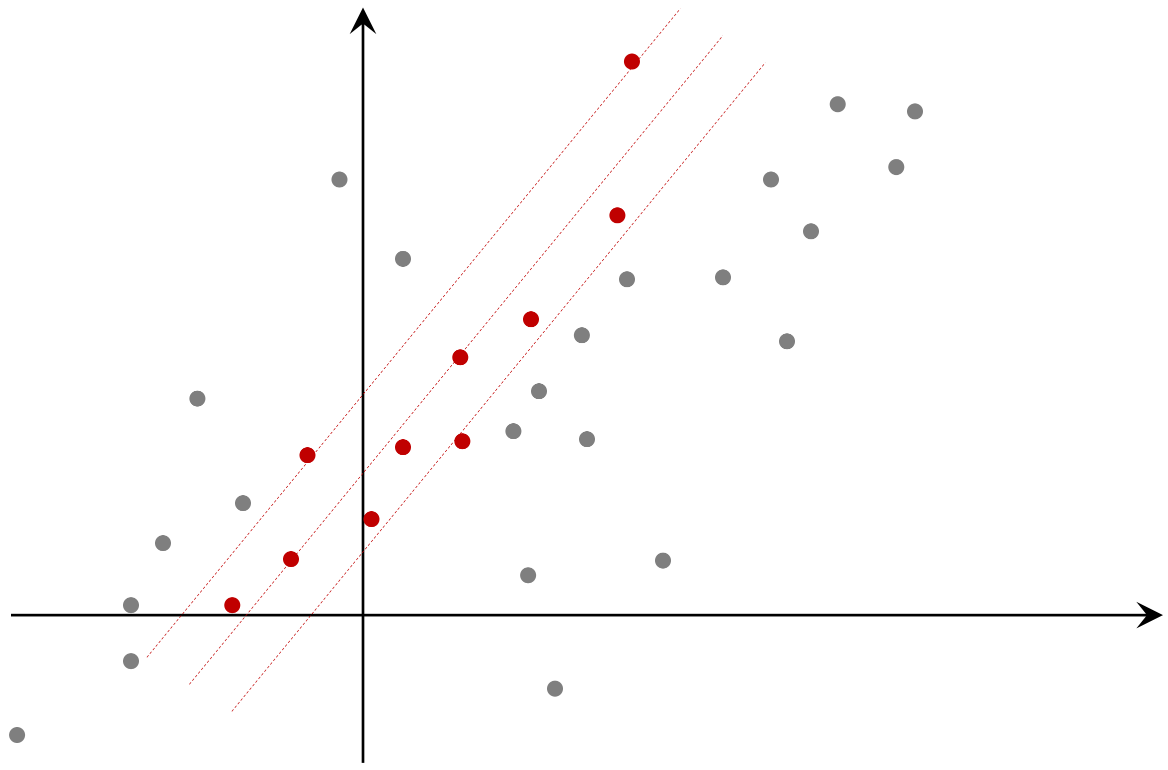
1. Random sampling
2. Model building
3. Thresholding
4. Inlier counting

RANSAC: Random Sample Consensus



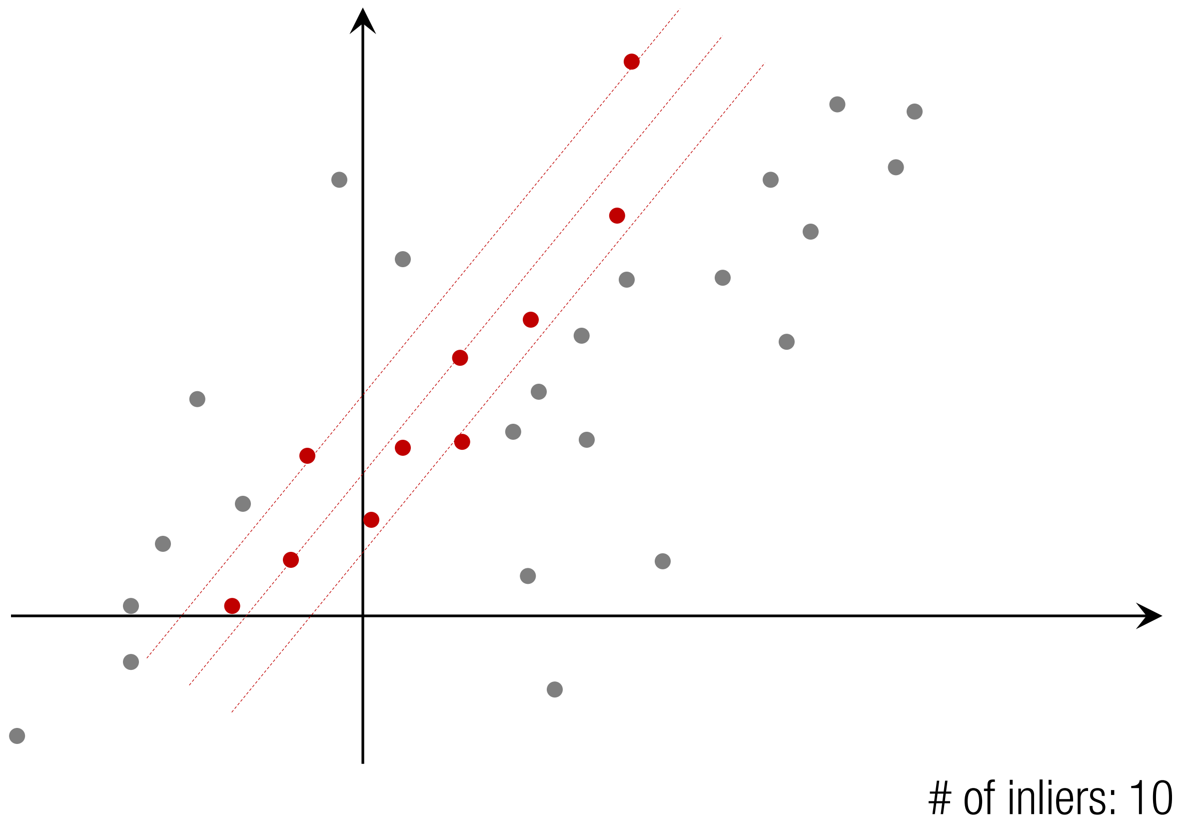
1. Random sampling
2. Model building
3. Thresholding
4. Inlier counting

RANSAC: Random Sample Consensus



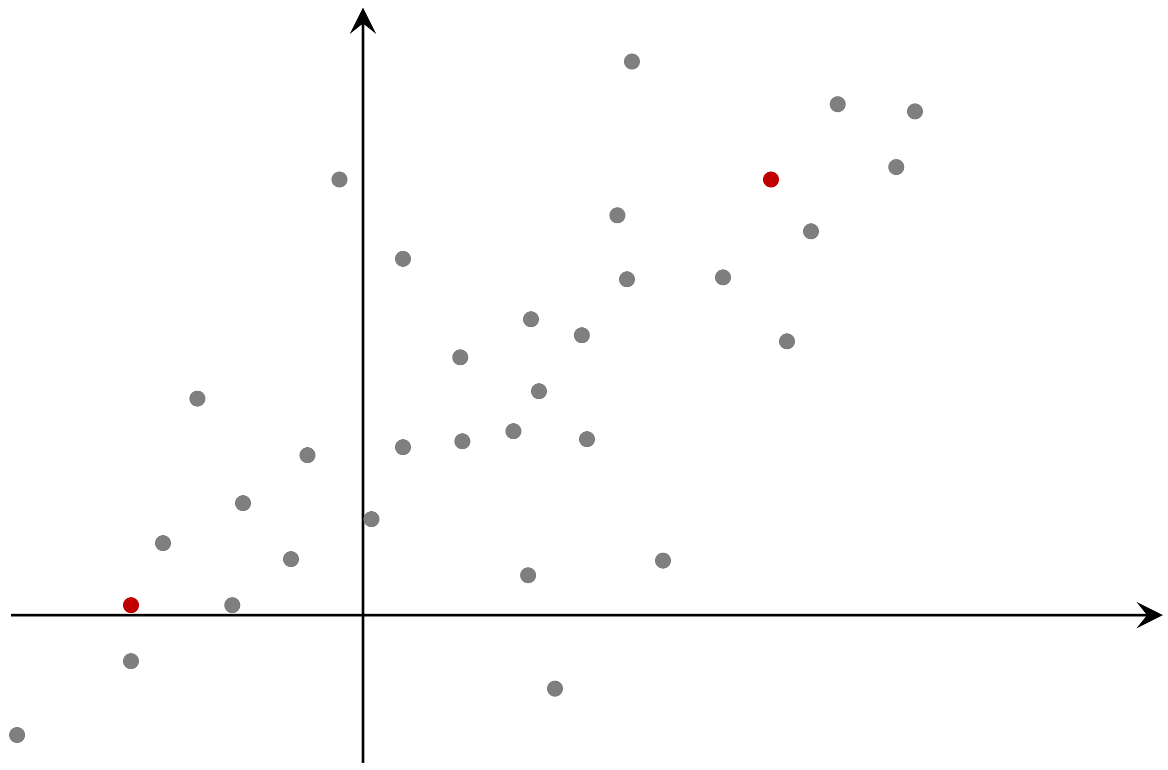
1. Random sampling
2. Model building
3. Thresholding
4. Inlier counting

