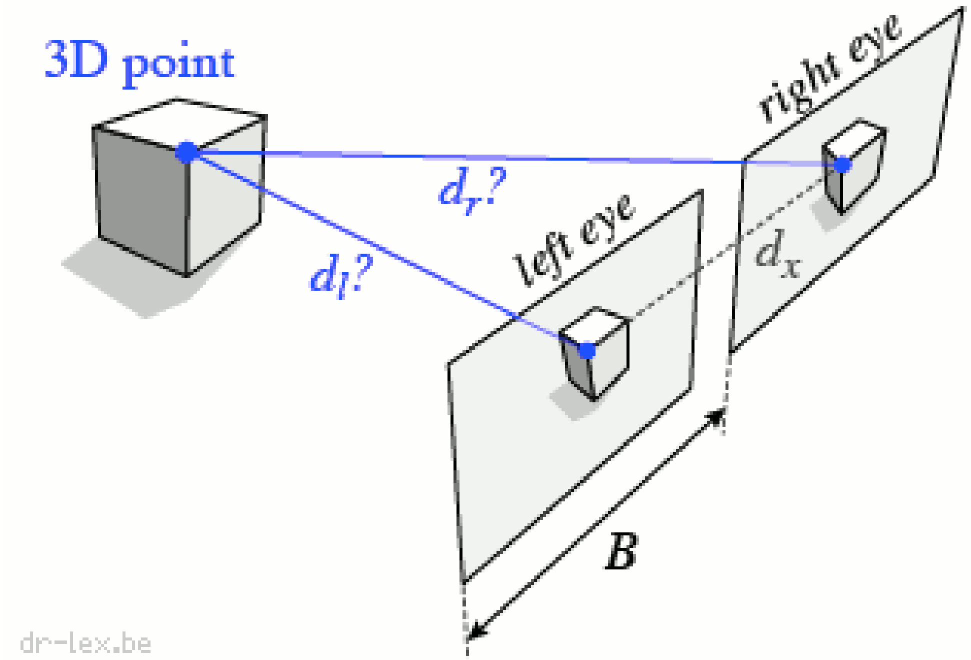
Computer Vision Project

Daniel Paredes

Outline

- Introduction
- Feature Detectors and Descriptors
    - Harris + FREAK
    - SURF
- RANSAC
- Pose Estimation
    - Fundamental Matrix
    - Essential Matrix
    - Triangulation
- Results / Conclusions

Introduction



Introduction

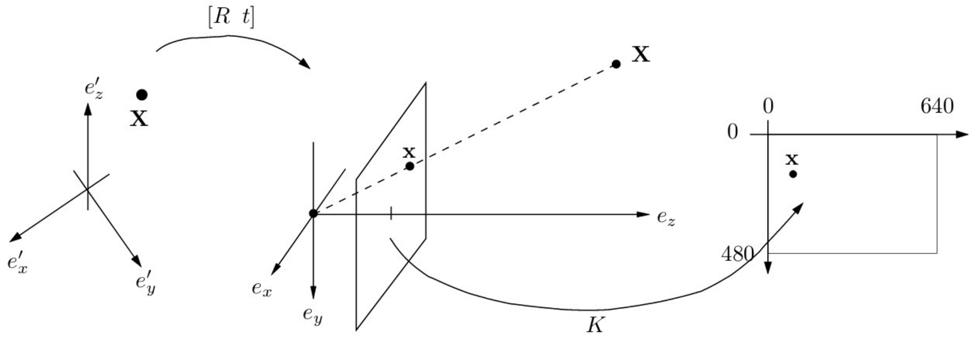
Feature Points

Introduction

Feature Points → Match Points

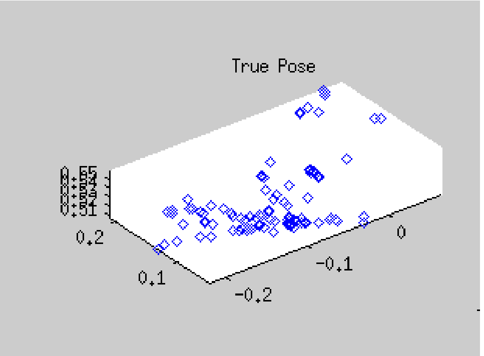
Introduction

Feature Points → **Match Points** → **Find Transformations**

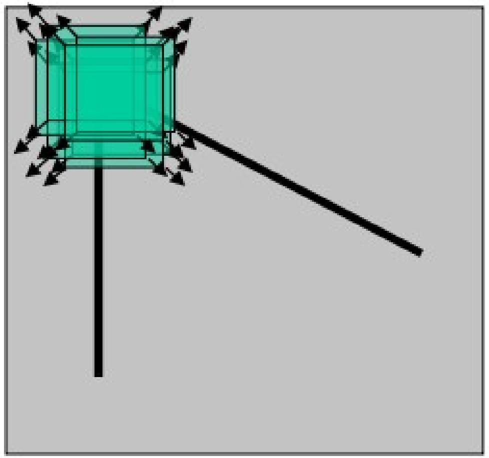


Introduction

Feature Points → Match Points → Find Transformations → 3D Triangulation

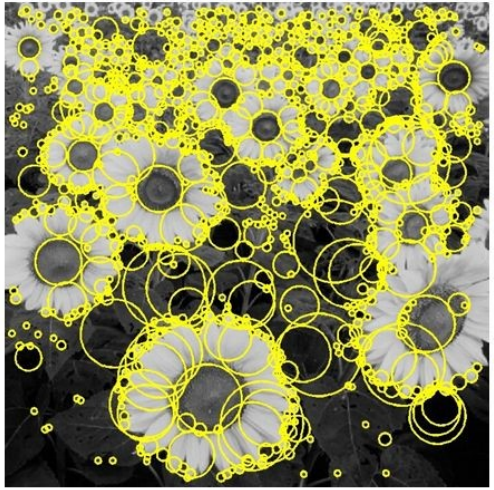


Feature Detectors



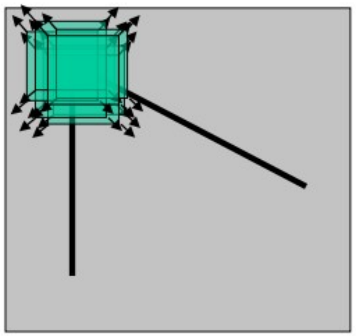
[http://hristoalexiev.blogspot.com]

Harris
Corner Detector

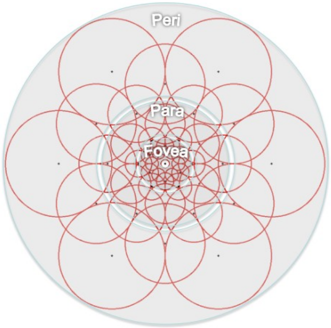


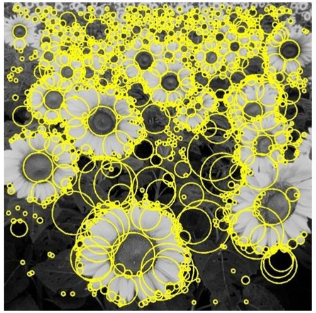
SURF detector
Blob detector

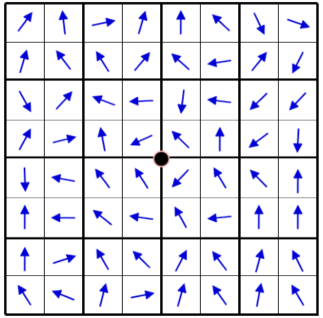
Feature Detectors + Feature Descriptors



+







[http://hristoalexiev.blogspot.com]

[FREAK-Alahi.2012]

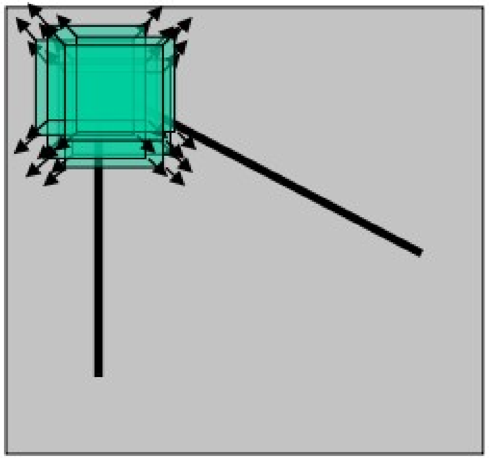
Harris Corner Detector

FREAK: Feature Descriptor

SURF: Feature Detector and Descriptor

Harris Corner Detector + FREAK

Feature Detectors: Harris Detector

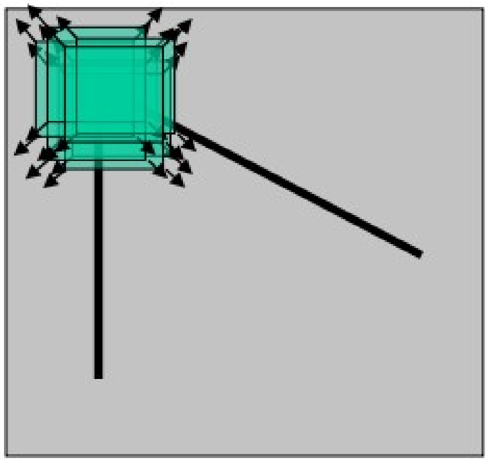


[http://hristoalexiev.blogspot.com]

Detects points, in which the gradient changes in two orthogonal directions

Harris
Corner Detector

Feature Detectors: Harris Detector



[http://hristoalexiev.blogspot.com]

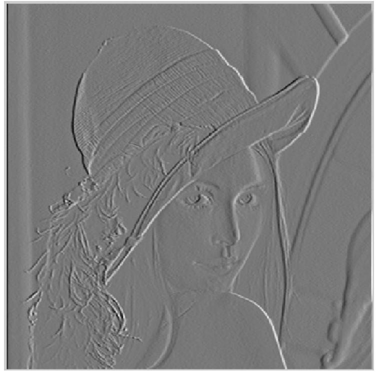
Harris
Corner Detector

Five steps:

1. Color to grayscale
2. Spatial derivative calculation
3. Structure tensor setup
4. Harris response calculation
5. Non-maximum suppression

