Panoramas

Kari Pulli
Research Fellow
Nokia Research Center Palo Alto

Are you getting the whole picture?

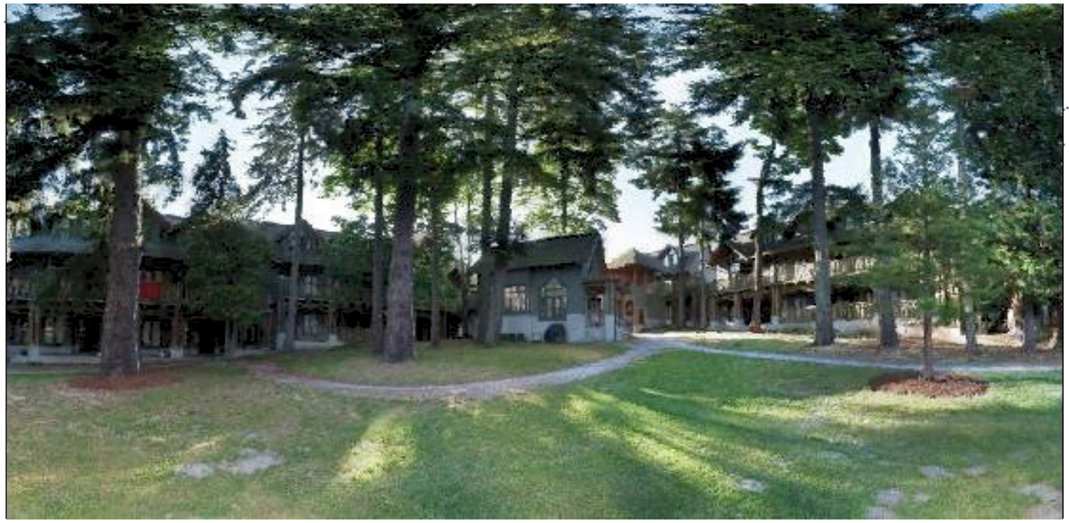
Compact Camera FOV = 50 x 35°

Slide from Brown & Lowe

Are you getting the whole picture?

Compact Camera FOV = 50 x 35°

Human FOV = 200 x 135°



NOKIA

Slide from Brown & Lowe

Are you getting the whole picture?

Compact Camera FOV	= 50 x 35°
Human FOV	= 200 x 135°
Panoramic Mosaic	= 360 x 180°

Slide from Brown & Lowe

Panorama
A wide-angle representation of the scene

Panorama of *Along the River During Qingming Festival*, 18th century remake of a 12th century original by Chinese artist Zhang Zeduan.