RANSAC: Random Sample Consensus



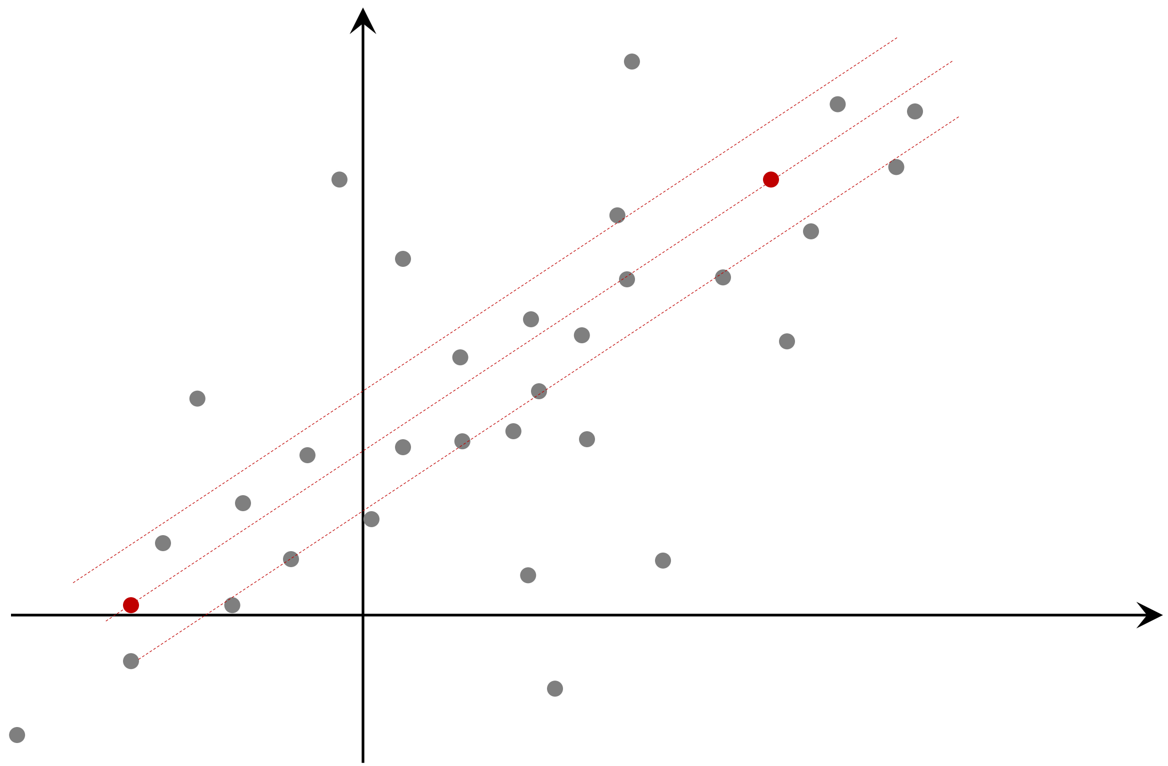
1. Random sampling
2. Model building
3. Thresholding
4. Inlier counting

RANSAC: Random Sample Consensus



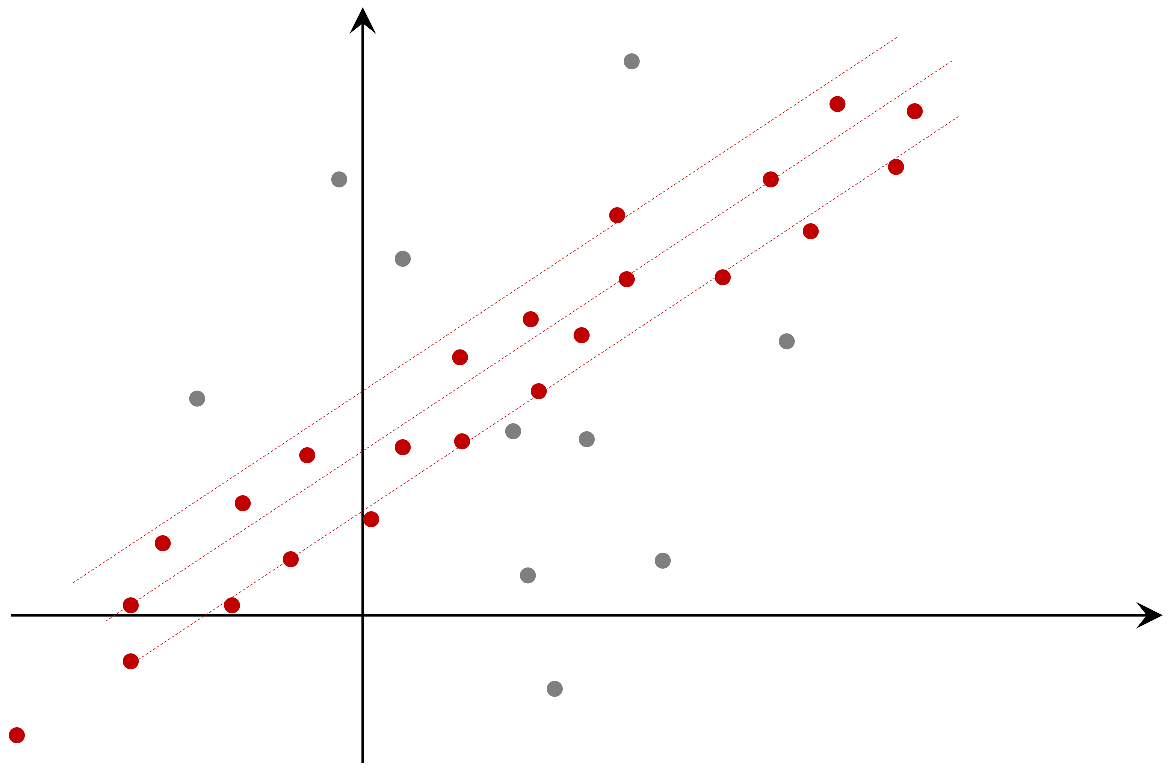
1. Random sampling
2. Model building
3. Thresholding
4. Inlier counting