Feature Detectors: Harris Detector

2. Image gradients:



$$I_x = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} * I$$

$$I_y = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} * I$$

Feature Detectors: Harris Detector

3. Structure tensor setup

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

Feature Detectors: Harris Detector

4. Harris response calculation

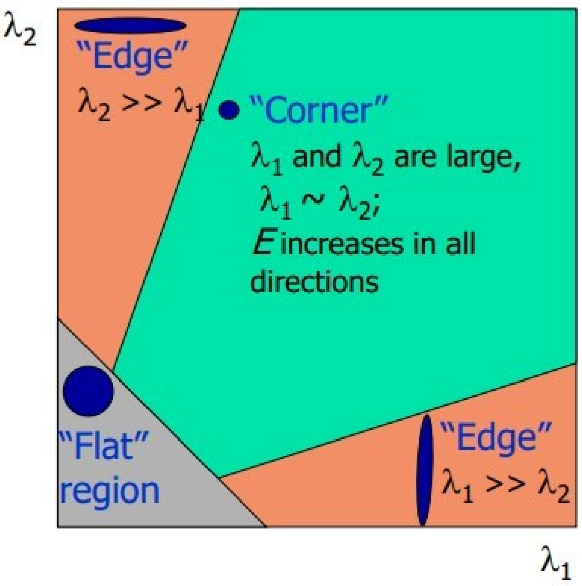
$$R = det(M) - k(trace(M))^2 = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2$$

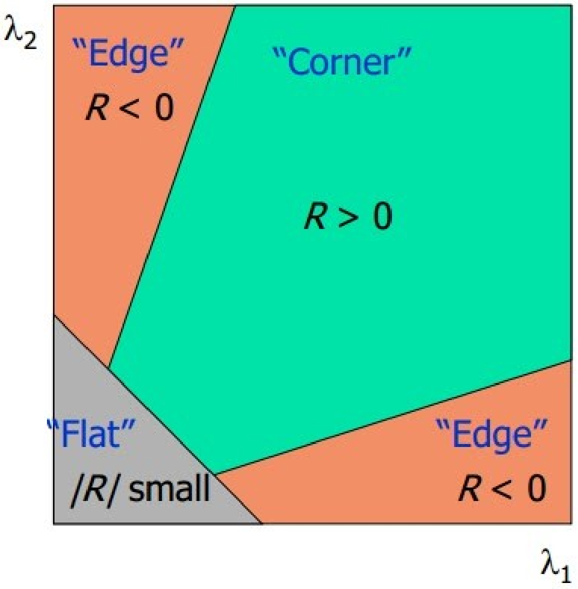
$$k \in [0.04, 0.06]$$

λ_1 , λ_2 are the eigenvalues of M

Feature Detectors: Harris Detector

4. Harris response calculation



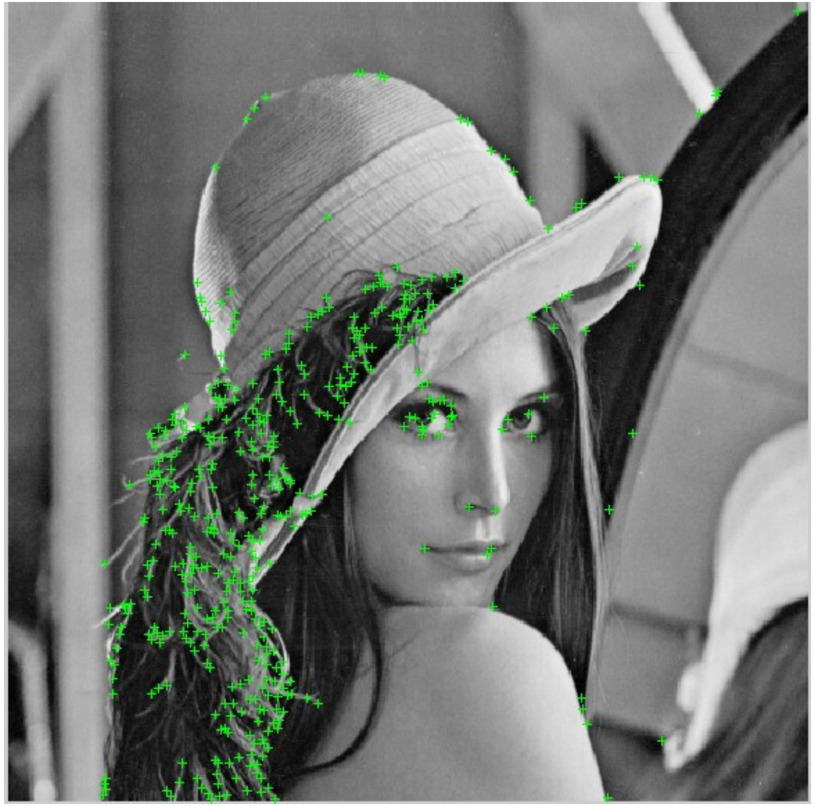


Feature Detectors: Harris Detector

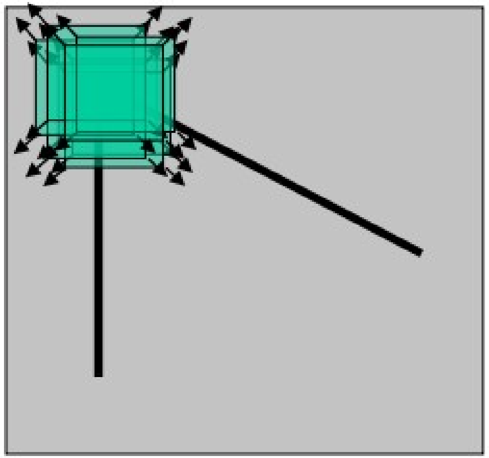
4. Non-maximum suppression

- Filter to find the maximum values within a window of size W
- Several points with R>0 can be very close

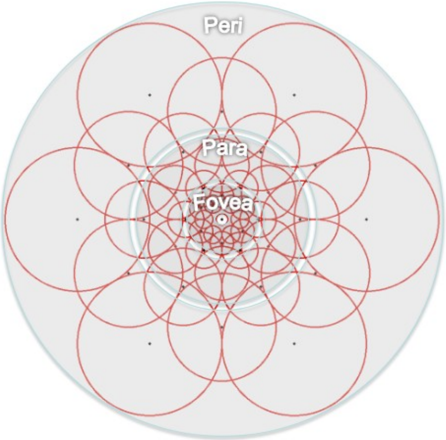
Feature Detectors: Harris Detector



Feature Descriptor: FREAK



+



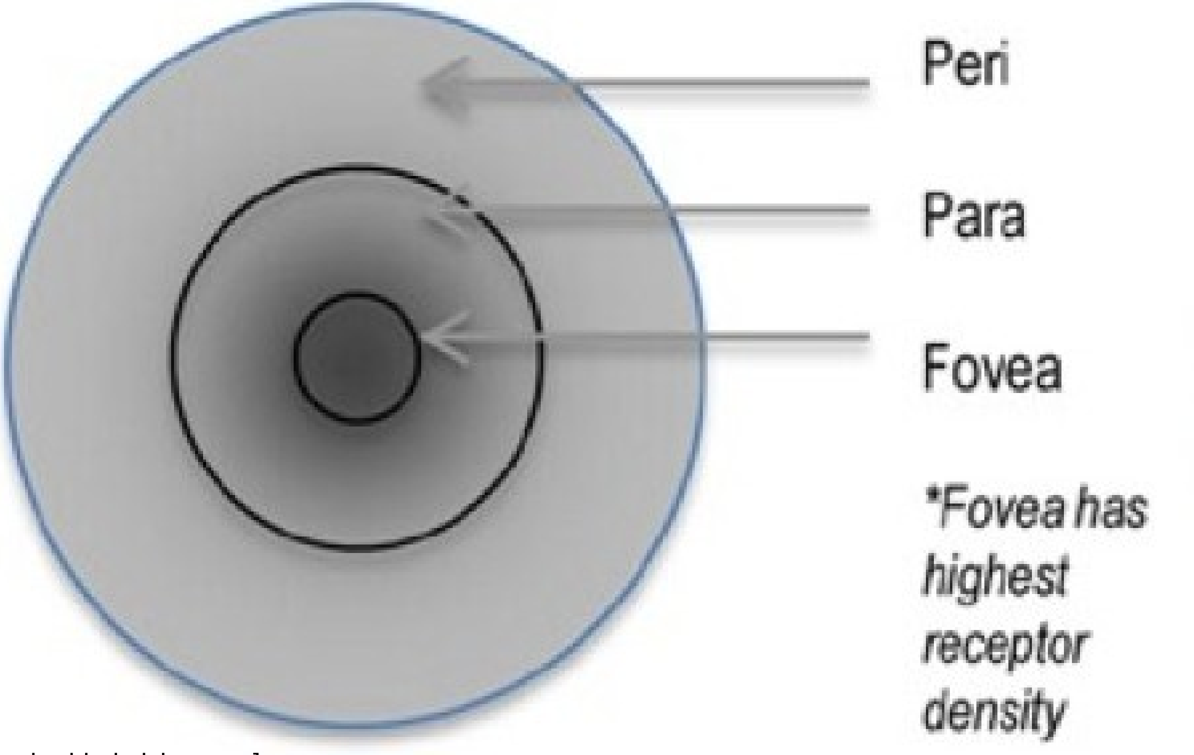
[http://hristoalexiev.blogspot.com]

[FREAK-Alahi.2012]

Harris
Corner Detector

FREAK:
Feature Descriptor

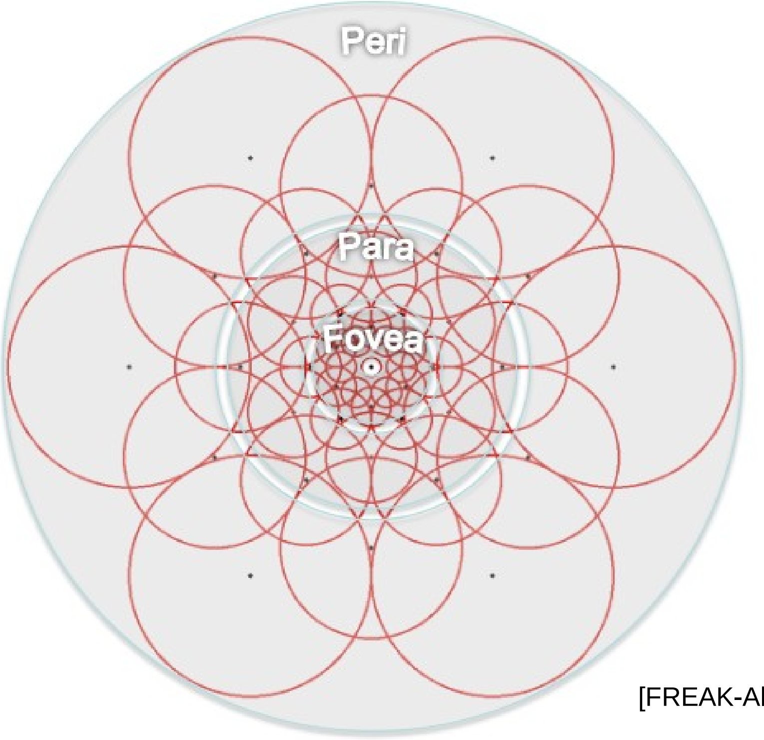
Feature Descriptor: FREAK



[www.embedded-vision.com]