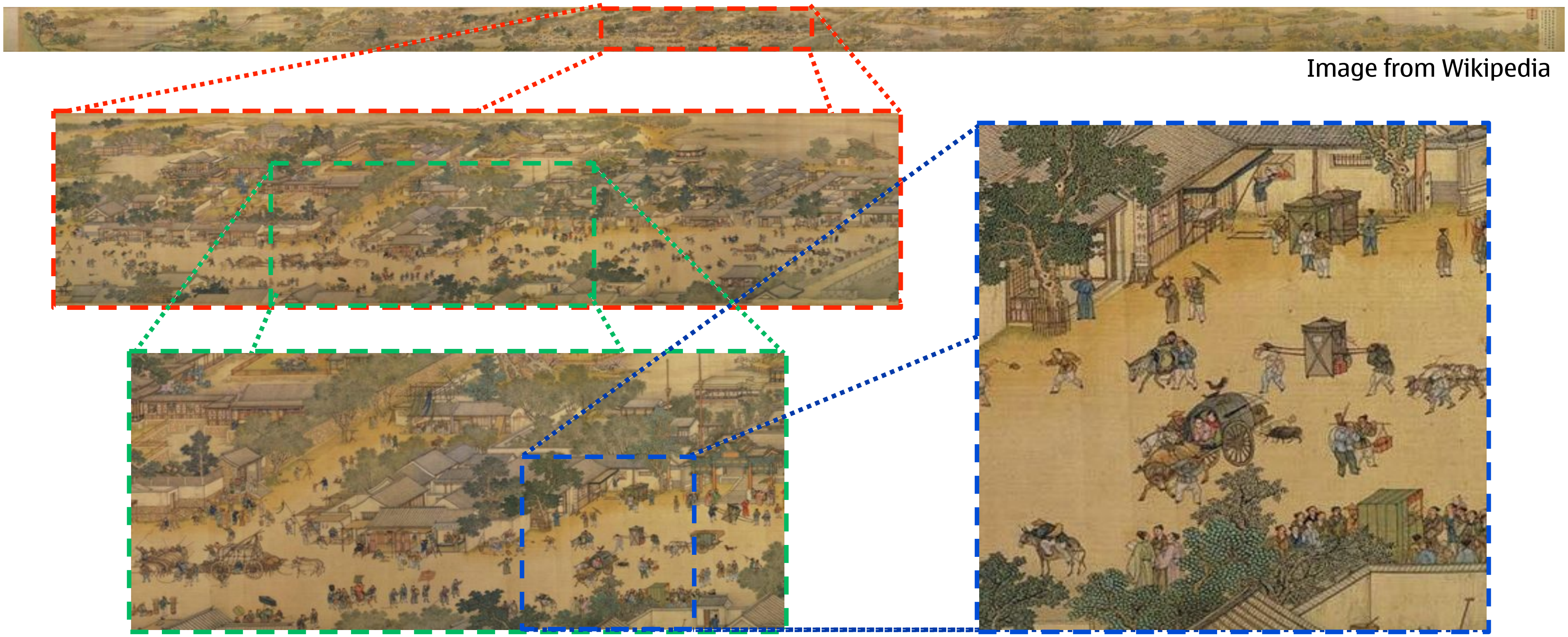
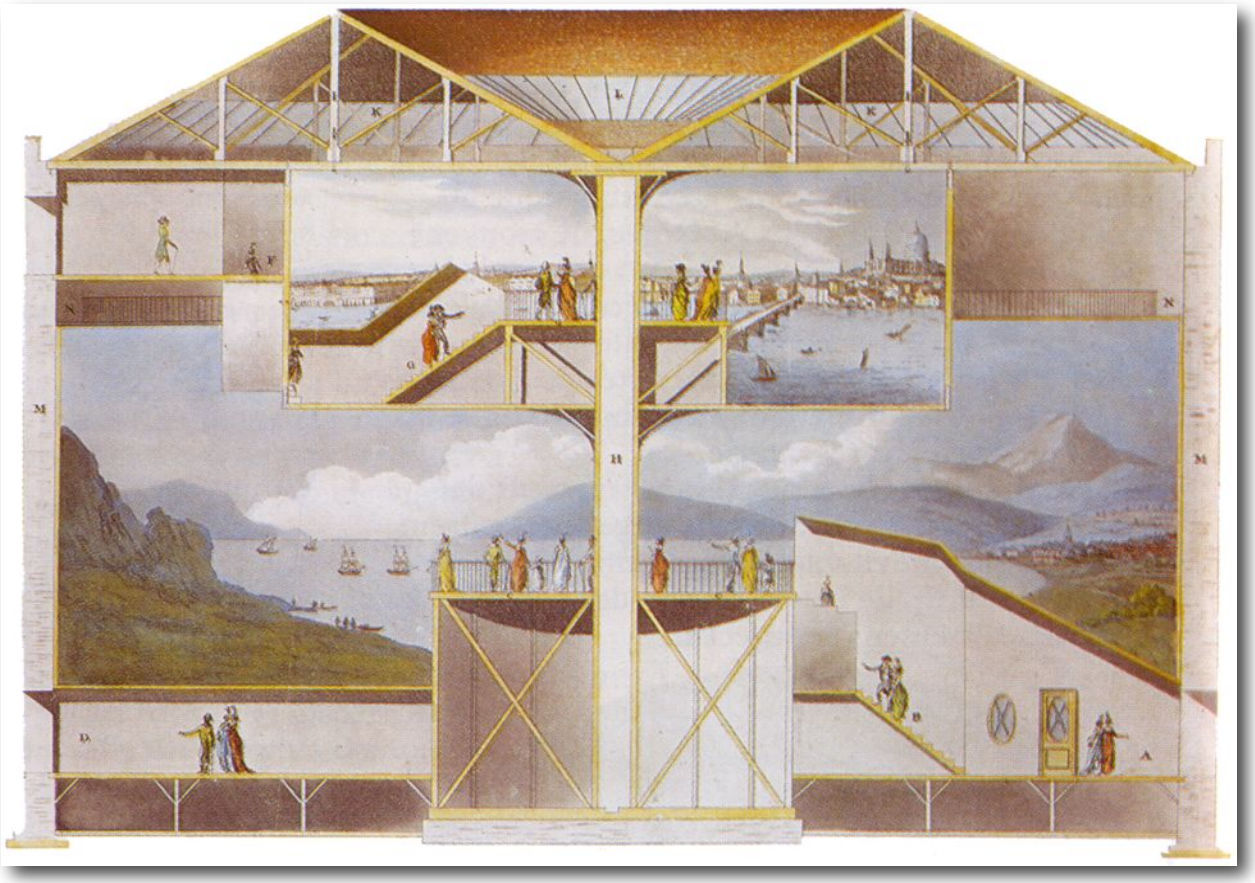


Image from Wikipedia

NOKIA

Panorama: Cinema for the early 19th century



Burford's Panorama, Leicester Square, London, 1801

Painting by Robert Mitchell

Creating panoramas with wide-angle optics

http://www.0-360.com

AF DX Fisheye-NIKKOR 10.5mm f/2.8G ED

Rotation cameras

Idea

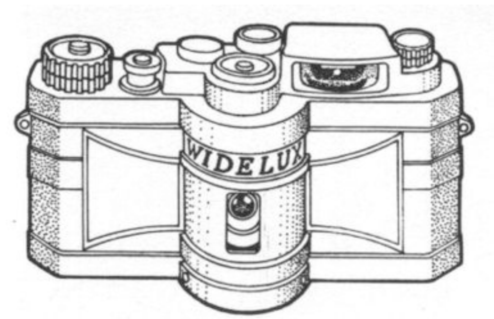
- rotate camera or lens so that a vertical slit is exposed

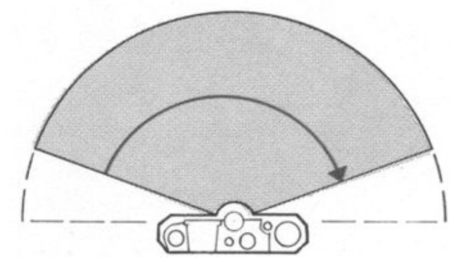
Swing lens

- rotate the lens and a vertical slit (or the sensor)
- typically can get 110-140 degree panoramas
- Widelux, Seitz, ...