RANSAC: Random Sample Consensus



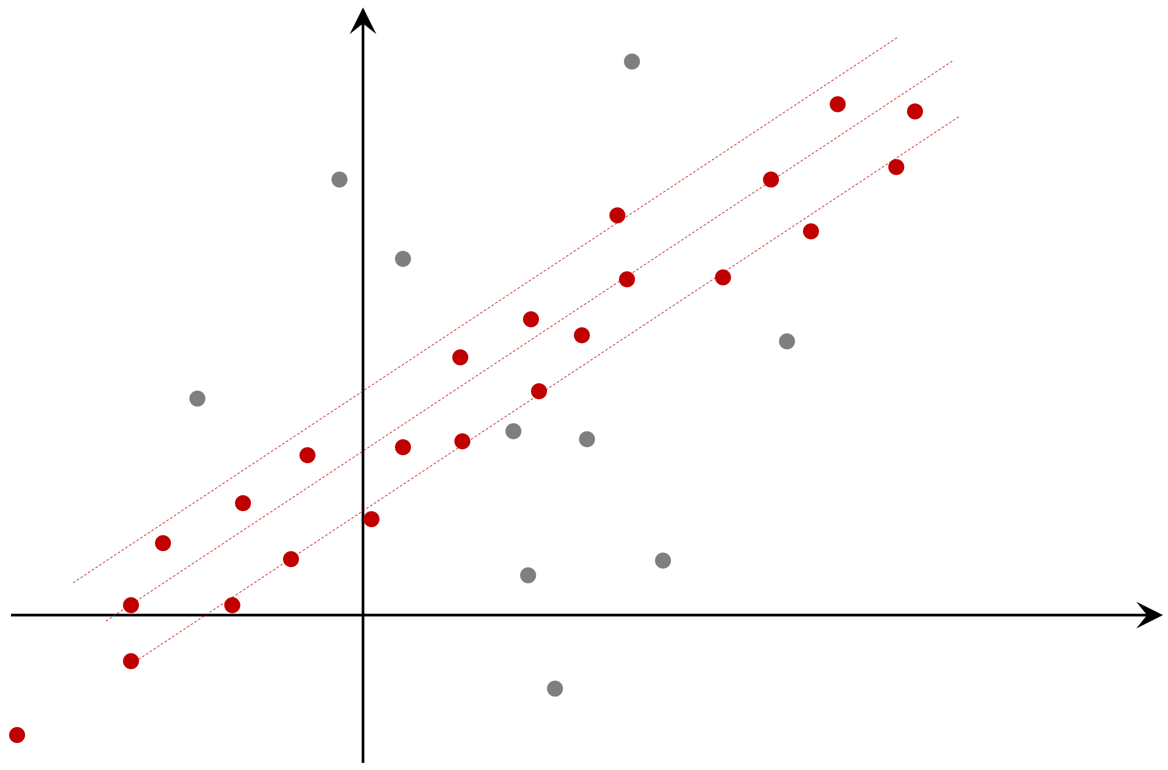
1. Random sampling
2. Model building
3. Thresholding
4. Inlier counting