FREAK:
Fast REtinA Keypoint

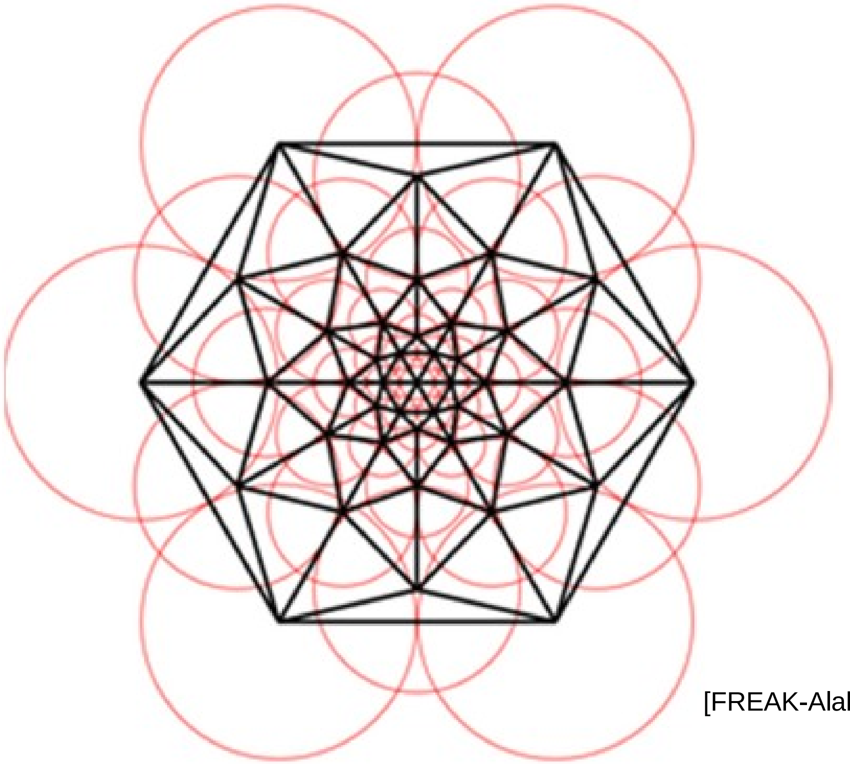
Feature Descriptor: FREAK



[FREAK-Alahi.2012]

Sampling Pattern

Feature Descriptor: FREAK

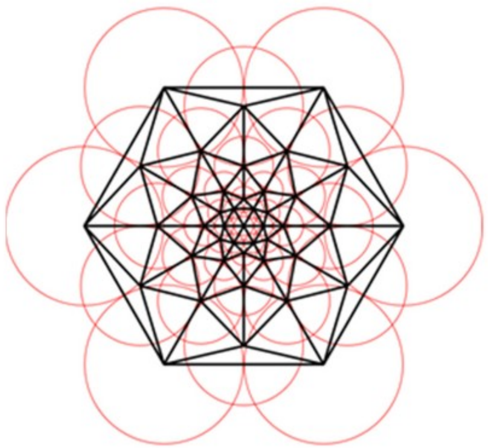


[FREAK-Alahi.2012]

Paired Points:

- Discard correlated Points
- Maximizing variance of pairs

Feature Descriptor: FREAK



[FREAK-Alahi.2012]

$$F = \sum_{0 \leq a < N} 2^a T(P_a)$$

$$T(P_a) = \begin{cases} 1 & \text{if } (I(P_a^{r_1}) - I(P_a^{r_2}) > 0, \\ 0 & \text{otherwise,} \end{cases}$$

Descriptor of size N, with P_a pairs

SURF

Feature Detectors: SURF detector

x = (x,y) is a point in an image **I**.

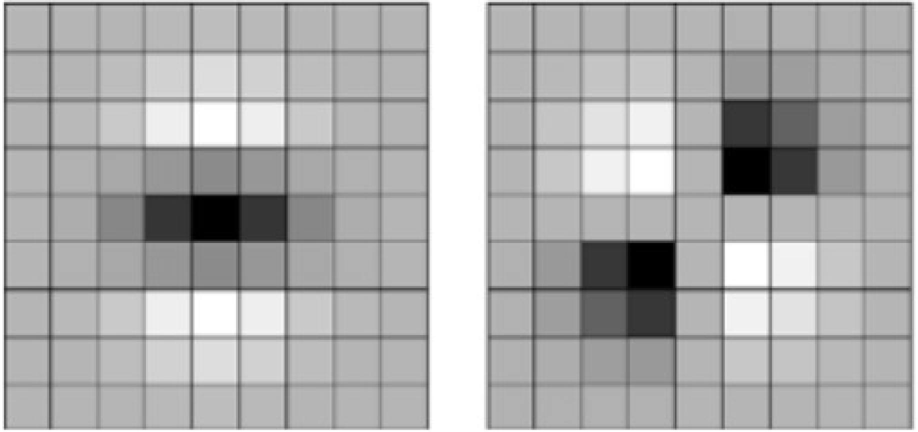
The Hessian is given by:

$$\mathcal{H}(\mathbf{x},\sigma) = \begin{bmatrix} L_{xx}(\sigma) & L_{xy}(\sigma) \\ L_{yx}(\sigma) & L_{yy}(\sigma) \end{bmatrix} * \mathcal{I}$$

Feature Detectors: SURF detector

[SURF-Bay.2008]

$$\begin{bmatrix} L_{xx}(\sigma) & L_{xy}(\sigma) \\ L_{yx}(\sigma) & L_{yy}(\sigma) \end{bmatrix}$$



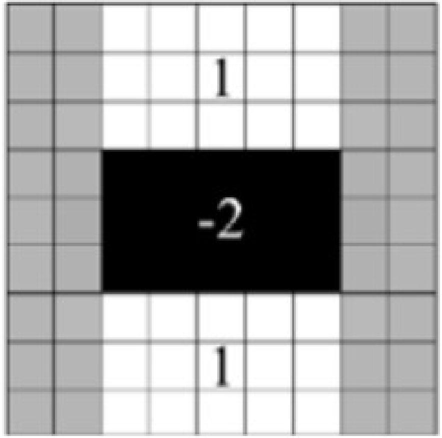
a) L_{yy} a) L_{xy}

Feature Detectors: SURF detector

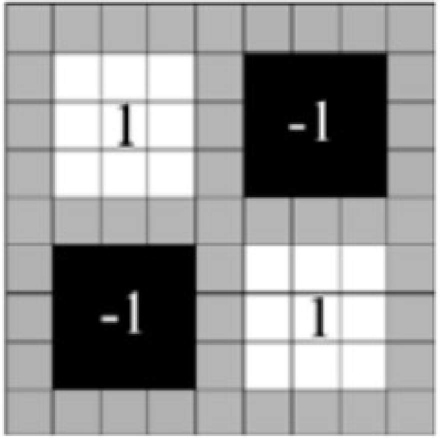
[SURF-Bay.2008]

$$\begin{bmatrix} L_{xx}(\sigma) & L_{xy}(\sigma) \\ L_{yx}(\sigma) & L_{yy}(\sigma) \end{bmatrix}$$

Approximation!