Full rotation

- whole camera rotates
- can get 360 degree panoramas
- Panoscan, Roundshot, ...





Swing-lens panoramic images

San Francisco in ruins, 1906

101 Ranch, Oklahoma, circa 1920

NOKIA

Flatback panoramic camera

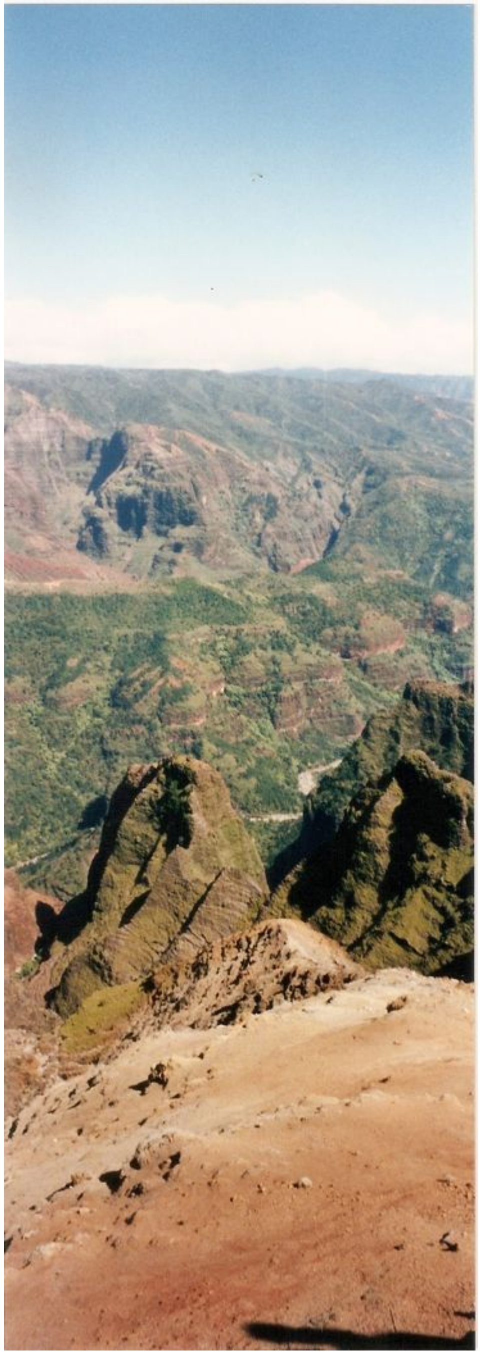
Lee Frost, Val D'Orcia, Tuscany, Italy

Disposable panoramic camera
wide-angle lens, limited vertical FOV

NOKIA


Press

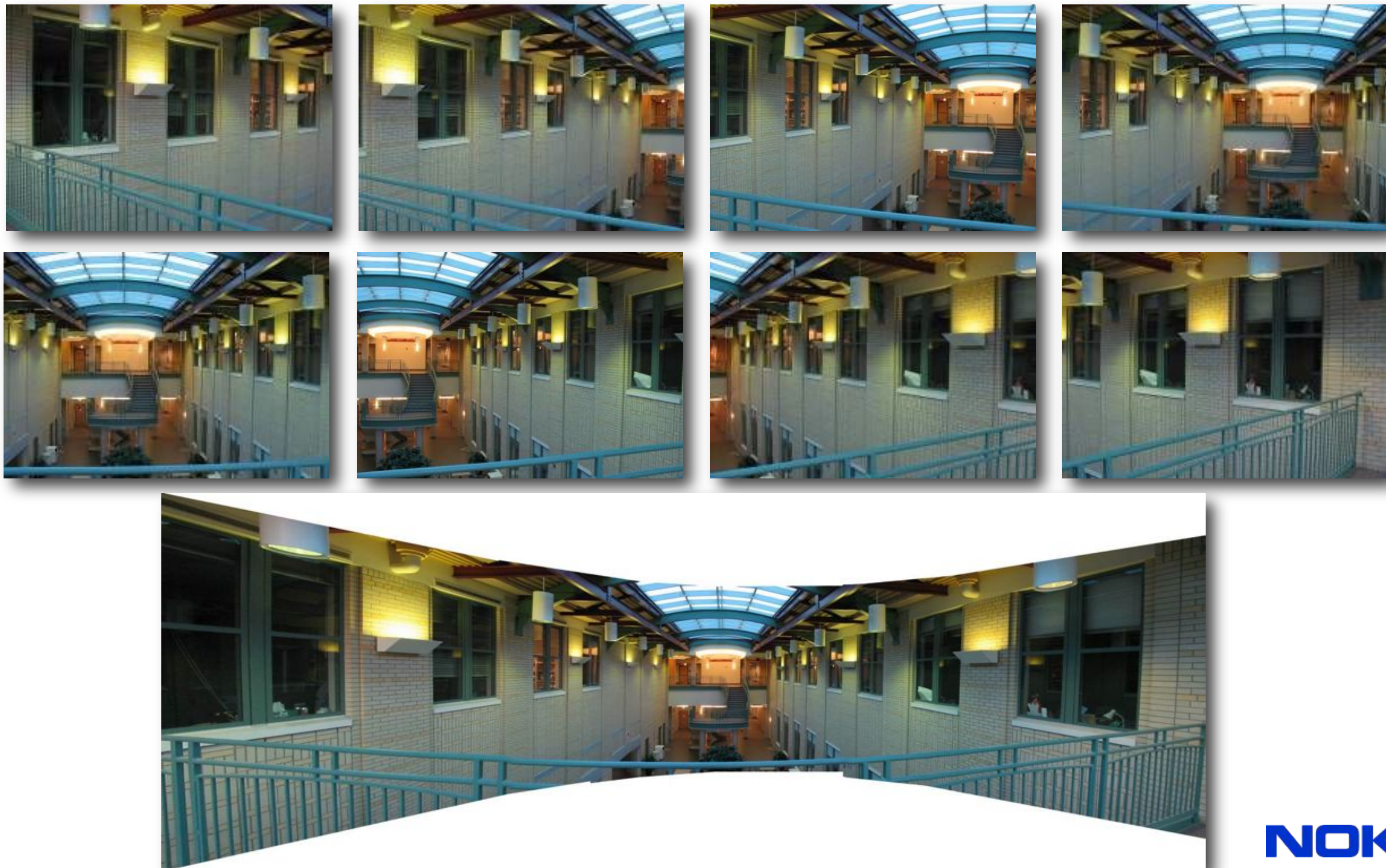

Press



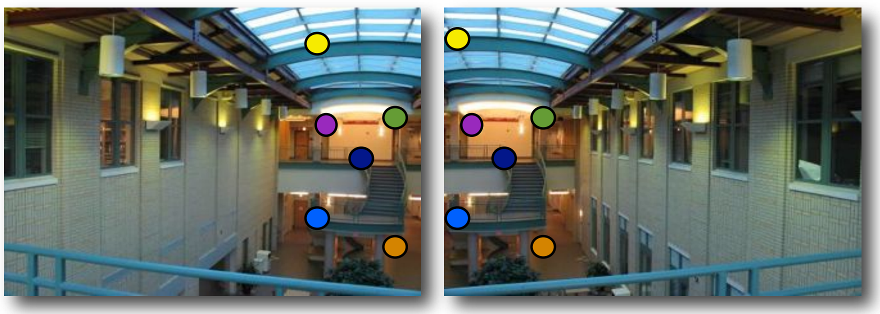

Press


Press