RANSAC: Random Sample Consensus



1. Random sampling
2. Model building
3. Thresholding
4. Inlier counting

RANSAC: Random Sample Consensus

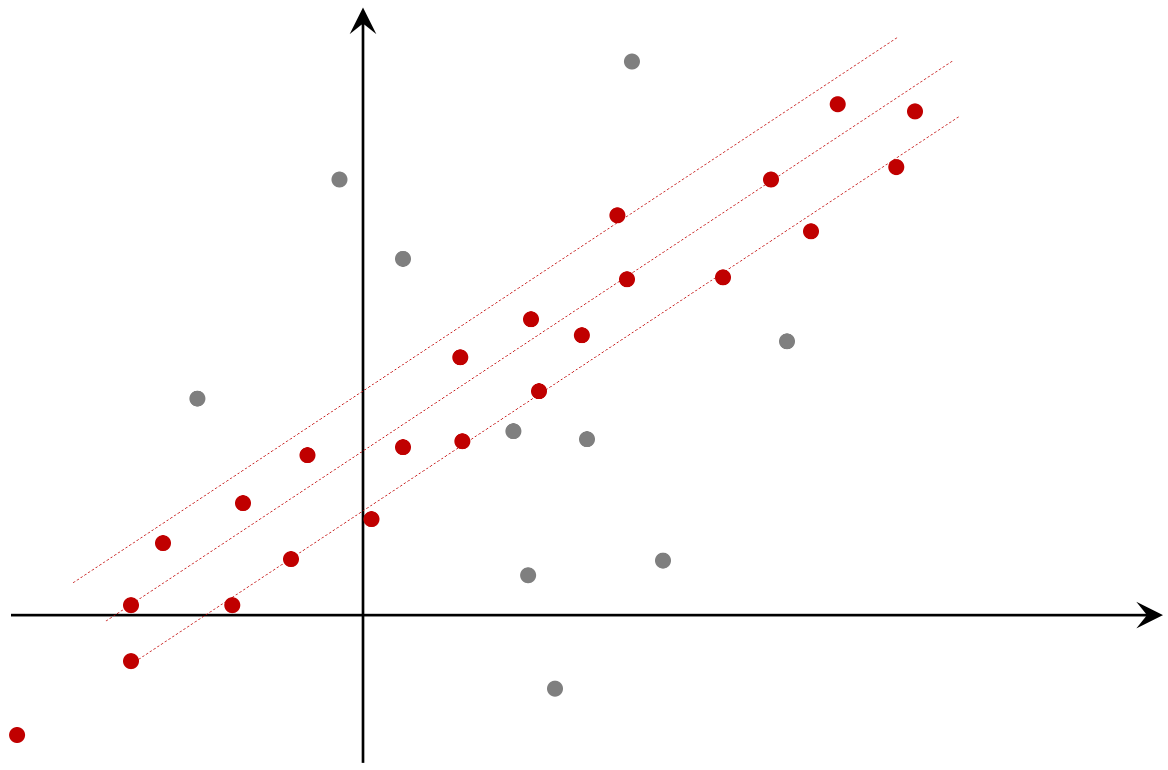


1. Random sampling
2. Model building
3. Thresholding
4. Inlier counting

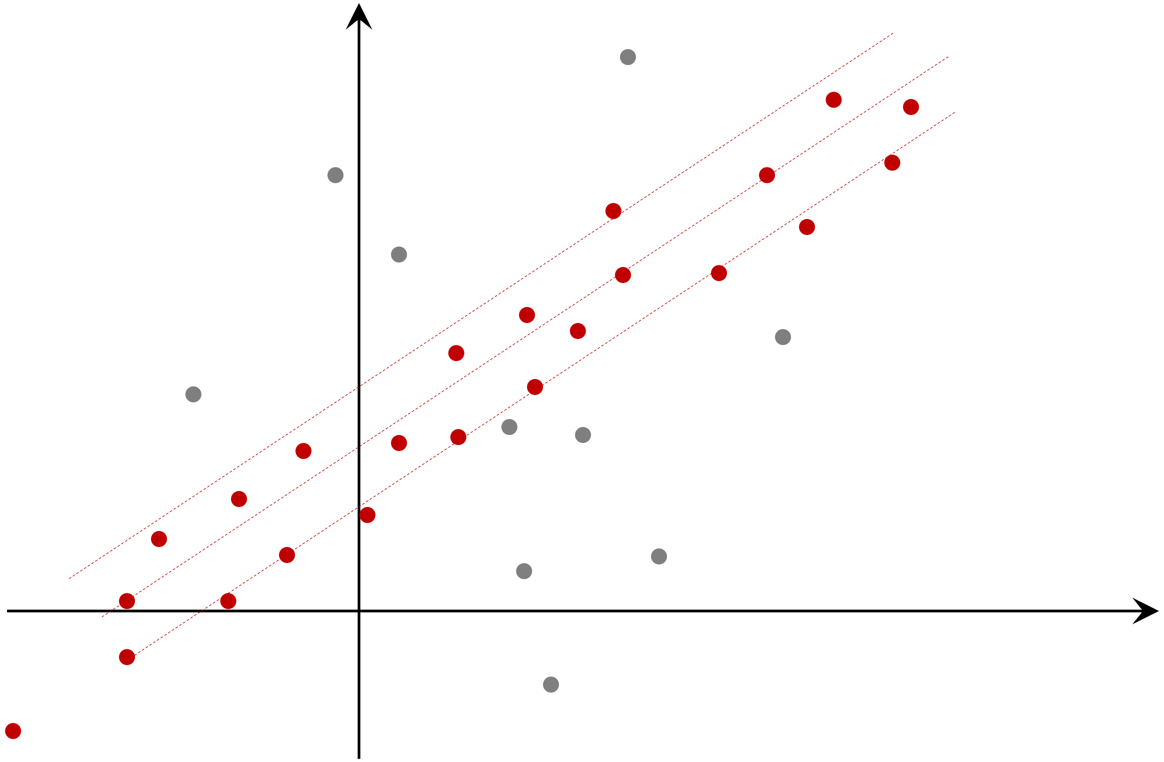
# of inliers: 23

Maximum number of inliers

RANSAC: Random Sample Consensus



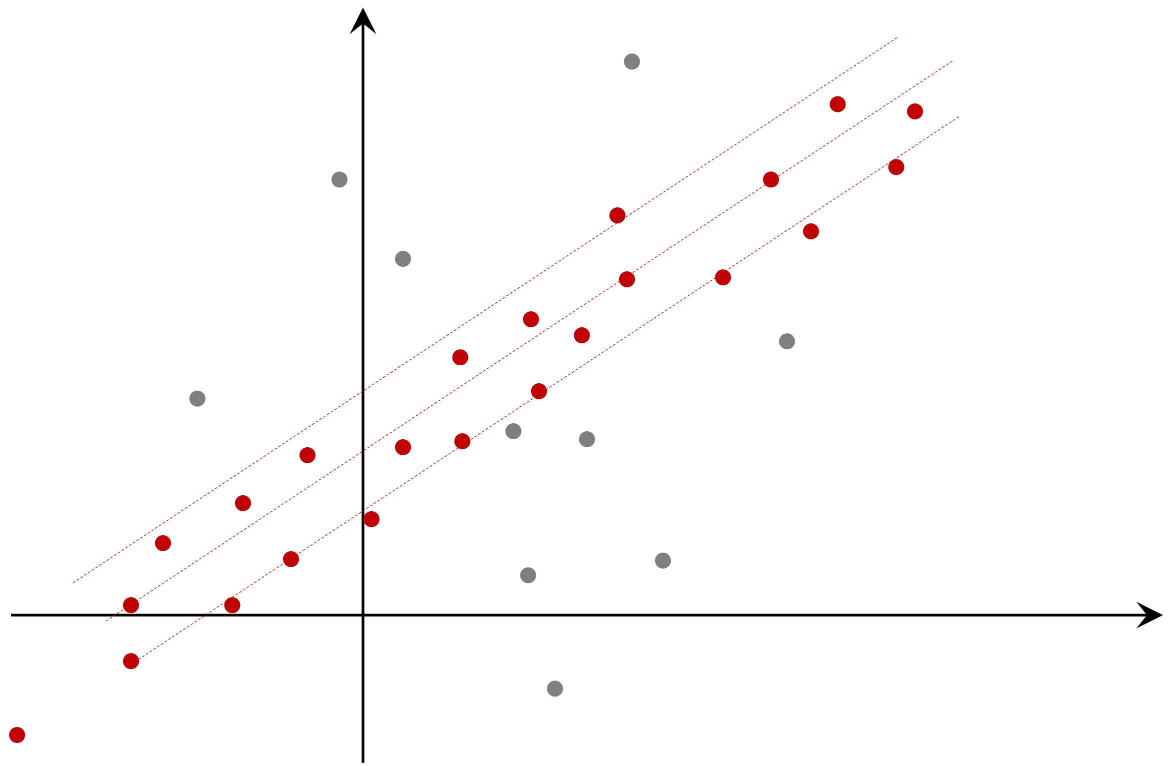
Required number of iterations with p success rate:



Required number of iterations with p success rate:

Probability of choosing an inlier:

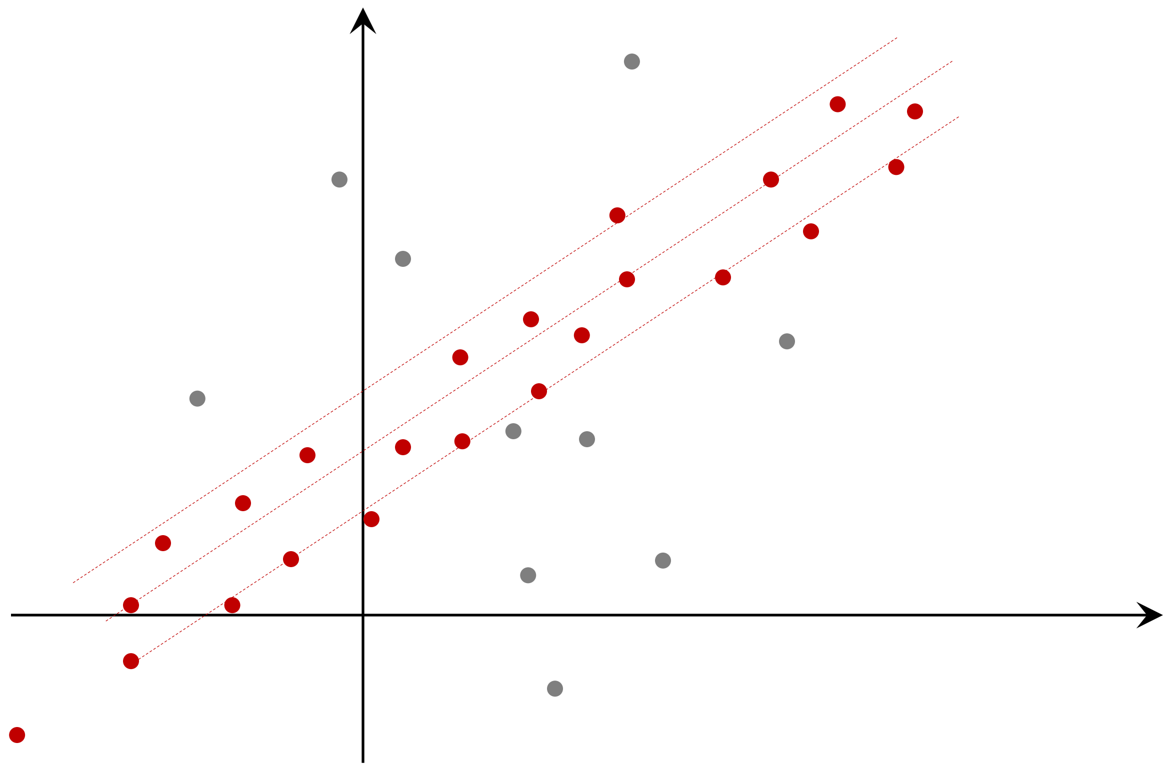
$$w = \frac{\#\text{ of inliers}}{\#\text{ of samples}}$$



Required number of iterations with p success rate:

Probability of choosing an inlier: $w = \frac{\#\text{ of inliers}}{\#\text{ of samples}}$

Probability of building a correct model: w^n where n is the number of samples to build a model.

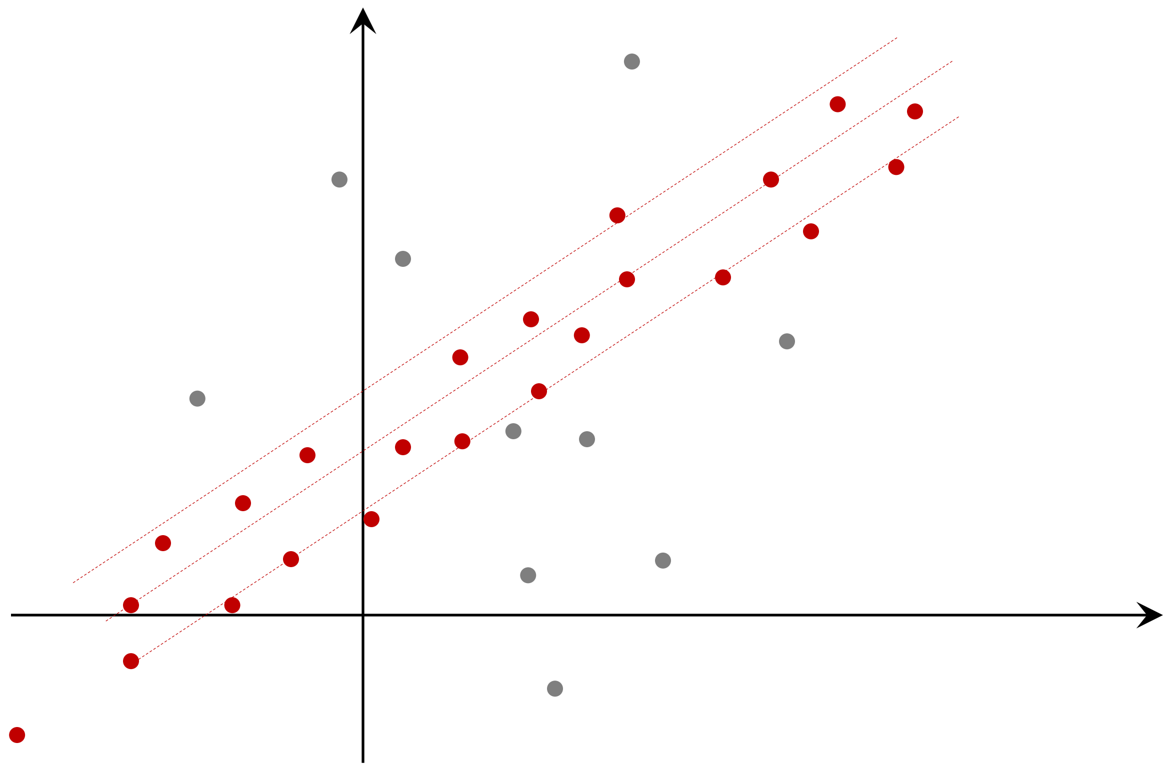


Required number of iterations with p success rate:

Probability of choosing an inlier: $w = \frac{\#\text{ of inliers}}{\#\text{ of samples}}$

Probability of building a correct model: w^n where n is the number of samples to build a model.