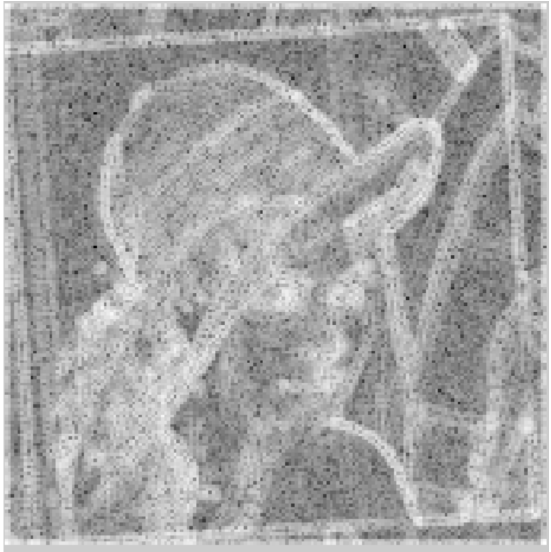
a) D_{yy}



a) D_{xy}

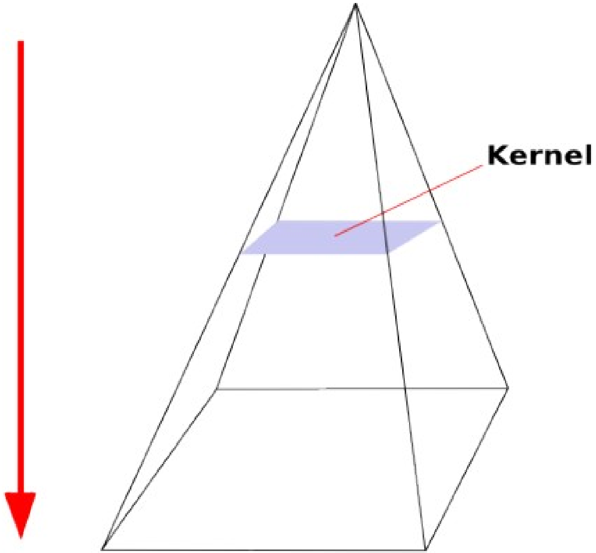
Feature Detectors: SURF detector

Candidates: $\det(H) = D_{xx}D_{yy} - (0.9D_{xy})^2$



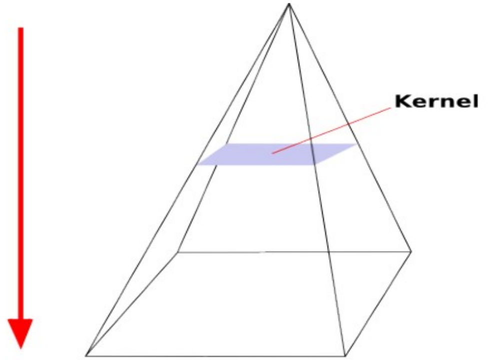
Feature Detectors: SURF detector

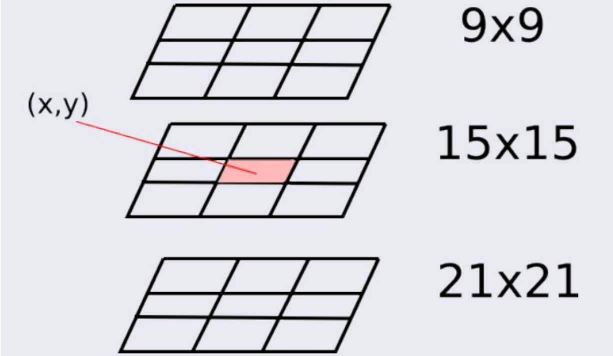
Scaling



Feature Detectors: SURF detector

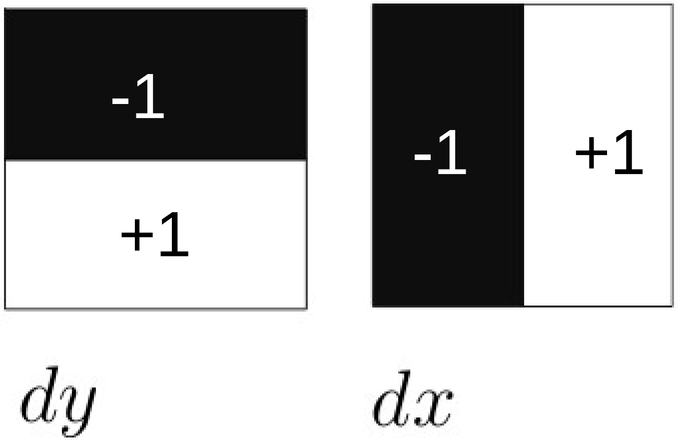
Scaling





3D Non-maximum suppression

Feature Descriptor: SURF

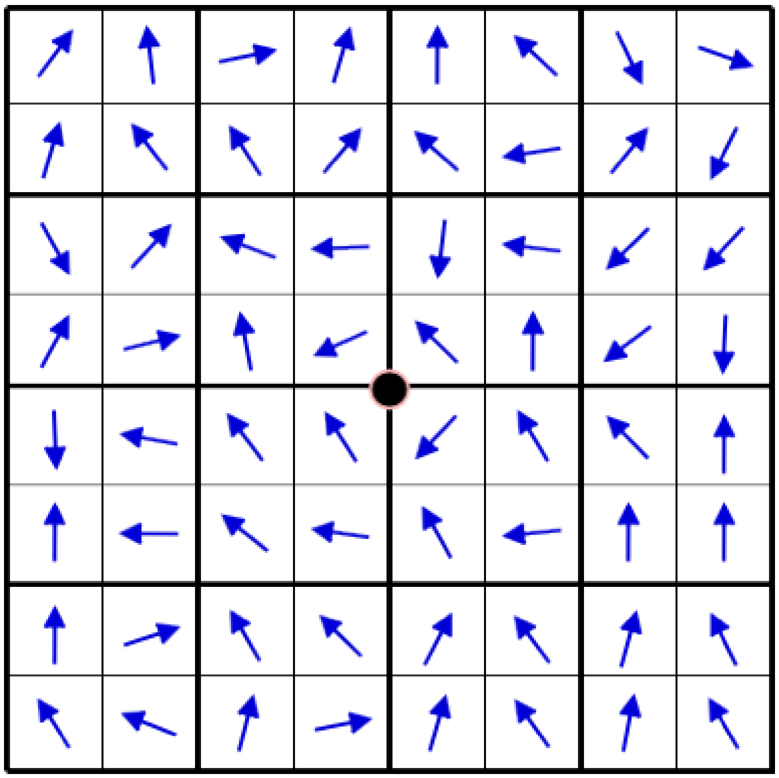


Haar Wavelet Filters

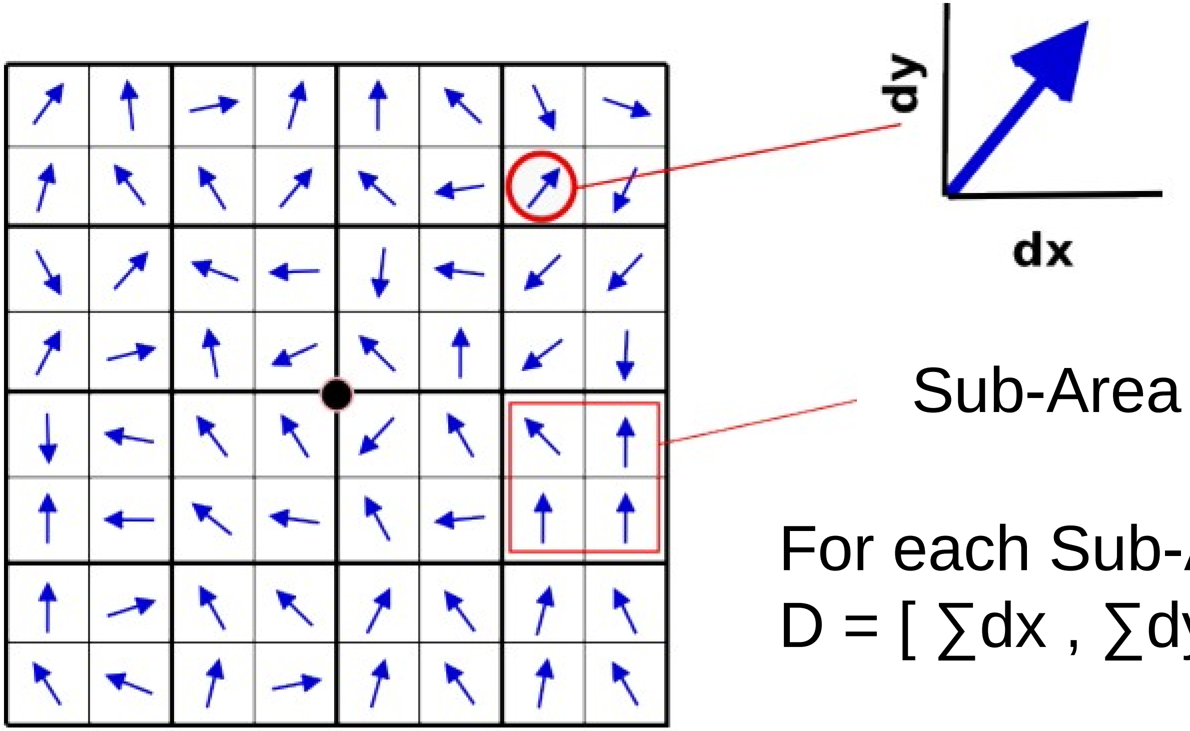
Feature Descriptor: SURF

For each detected feature

- 1 Interested Area
- Divided 4x4 Sub-areas



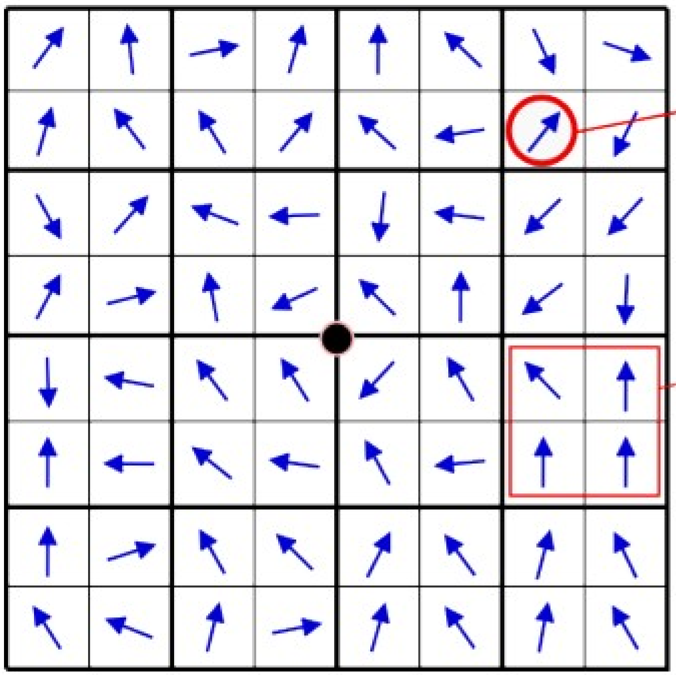
Feature Descriptor: SURF

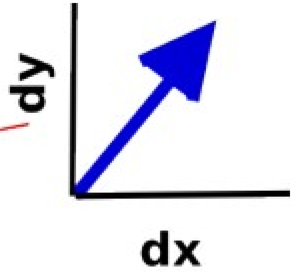


For each Sub-Area:

$D = [\ \sum dx\ ,\ \sum dy,\ \sum|dx|\ ,\ \sum|dy|\ ]$

Feature Descriptor: SURF





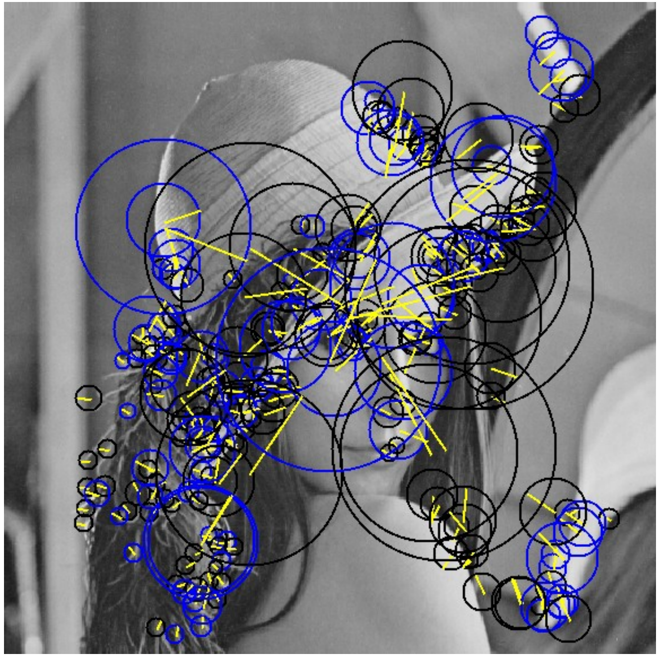
Each point has a descriptor of 64 values