Stitching images together to make a mosaic



Stitching images together to make a mosaic



Given a set of images that should stitch together

- by rotating the camera around its center of perspective

Step 1: Find corresponding features in a pair of image

Step 2: Compute transformation from 2nd to 1st image

Step 3: Warp 2nd image so it overlays 1st image

Step 4: Blend images where they overlap one another

repeat for 3rd image and mosaic of first two, etc.

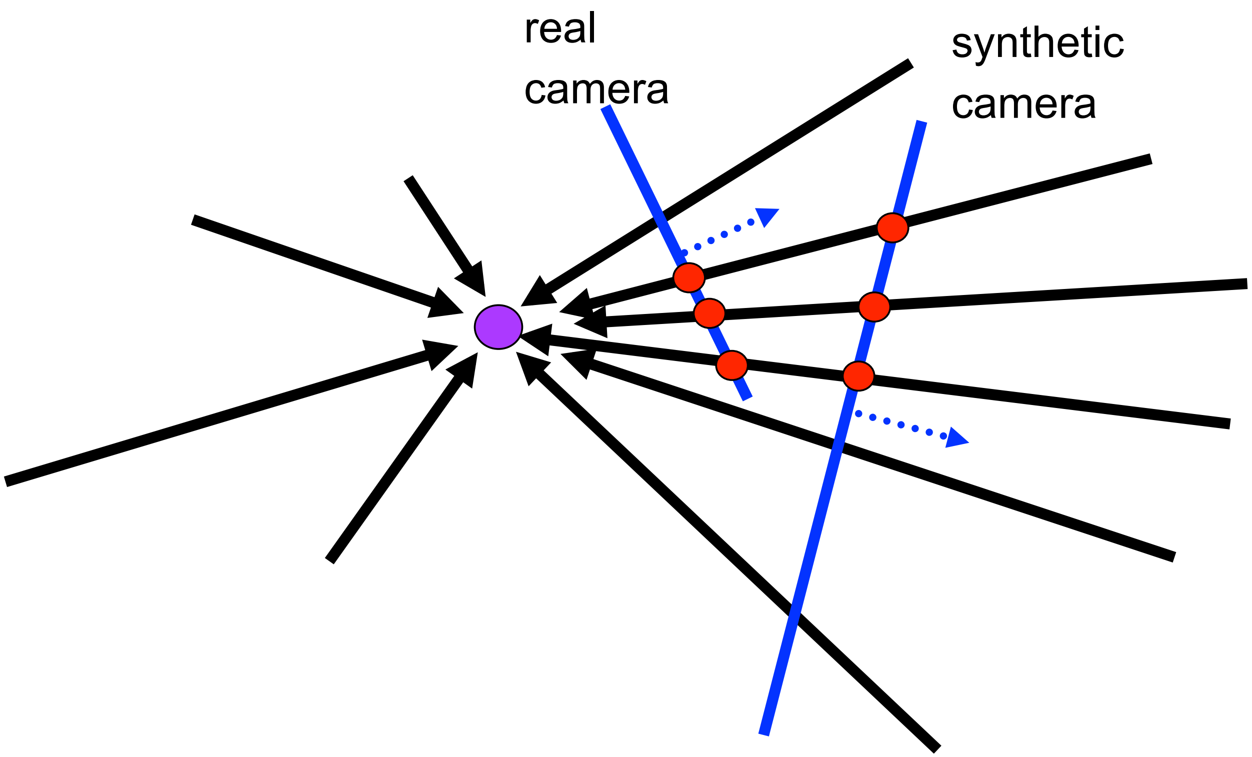
Aligning images: Translation?