Probability of not building a correct model during k iterations: $\left(1 - w^n\right)^k$



Required number of iterations with p success rate:

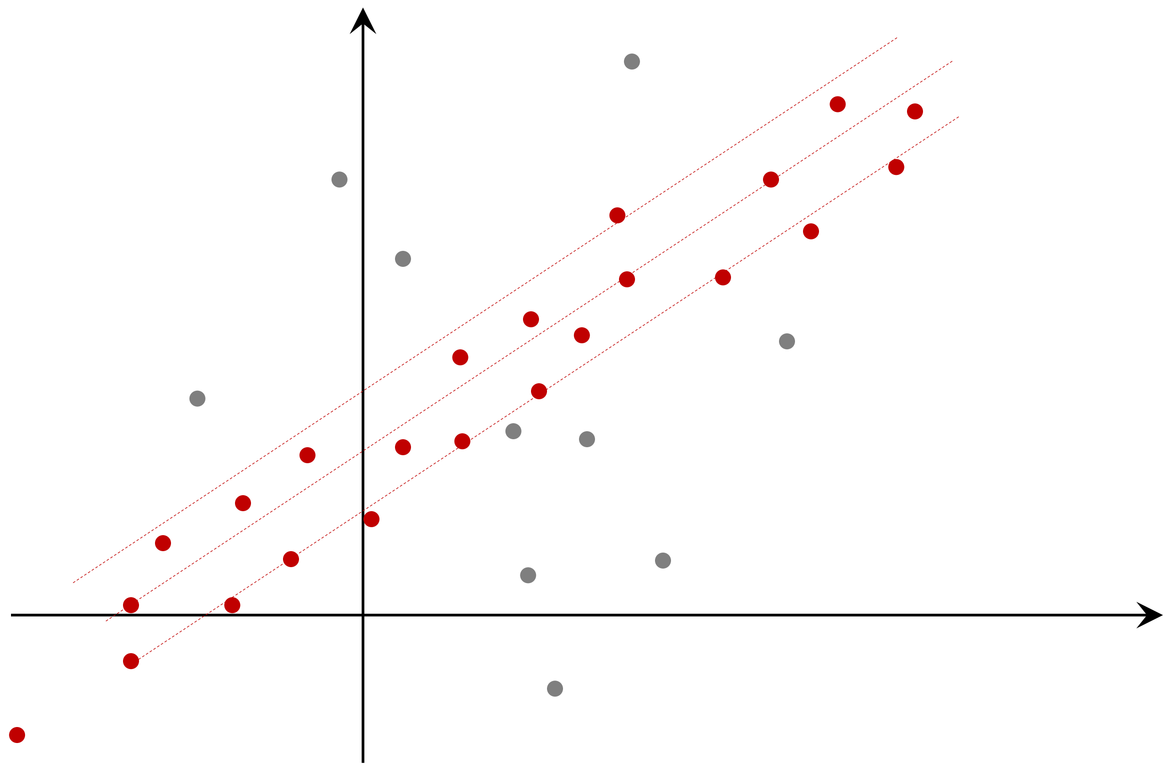
Probability of choosing an inlier: $w = \frac{\#\text{ of inliers}}{\#\text{ of samples}}$

Probability of building a correct model: w^n where n is the number of samples to build a model.

Probability of not building a correct model during k iterations: $(1-w^n)^k$

$(1-w^n)^k = 1-p$ where p is desired RANSAC success rate.

$$k = \frac{\log(1-p)}{\log(1-w^n)}$$



Required number of iterations with p success rate:

$$k = \frac{\log(1-p)}{\log(1-w^n)} \qquad \text{where} \quad w = \frac{\#\text{ of inliers}}{\#\text{ of samples}}$$

Probability of choosing an inlier: $w = \dfrac{\#\text{ of inliers}}{\#\text{ of samples}}$

Probability of building a correct model: w^n where n is the number of samples to build a model.

Probability of not building a correct model during k iterations: $(1-w^n)^k$

$(1-w^n)^k = 1-p$ where p is desired RANSAC success rate.

$$k = \frac{\log(1-p)}{\log(1-w^n)}$$

$\mathbf{I}_1$ $\mathbf{I}_2$

$$\left\{\begin{matrix} \mathbf{v}_1 \leftrightarrow \mathbf{u}_1 \\ \mathbf{v}_2 \leftrightarrow \mathbf{u}_2 \\ \mathbf{v}_3 \leftrightarrow \mathbf{u}_3 \\ \mathbf{v}_4 \leftrightarrow \mathbf{u}_4 \end{matrix}\right\} \rightarrow \mathbf{H}$$

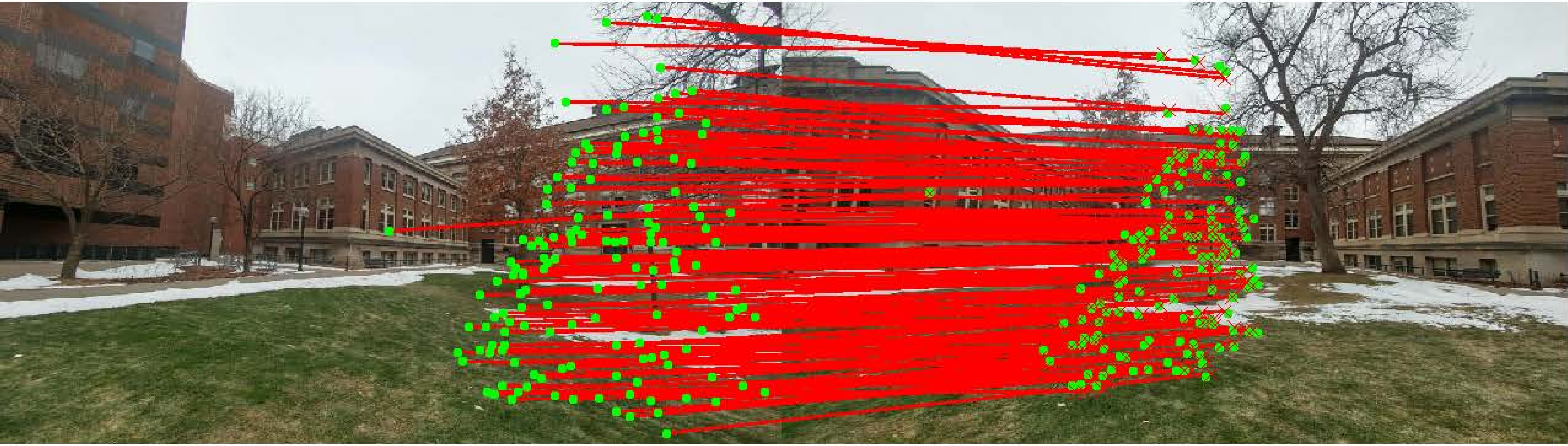
Homography computation

$\mathbf{I}_1$ $\mathbf{I}_2$

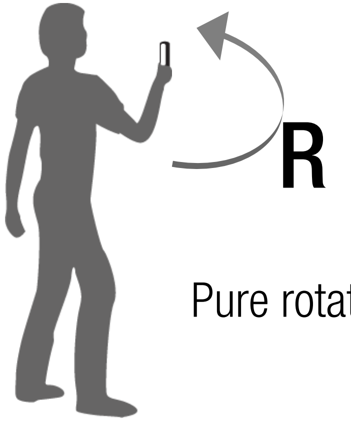
$$\left\{\begin{matrix} \mathbf{v}_1 \leftrightarrow \mathbf{u}_1 \\ \mathbf{v}_2 \leftrightarrow \mathbf{u}_2 \\ \mathbf{v}_3 \leftrightarrow \mathbf{u}_3 \\ \mathbf{v}_4 \leftrightarrow \mathbf{u}_4 \end{matrix}\right\} \rightarrow \mathbf{H} \longrightarrow \qquad \longrightarrow \quad \mathbf{v} = \mathbf{H}\mathbf{u}$$