Sub-Area

For each Sub-Area:
D = [∑dx , ∑dy, ∑|dx| , ∑|dy|]

Example: SURF

Feature Detector

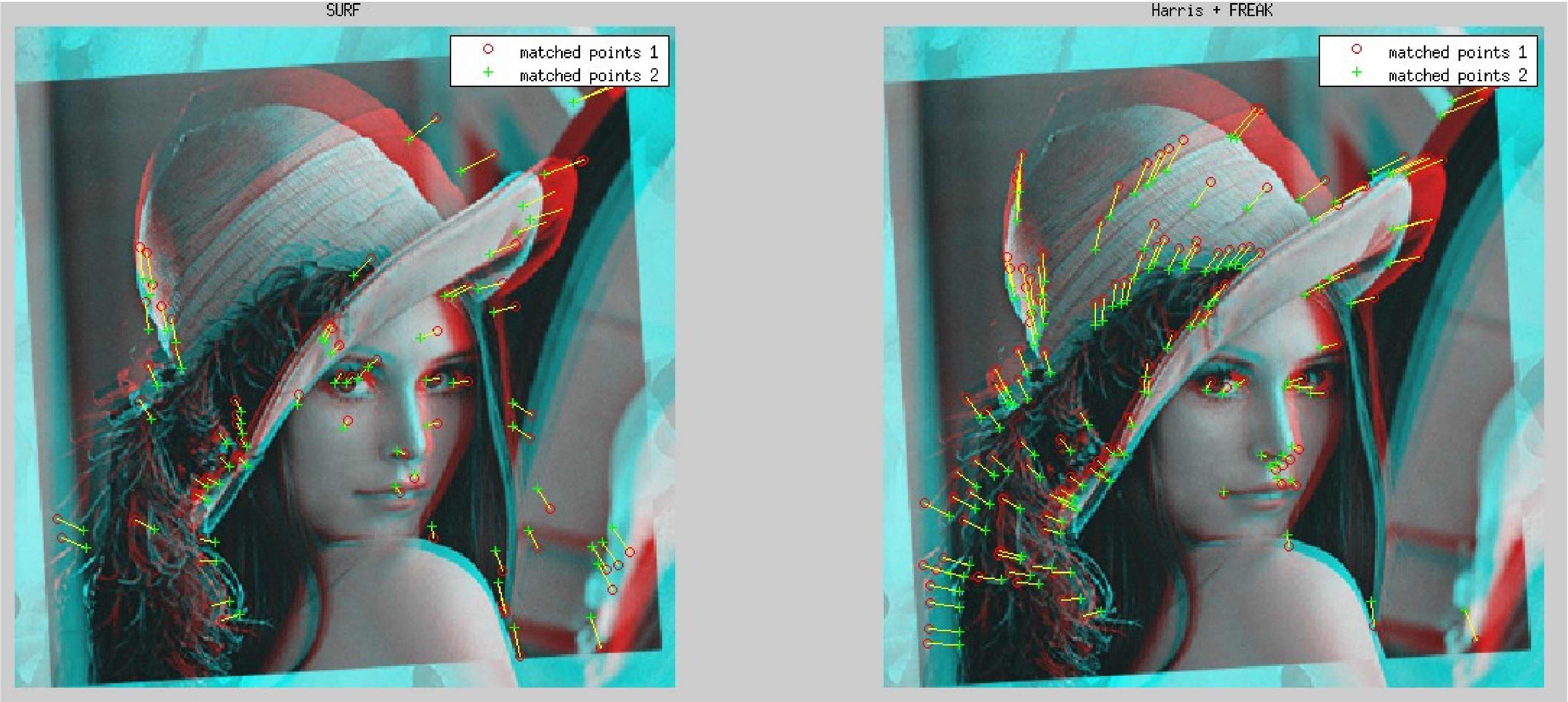


Feature Descriptor

Comparison: Harris-FREAK vs SURF

LENA - Rotation and Scaling

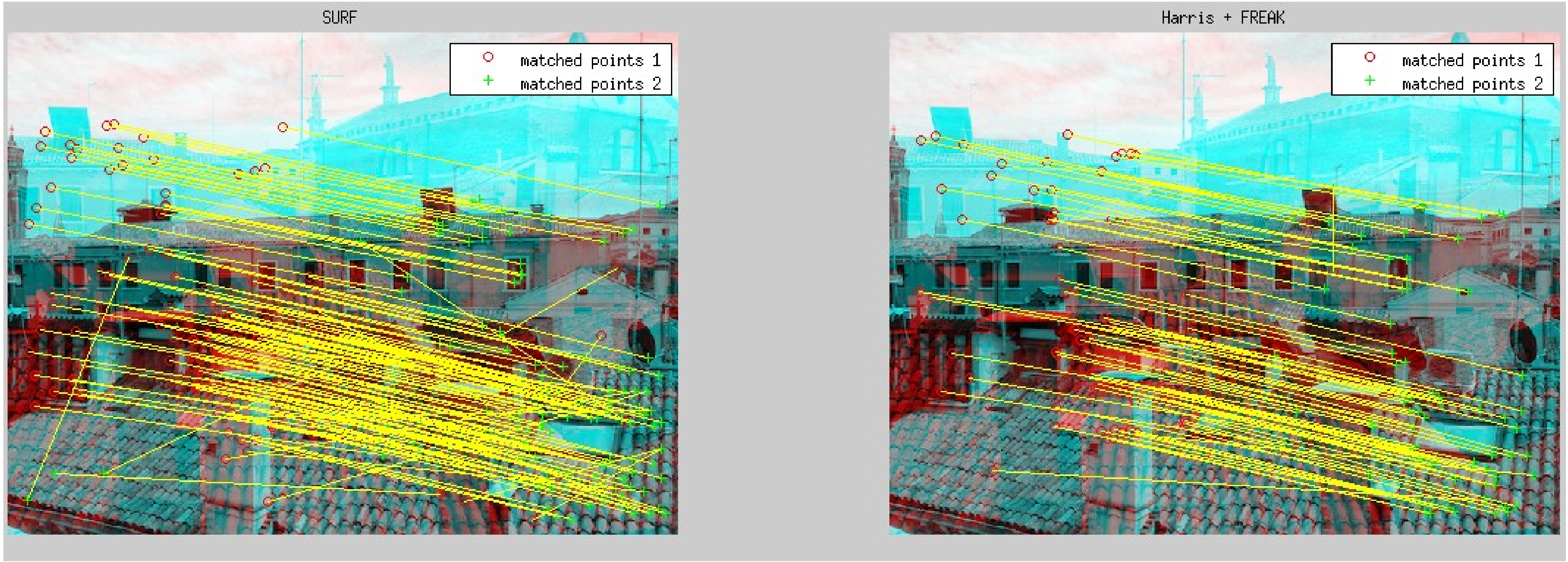
Comparison: Harris-FREAK vs SURF



Comparison: Harris-FREAK vs SURF

ROOFS - Perspective

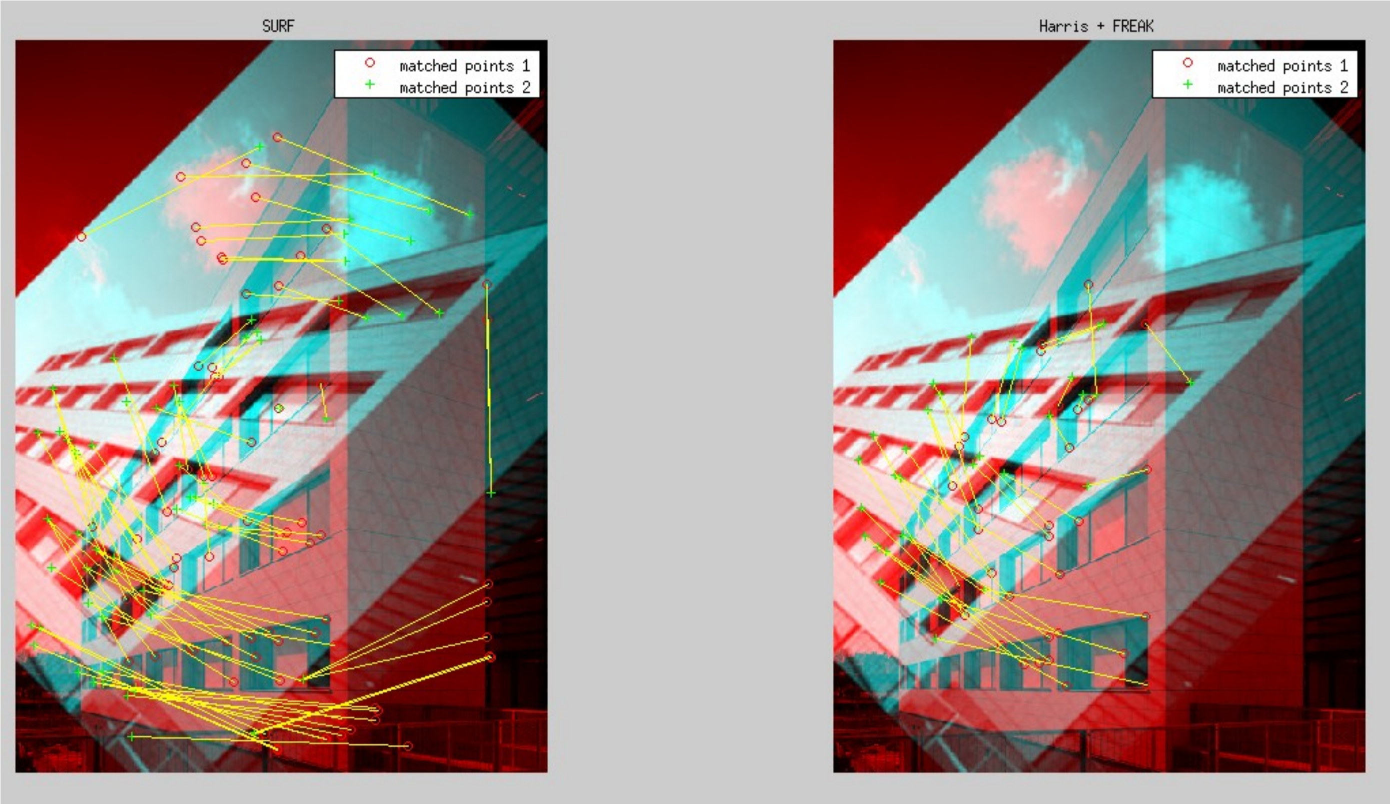
Comparison: Harris-FREAK vs SURF



Comparison: Harris-FREAK vs SURF

BUILDING - Rotation

Comparison: Harris-FREAK vs SURF



Comparison: Harris-FREAK vs SURF

Number of features

	Lena	Roof	Building
SURF	521 306	1557 1061	109 130
Harris + FREAK	362 263	2124 1646	274 274