left on top

right on top

Translations are not enough to align the images

A pencil of rays contains all views



Can generate any synthetic camera view
as long as it has **the same center of projection**!
… and scene geometry does not matter …

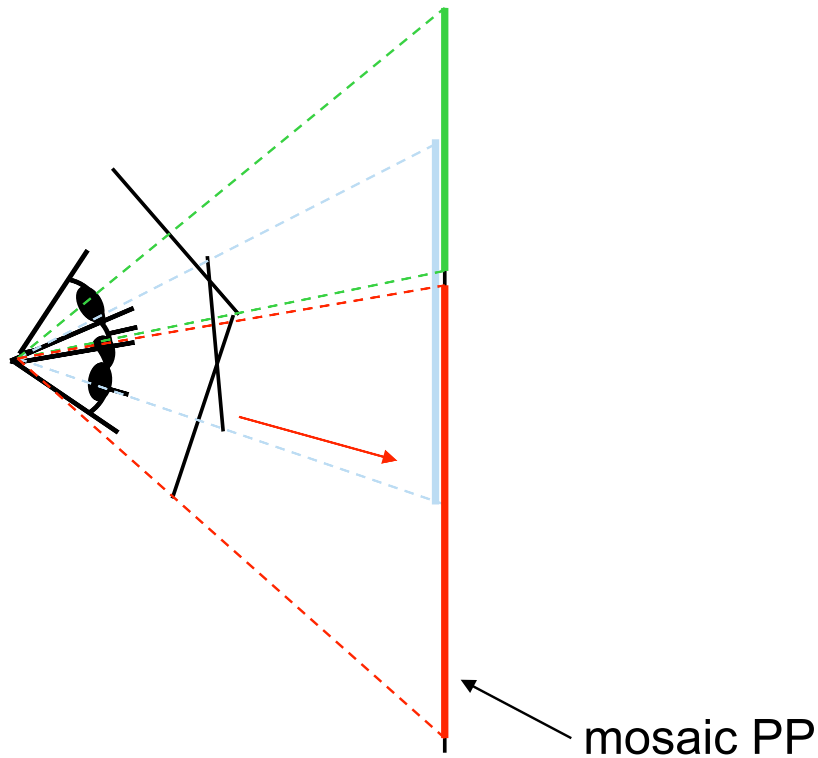
Reprojecting an image onto a different picture plane

the sidewalk art of Julian Beever

the view on any picture plane can be projected onto any other plane in 3D without changing its appearance as seen from the center of projection

NOKIA

Image reprojection

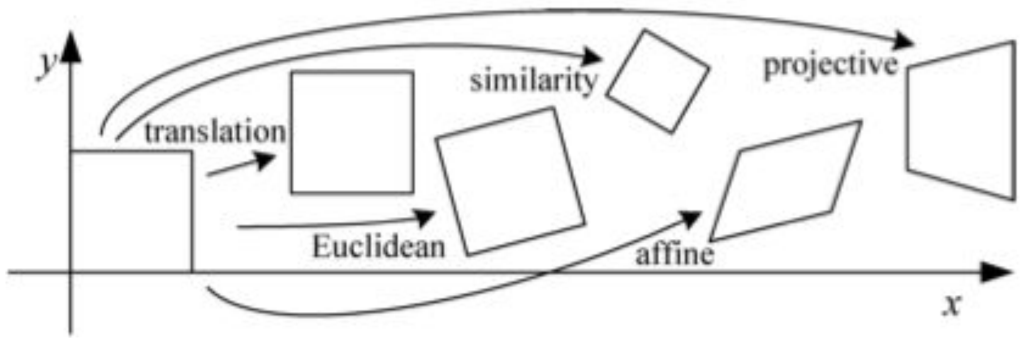


The mosaic has a natural interpretation in 3D

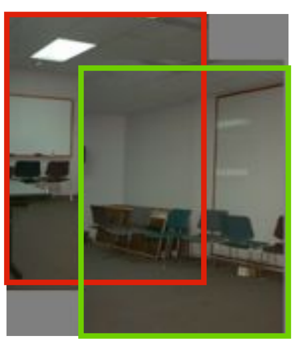
- the images are reprojected onto a common plane
- the mosaic is formed on this plane
- mosaic is a *synthetic wide-angle camera*

NOKIA

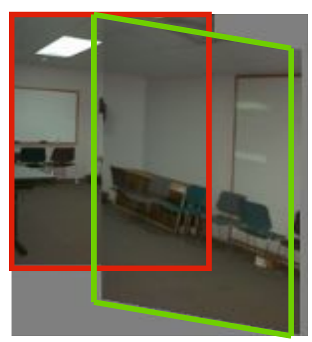
Which transform to use?