Homography computation

Inlier evaluation



I_1 $\qquad$ I_2



Pure rotation

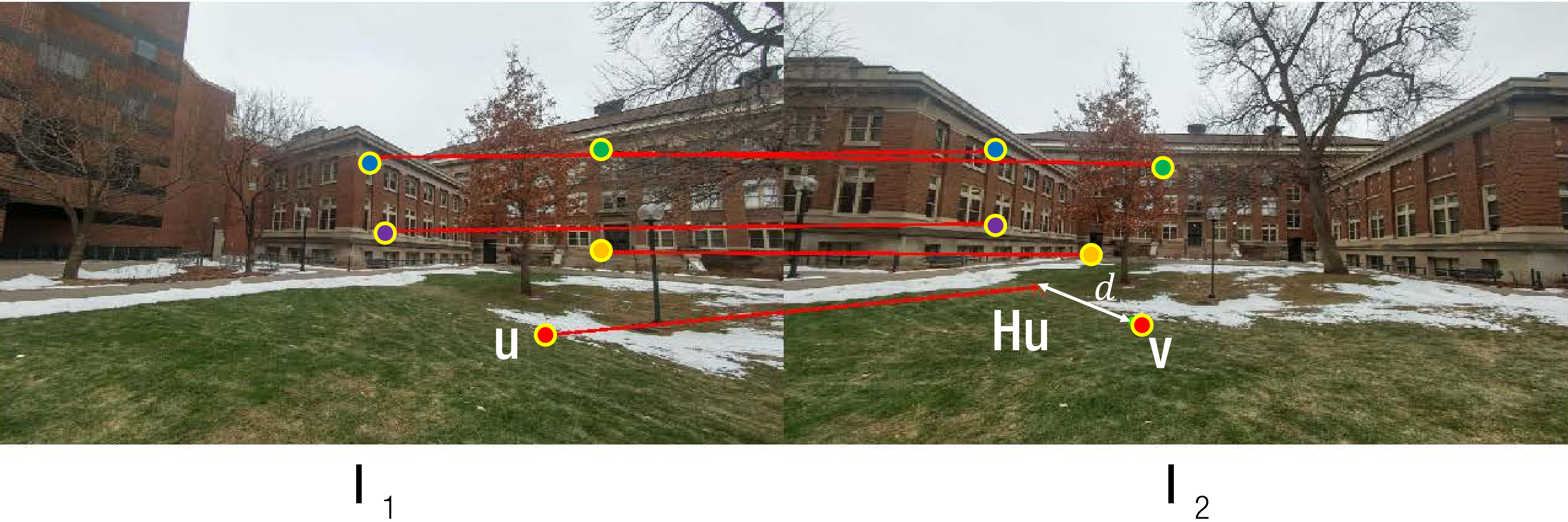
$$I_2(\mathbf{v}) = I_1(\mathbf{Hu})$$

where

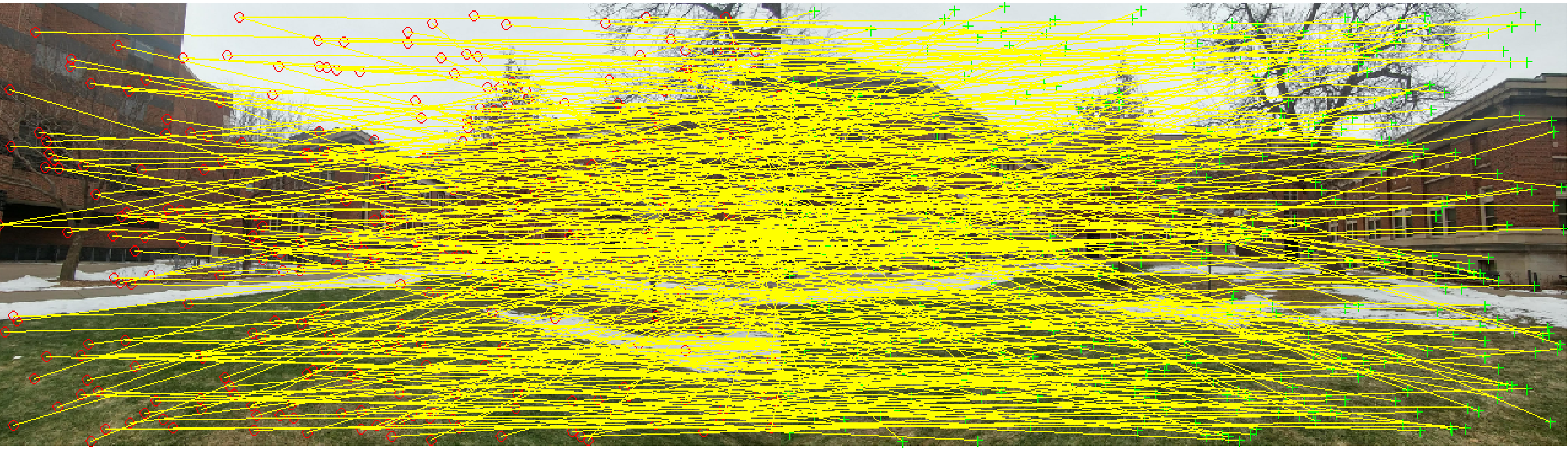
$$\mathbf{v} = \mathbf{Hu}$$

I_1 I_2

If the correspondence is bad, the computed homography will fit the four points still perfectly, but how do we know it is wrong?

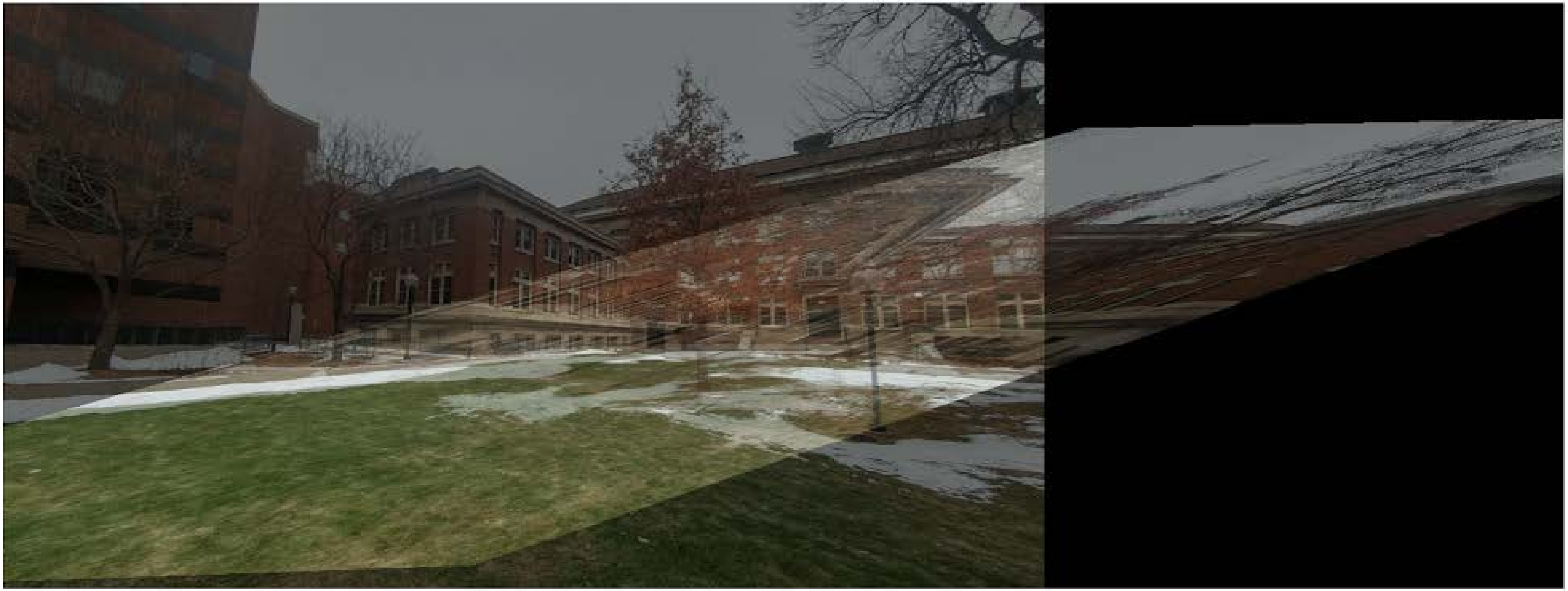


If the correspondence is bad, it has no prediction power!