Comparison: Harris-FREAK vs SURF

Matched Features

	Lena	Roof	Building
SURF	79	135	72
Harris + FREAK	34	77	135

Comparison: Harris-FREAK vs SURF

Time in sec

	Lena	Roof	Building
SURF	0.1206 0.0555	0.1020 0.0767	0.0222 0.0244
Harris + FREAK	0.2472 0.1326	0.2225 0.1873	0.0938 0.0907

Comparison: Harris-FREAK vs SURF

- SURF is faster
- Harris detects more features
- SURF get more matches

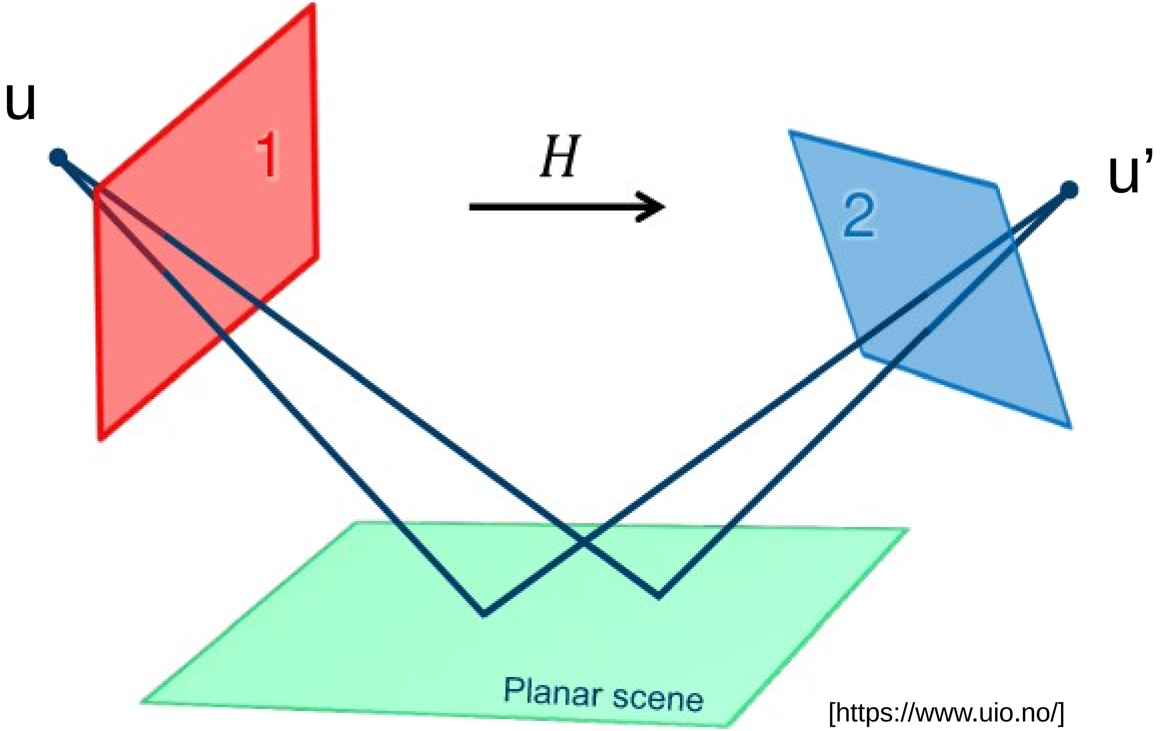
RANSAC

RANSAC

Refinement of matching points

1. Select a random subset of the original data.
2. Model fitting with subset - Least Squares Regression
3. Test Model with full data - Get number of Inliers
4. If the number of inliers < threshold, keep trying

RANSAC - Homography



$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix}$$

Hu = u’

[https://www.uio.no/]

RANSAC - Homography

$$\begin{bmatrix} 0 & 0 & 0 & -u_1 & -v_1 & -1 & v_1'u_1 & v_1'v_1 & v_1' \\ u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1'u_1 & -u_1'v_1 & -u_1' \\ 0 & 0 & 0 & -u_2 & -v_2 & -1 & v_2'u_2 & v_2'v_2 & v_2' \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -u_2'u_2 & -u_2'v_2 & -u_2' \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \end{bmatrix}$$

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

RANSAC

1. Random subset - **At least 4 Point**
2. Model fitting: **minimize || Ah - 0 ||**

   Solution: SVD(A) = USV^T, h = v_9

3. Inliers: all points such that

   $$\mathbf{dist(u' , uH_{est}) + dist(u, H^{-1}u') > threshold}$$

4. If the number of inliers < threshold, keep trying

RANSAC: Pre match - SSD

287 Points

296 Points

180 Matches

RANSAC:

287 Points

296 Points

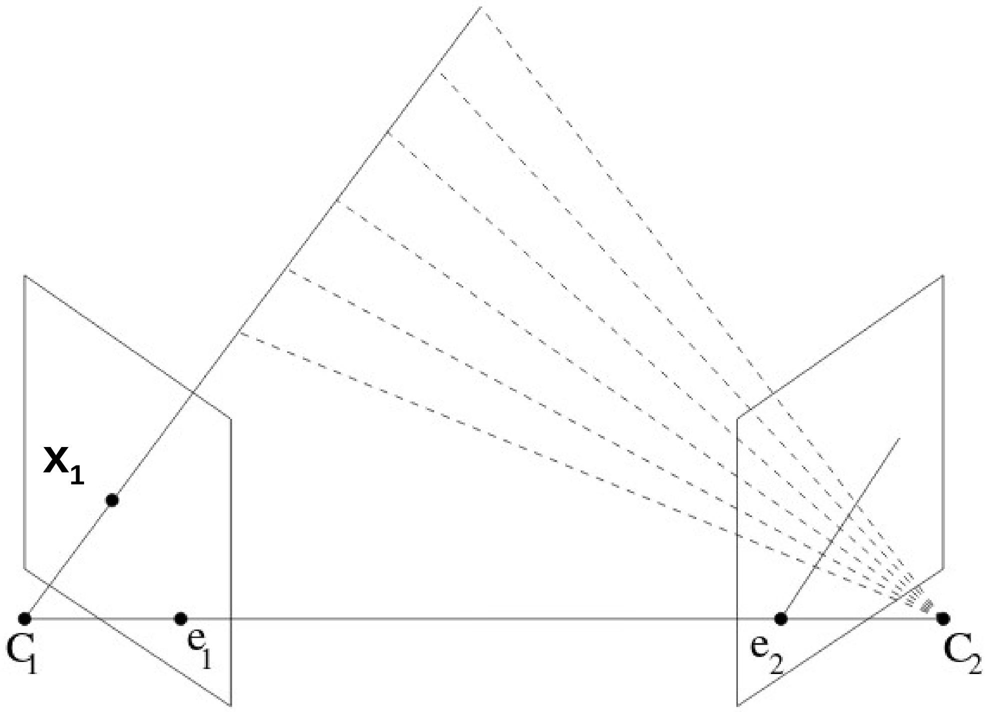
91 Matches

Pose Estimation

Pose Estimation: Fundamental Matrix

- It maps points in image 1 to lines in image 2
- Encapsulates the geometry between two views
- Require any knowledge of the camera's internal parameter

Pose Estimation: Fundamental Matrix



$$x_2^T F x_1 = 0$$

Pose Estimation: Fundamental Matrix

$$\begin{bmatrix} x'_i & y'_i & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = 0$$

Pose Estimation: Fundamental Matrix

$$\begin{bmatrix} x_i' & y_i' & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = 0$$

How do we find **F**?

Pose Estimation: Fundamental Matrix

Given m point correspondences…

$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_mx'_m & x_my'_m & x_m & y_mx'_m & y_my'_m & y_m & x'_m & y'_m & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{21} \\ f_{31} \\ f_{12} \\ f_{22} \\ f_{32} \\ f_{13} \\ f_{23} \\ f_{33} \end{bmatrix} = 0$$

The Eight Point Algorithm

$$\mathbf{A}\mathrm{f} = 0$$

Pose Estimation: Fundamental Matrix

The Eight Point Algorithm

$$\min \| \mathbf{A}f \|^2$$
$$\text{such that } \| f \|^2 = 1$$

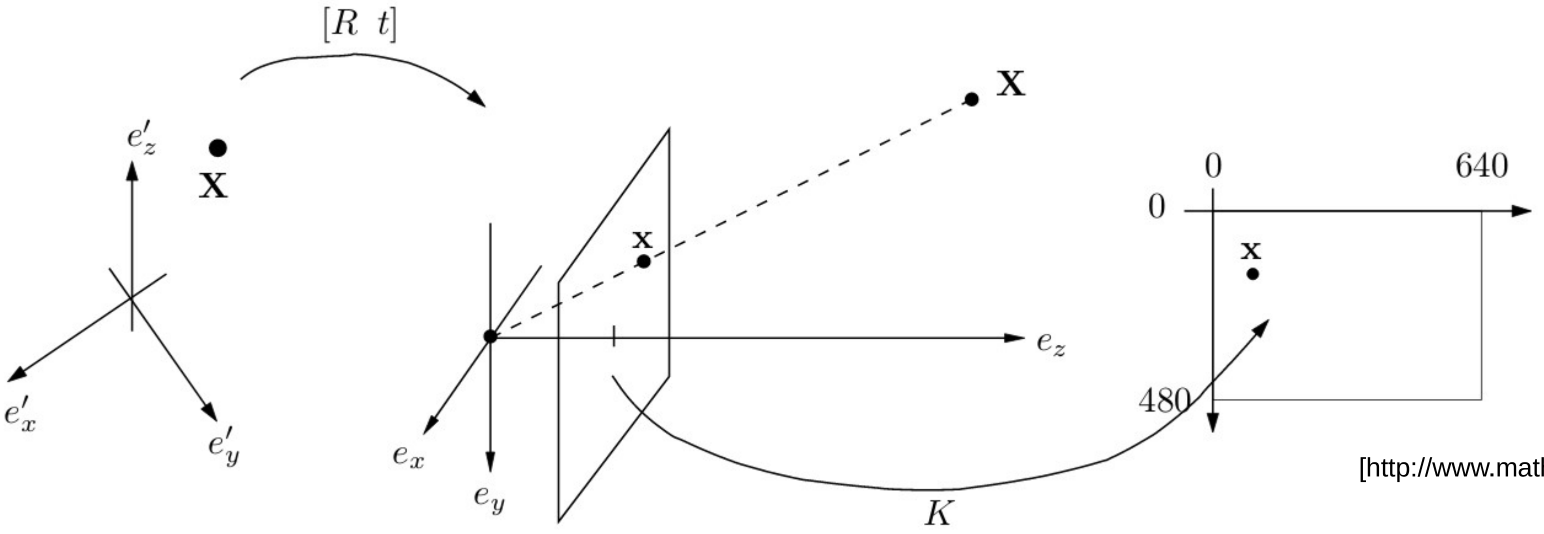
Solution: $SVD(A) = USV^T$, $f = v_9$

Pose Estimation: Fundamental Matrix

```
F =

   -0.0000    0.0000   -0.0125
   -0.0000   -0.0000    0.0190
    0.0150   -0.0177   -0.9995


ans =

  -1.4948e-04
```

$$x_2^T F x_1 = 0$$

Pose Estimation: Essential Matrix



[http://www.maths.lth.se]

$$\mathbf{x} = P\mathbf{X}$$

P = K [R | t]