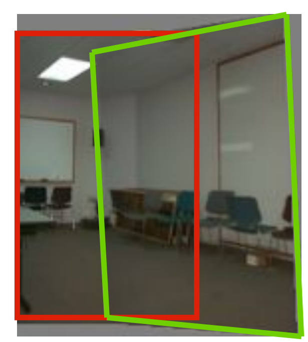
Translation | **Affine** | **Perspective**



2 unknowns



6 unknowns



8 unknowns

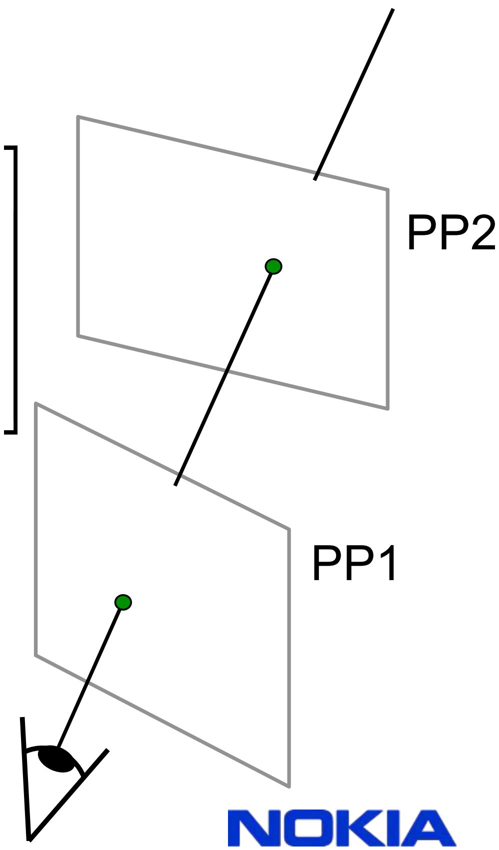
Homography

Projective mapping between any two PPs with the same center of projection

- rectangle should map to arbitrary quadrilateral
- parallel lines aren't
- but must preserve straight lines

called Homography

$$\underset{\mathbf{p'}}{\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix}} = \underset{\mathbf{H}}{\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}} \underset{\mathbf{p}}{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}$$



To apply a homography **H**

- Compute **p' = Hp** (regular matrix multiply)
- Convert **p'** from homogeneous to image coordinates [x',y'] (divide by w)