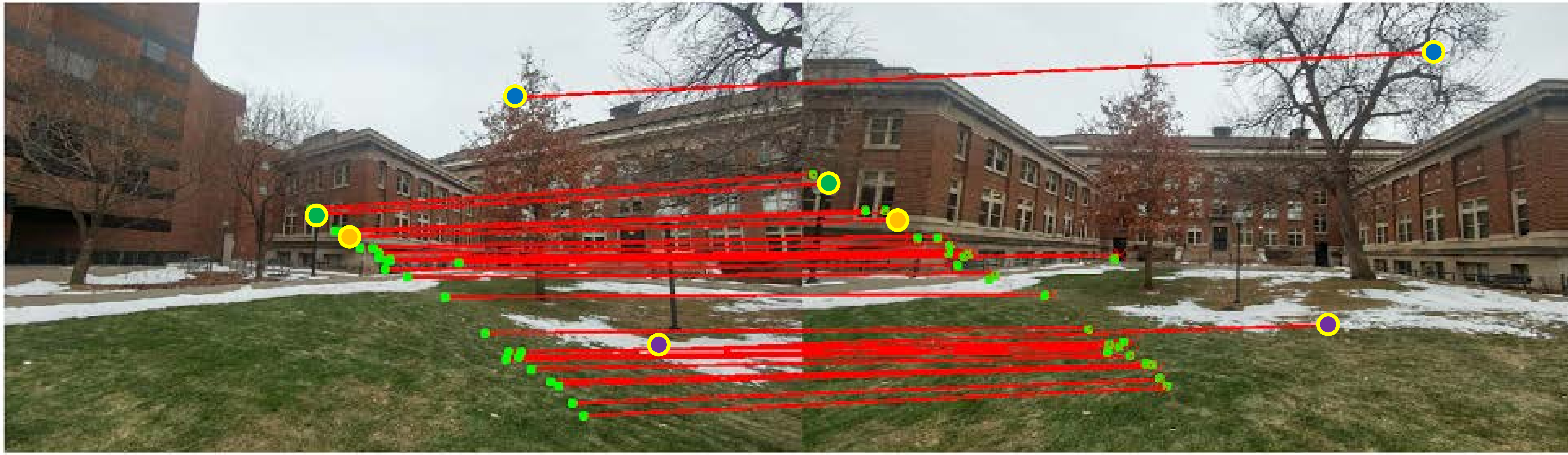



Hu
d
V

# of inliers: 16 out of 1865

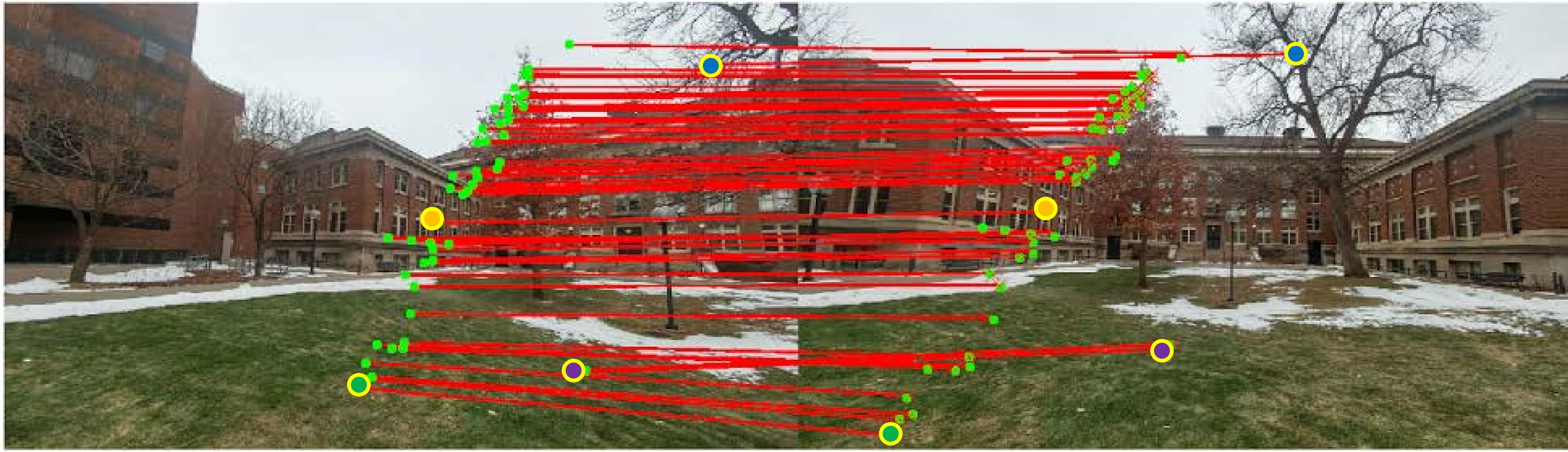


# of inliers: 16 out of 1865



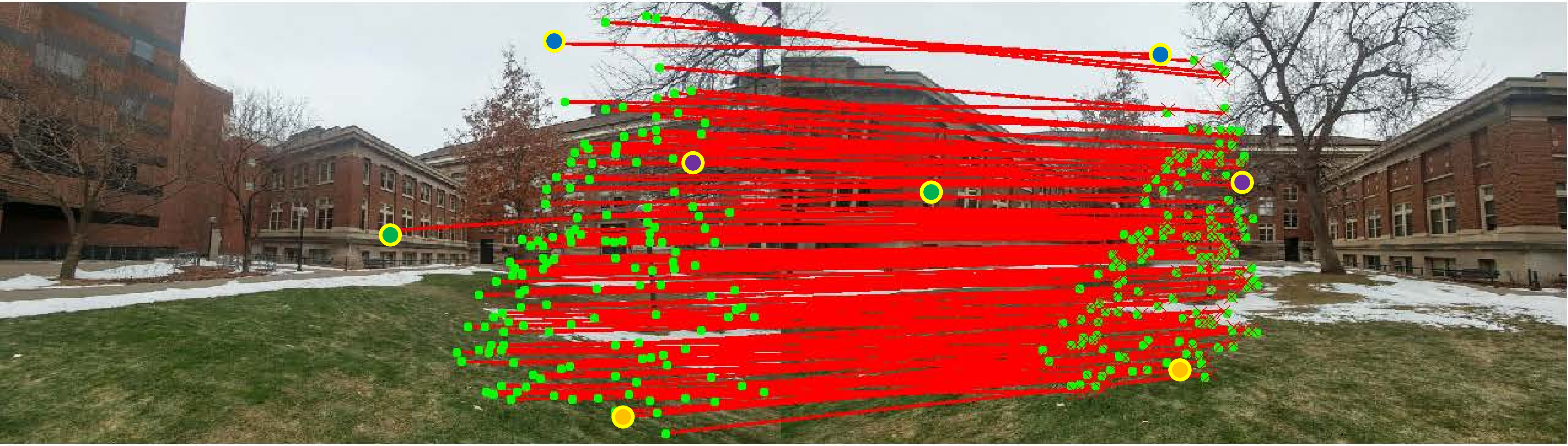
# of inliers: 36 out of 1865

# of inliers: 36 out of 1865



# of inliers: 57 out of 1865

# of inliers: 57 out of 1865



# of inliers: 216 out of 1865

# of inliers: 216 out of 1865

Euclidean Transform (Translation)