Pose Estimation: Essential Matrix

$$\mathbf{x} = P\mathbf{X}$$

P = K [R | t]

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

	Left sensor	Right sensor
fx	1398.41	1401.62
fy	1398.41	1401.62
cx	923.113	944.482
cy	550.247	548.772

Pose Estimation: Essential Matrix

$$\mathbf{x} = P\mathbf{X} \qquad \mathbf{P = K\ [\ R\ |\ \ t\ ]}$$

$$E = K_2^T F K_1$$

- Relates normalized image points

Pose Estimation: Essential Matrix

$\mathbf{x} = P\mathbf{X}$ $\quad$ $\mathbf{P = K\,[\,R\,|\,t\,]}$ $\quad$ $K^{-1}x = [\,R\,|\,t\,]\,\mathbf{X}$

$x' = [R\,|\,t\,]\,\mathbf{X}$

$E = K_2^T F K_1$ $\quad$ $x' = P'\,\mathbf{X}$

- Relates normalized image points
  Normalized image coordinates have the origin at the optical center of the image

Pose Estimation: Rotation and Translation

The Essential Matrix for a pair of cameras of the form:

$\mathbf{P'}_1 = [\ I\ 0\ ]$ and $\mathbf{P'}_2 = [\ R\ t\ ]$

$$E = [t]_x R = SR$$

$$S = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$$

Pose Estimation: Rotation and Translation

$$E = [t]_x R = SR \qquad W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad Z = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

If $[U, \sim, V] = \mathrm{svd}(E)$

$$S_1 = -UZU^T,\ S_2 = -S_1$$

$$R_1 = -UW^TV^T,\ R_2 = UWV^T$$

Pose Estimation: Rotation and Translation

$$E = [t]_x R = SR$$

There are 4 possible solutions for $\mathbf{P_2}$**’**

$P_2' = [\ R_1 \ \ t\ ]$

$P_2' = [\ R_1 \ -t\ ]$

$P_2' = [\ R_2 \ \ t\ ]$

$P_2' = [\ R_2 \ -t\ ]$

Pose Estimation: Rotation and Translation

$$E = [t]_x R = SR$$

There are 4 possible solutions for $\mathbf{P_2'}$

$P_2' = [\ R_1\ \ t\ ]$

$P_2' = [\ R_1\ -t\ ]$

$P_2' = [\ R_2\ \ t\ ]$

$P_2' = [\ R_2\ -t\ ]$

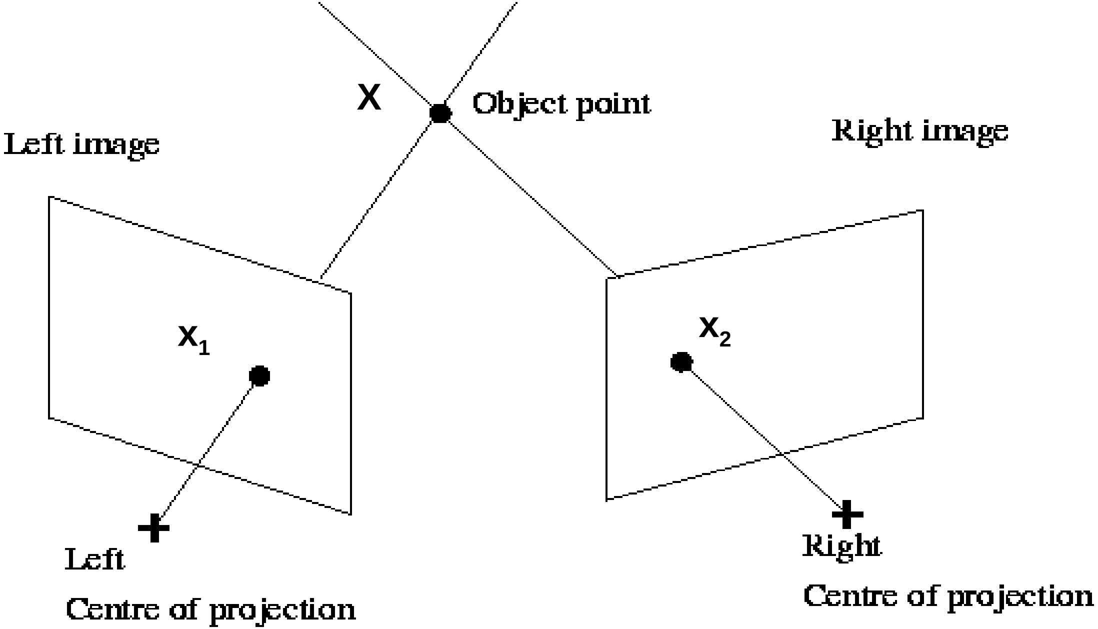
We have to test them all!!

Pose Estimation: Linear Triangulation

Recall:

$\mathbf{x_1} = P_1X$

$\mathbf{x_2} = P_2X$



Pose Estimation: Linear Triangulation

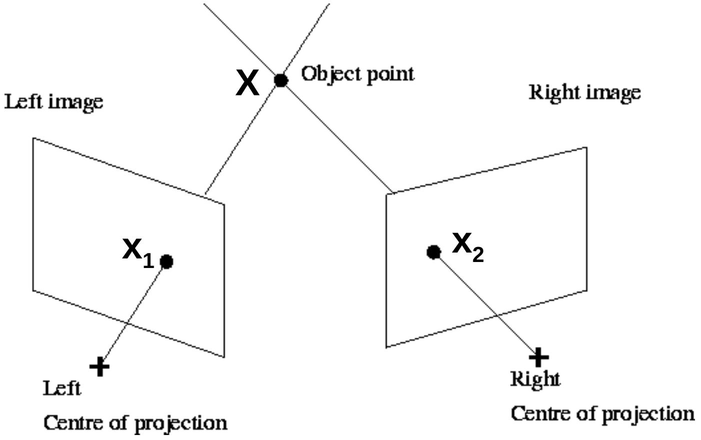
Recall:

$\mathbf{x_1} = P_1X$

$\mathbf{x_2} = P_2X$

$$P_i = \begin{bmatrix} p_i^{1T} \\ p_i^{2T} \\ p_i^{3T} \end{bmatrix}$$

$$\begin{bmatrix} x_1 p_1^{3T} - p_1^{1T} \\ y_1 p_1^{3T} - p_1^{2T} \\ x_2 p_2^{3T} - p_2^{1T} \\ y_2 p_2^{3T} - p_2^{2T} \end{bmatrix} \mathbf{X} = 0$$



Solve: **AX** = 0

Solution: SVD(**A**) = USV^T, $X = v_4$

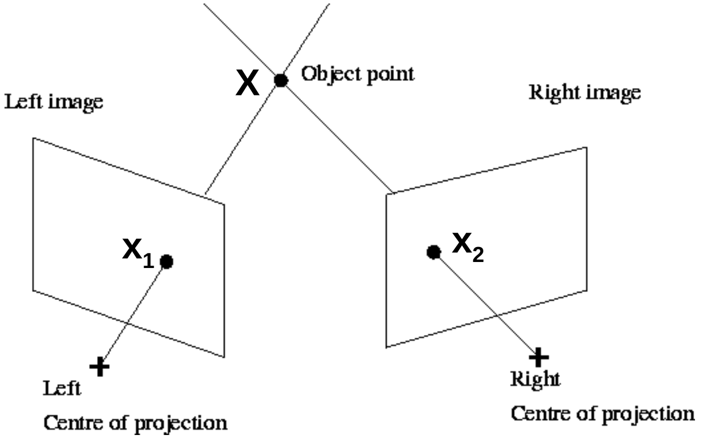
Pose Estimation: Linear Triangulation

Recall:

$\mathbf{x_1} = P_1 X$

$\mathbf{x_2} = P_2 X$

$$P_i = \begin{bmatrix} p_i^{1T} \\ p_i^{2T} \\ p_i^{3T} \end{bmatrix} \qquad \begin{bmatrix} x_1 p_1^{3T} - p_1^{1T} \\ y_1 p_1^{3T} - p_1^{2T} \\ x_2 p_2^{3T} - p_2^{1T} \\ y_2 p_2^{3T} - p_2^{2T} \end{bmatrix} \mathbf{X} = 0$$