Homography from mapping quads

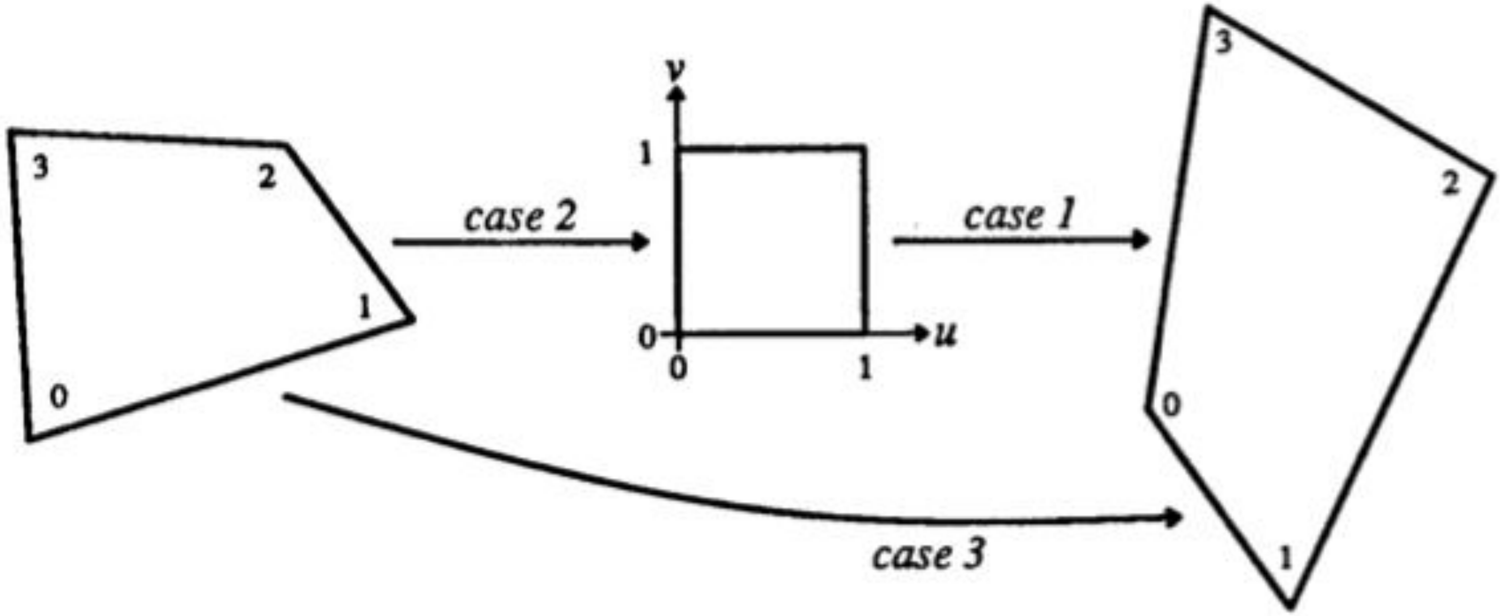


Figure 2.8: *Quadrilateral to quadrilateral mapping as a composition of simpler mappings.*

Fundamentals of Texture Mapping and Image Warping
Paul Heckbert, M.Sc. thesis, U.C. Berkeley, June 1989, 86 pp.
http://www.cs.cmu.edu/~ph/texfund/texfund.pdf

Homography from *n* point correspondences

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

p' **H** **p**

Multiply out

- $wx' = h_{11} x + h_{12} y + h_{13}$
- $wy' = h_{21} x + h_{22} y + h_{23}$
- $w = h_{31} x + h_{32} y + h_{33}$

Get rid of w

- $(h_{31} x + h_{32} y + h_{33})x' – (h_{11} x + h_{12} y + h_{13}) = 0$
- $(h_{31} x + h_{32} y + h_{33})y' – (h_{21} x + h_{22} y + h_{23}) = 0$

Create a new system Ah = 0

- Each point constraint gives two rows of A
- [-x -y -1 0 0 0 xx' yx' x']
- [0 0 0 -x -y -1 xy' yy' y']

$$h = \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix}$$

Solve with singular value decomposition of $A = USV^T$

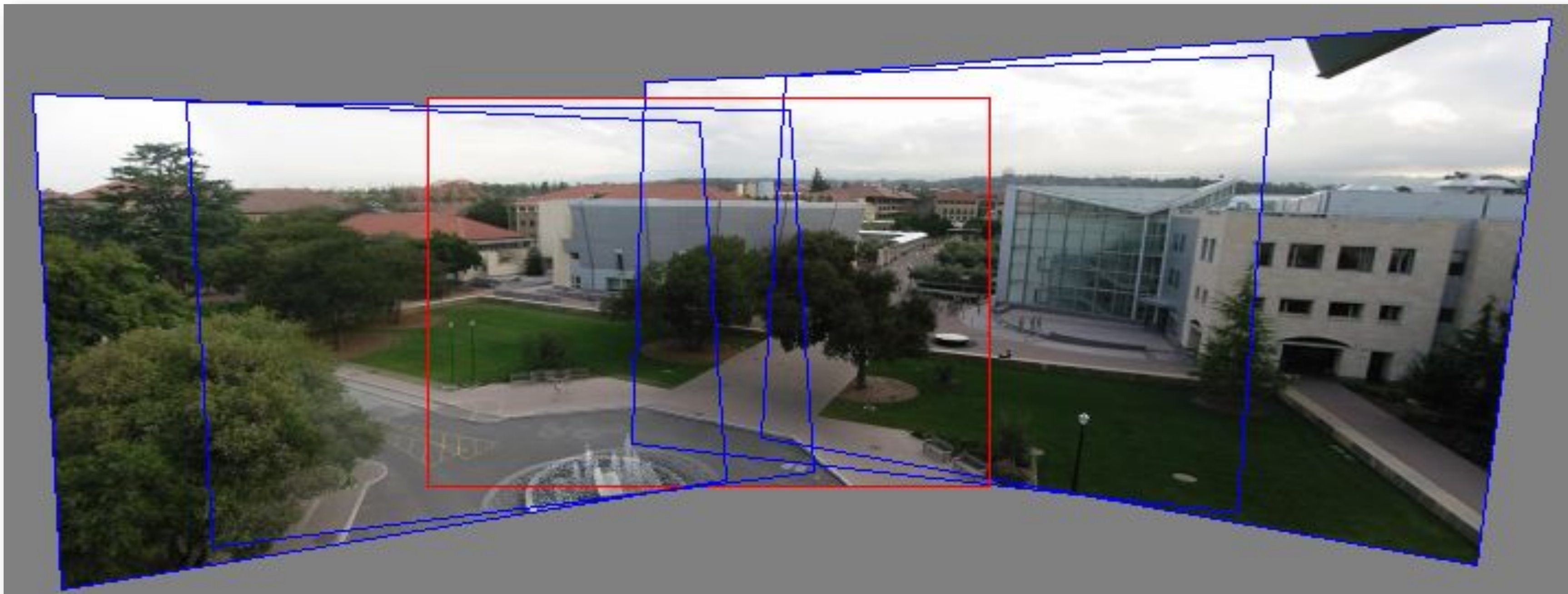
- solution is in the nullspace of A
- the last column of V (= last row of V^T)