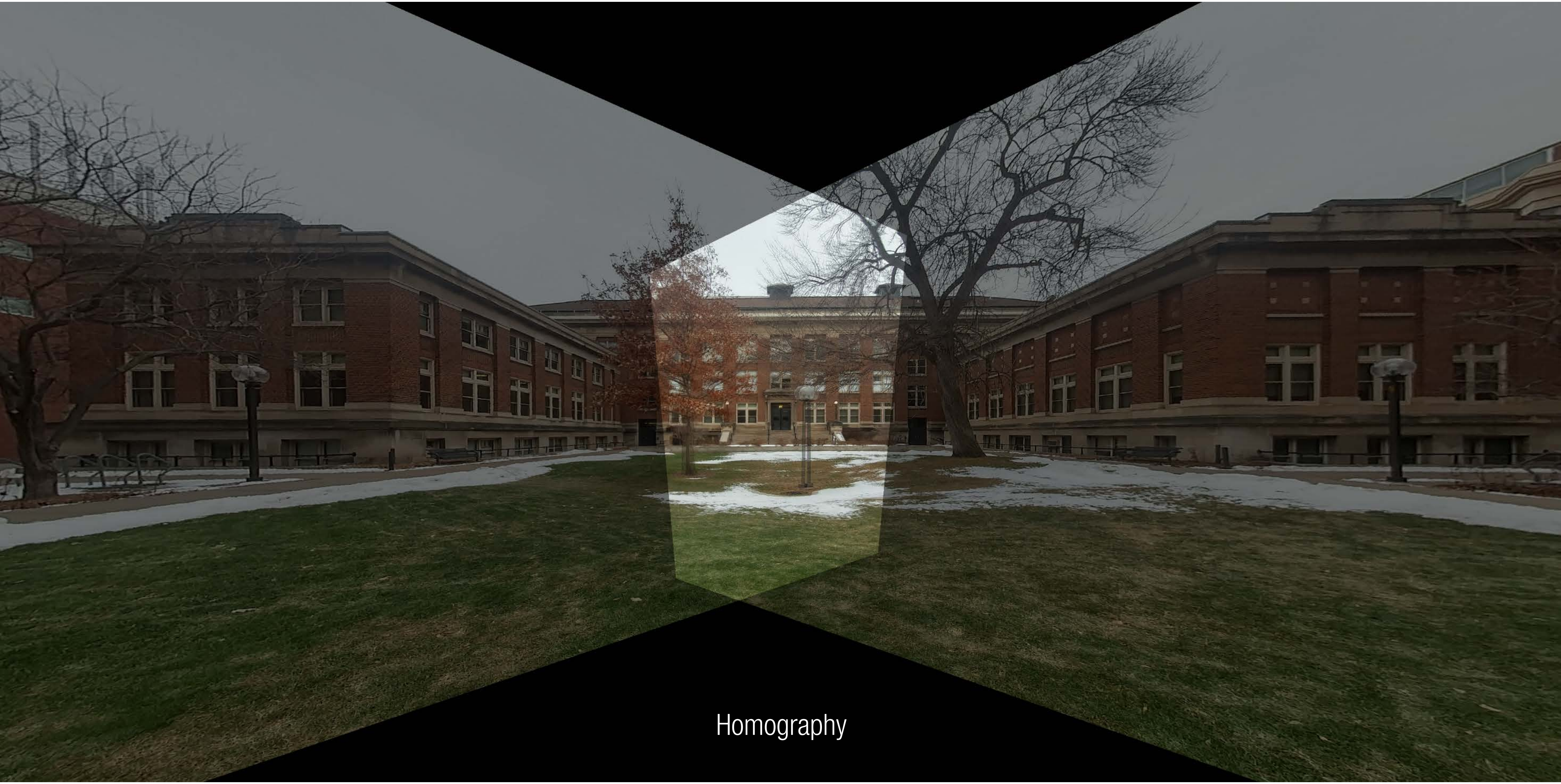
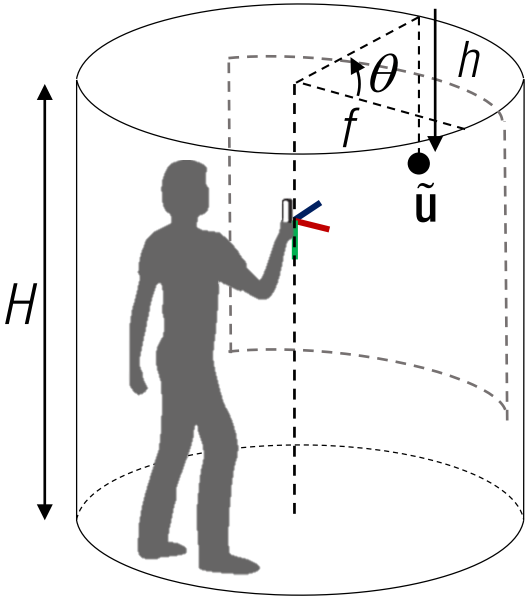

Homography

Image Panorama (Cylindrical Projection)



First camera:

Point on cylindrical surface: $[h, \theta]$

$\longleftrightarrow$ Point in 3D space: $[f\cos(\theta), h, f\sin(\theta)]$

$\longleftrightarrow$ Point in image coordinate: $\mathrm{K}[f\cos(\theta), h, f\sin(\theta)]^{\mathrm{T}}$

Image Panorama (Cylindrical Projection)

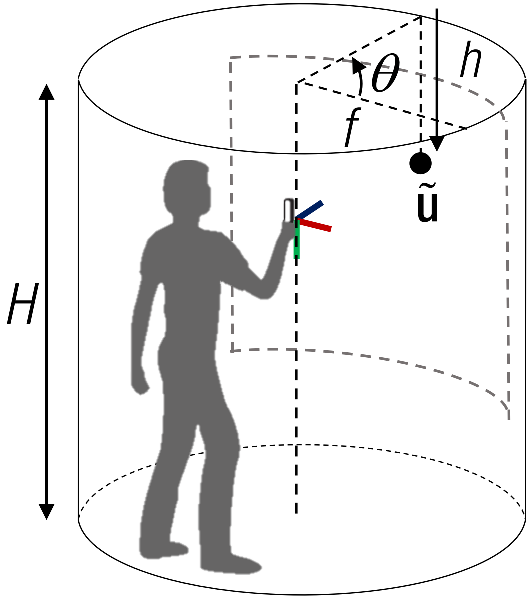
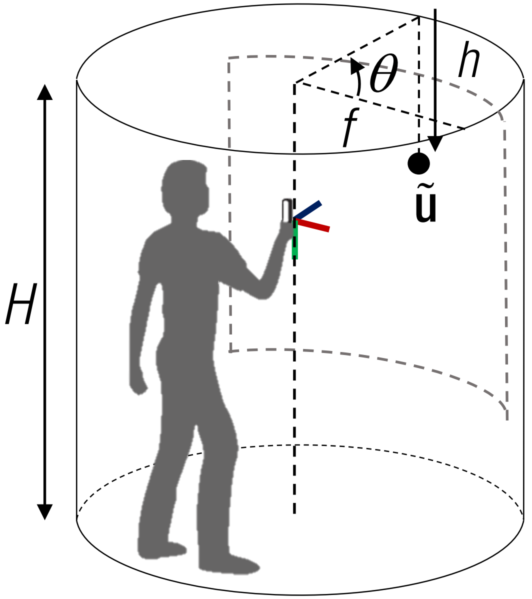


Image Panorama (Cylindrical Projection)



Second camera:

Point on cylindrical surface: $[h, \theta]$

$\longleftrightarrow$ Point in 3D space: $[f\cos(\theta), h, f\sin(\theta)]$

$\longleftrightarrow$ Point in image coordinate: $\mathrm{K}\mathrm{R}[f\cos(\theta), h, f\sin(\theta)]^{\mathrm{T}}$

where R is given by $\mathrm{R} = \mathrm{K}^{-1}\mathrm{H}\mathrm{K}$

Image Panorama (Cylindrical Projection)

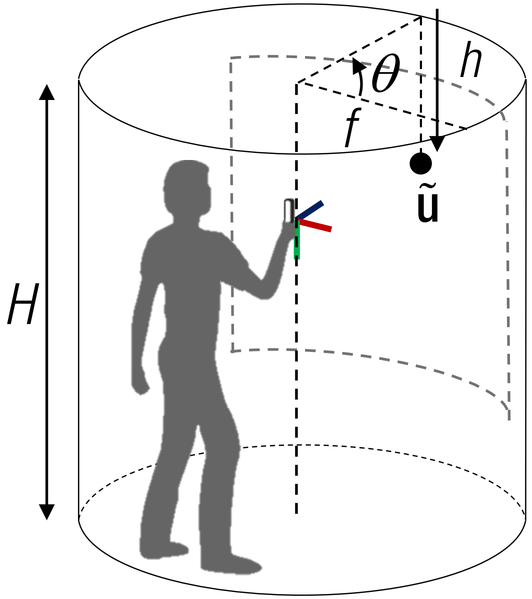


Image Panorama (Cylindrical Projection)