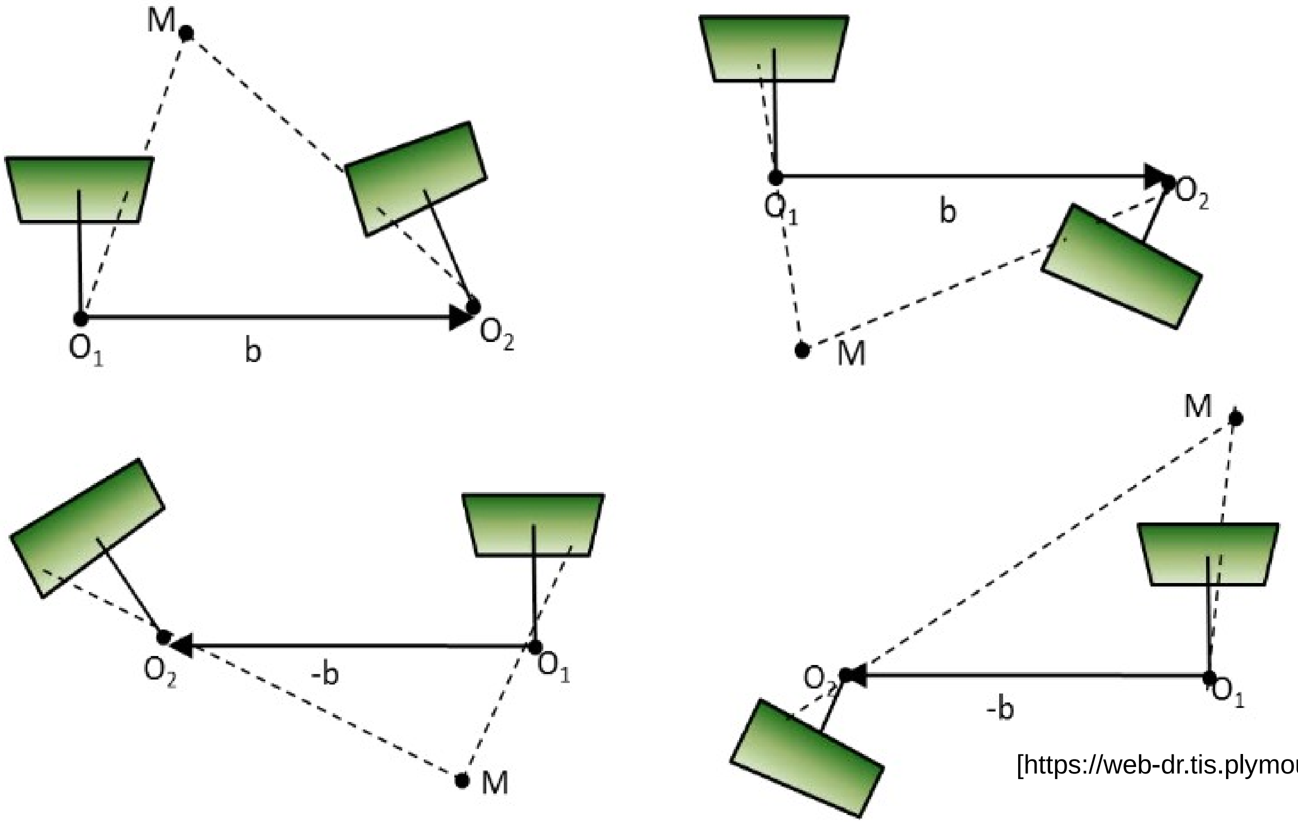


For each point!

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

Solution: $SVD(\mathbf{A}) = USV^T$, $X = v_4$

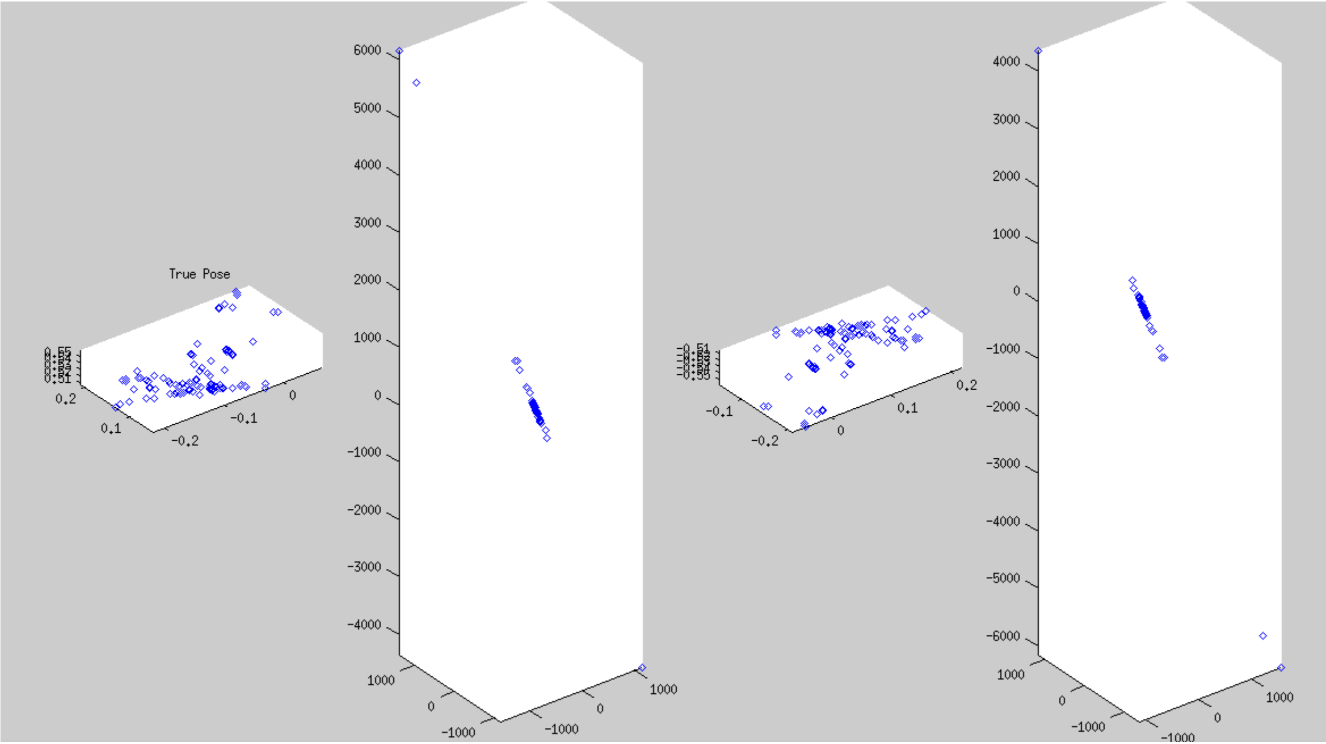
Pose Estimation: Linear Triangulation



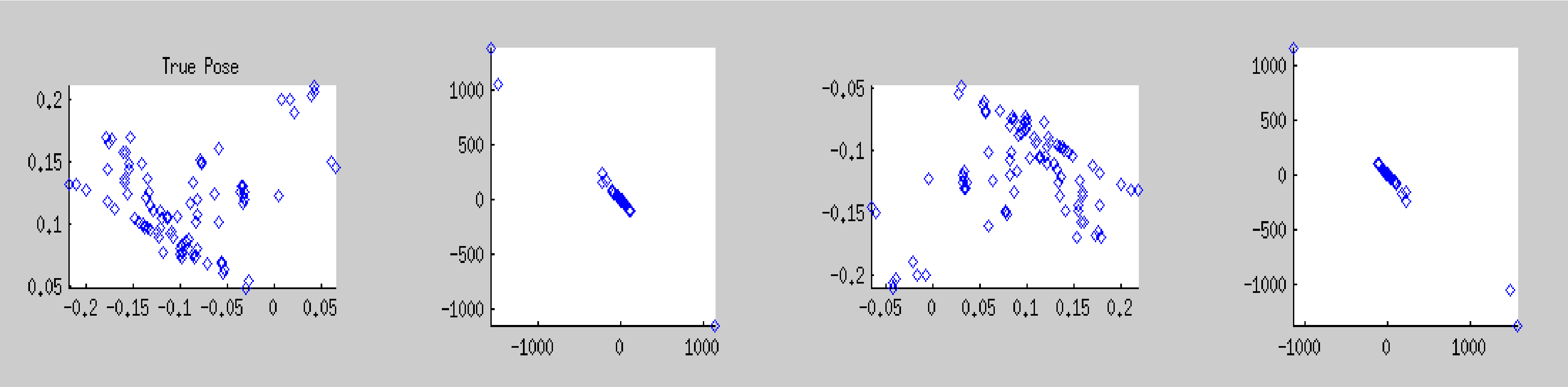
[https://web-dr.tis.plymouth.ac.uk]

Pose Estimation: Linear Triangulation

Pose Estimation: Linear Triangulation

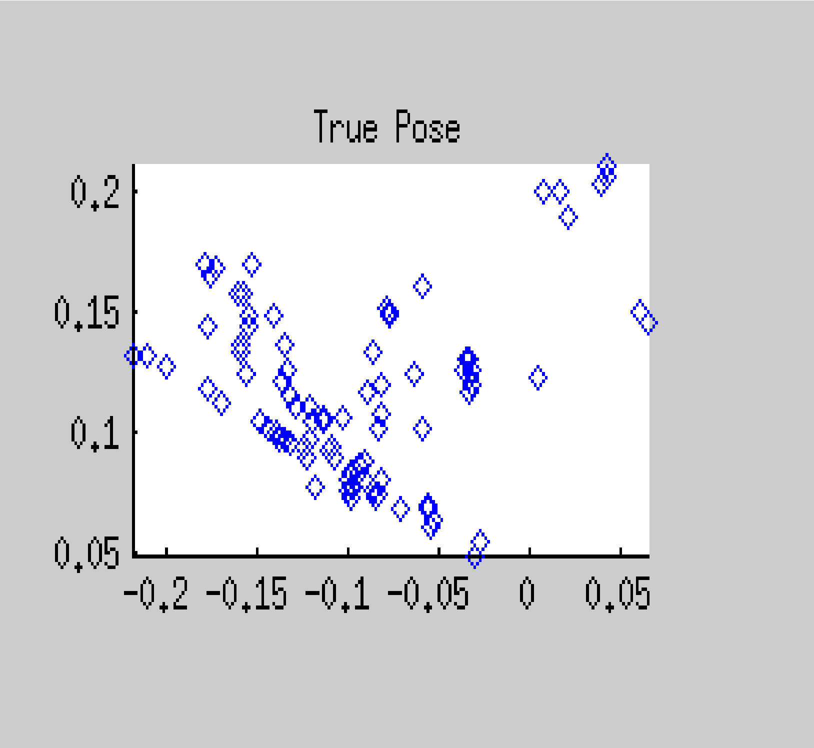


Pose Estimation: Linear Triangulation



X-Y view

Pose Estimation: Linear Triangulation



X-Y view

Conclusions:

- Feature detection is important, but feature description is critical
  - Scaling, rotation, translation, and other transformations
  - Matching is based on feature descriptors

- Computational Expensive: SVD

- It’s a recipe: Implementation is straightforward

Questions?