NOKIA

Python test code

```
from numpy import *

# create original points
X = ones([3,4])
X[:2,:] = random.rand(2,4)
x,y = X[0],X[1]
# create projective matrix
H = random.rand(3,3)
# create the target points
Y = dot(H,X)
# homogeneous division
YY = (Y / Y[2])[:2,:]
u,v = YY[0],YY[1]

A = zeros([8,9])
for i in range(4):
    A[2*i  ] = [-x[i], -y[i], -1,     0,     0,  0, x[i] * u[i], y[i] * u[i], u[i]]
    A[2*i+1] = [    0,     0,  0, -x[i], -y[i], -1, x[i] * v[i], y[i] * v[i], v[i]]

[u,s,vt] = linalg.svd(A)

# reorder the last row of vt to 3x3 matrix
HH = vt[-1,:].reshape([3,3])

# test that the matrices are the same (within a multiplicative factor)
print H - HH * (H[2,2] / HH[2,2])
```

Summary of perspective stitching



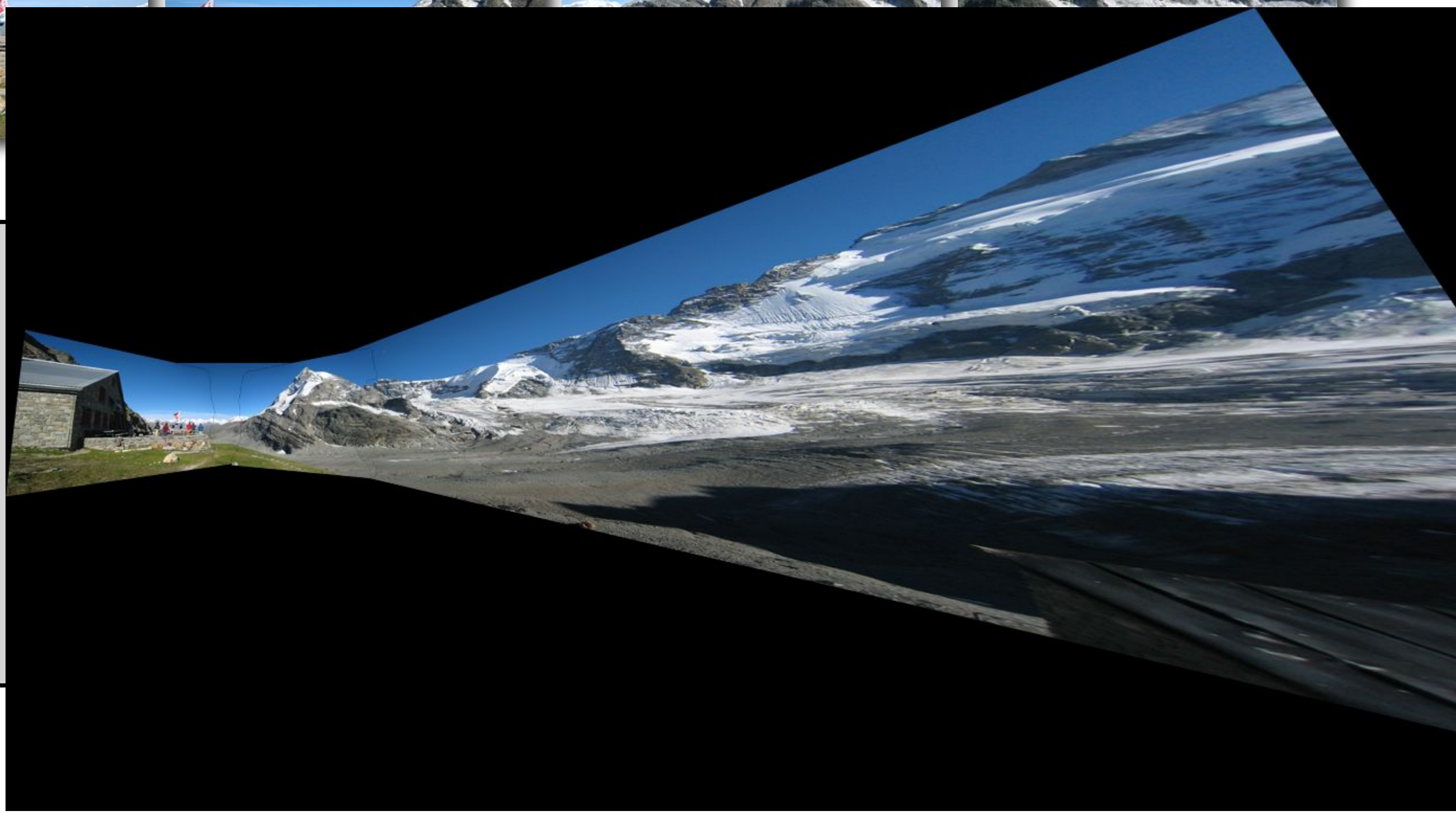
- Pick one image, typically the central view (red outline)
- Warp the others to its plane
- Blend

Example

common picture plane of mosaic image

perspective reprojection

Using 4 shots instead of 3



Back to 3 shots

cylindrical reprojection

Back to 3 shots

surface of cylinder

cylindrical reprojection

NOKIA

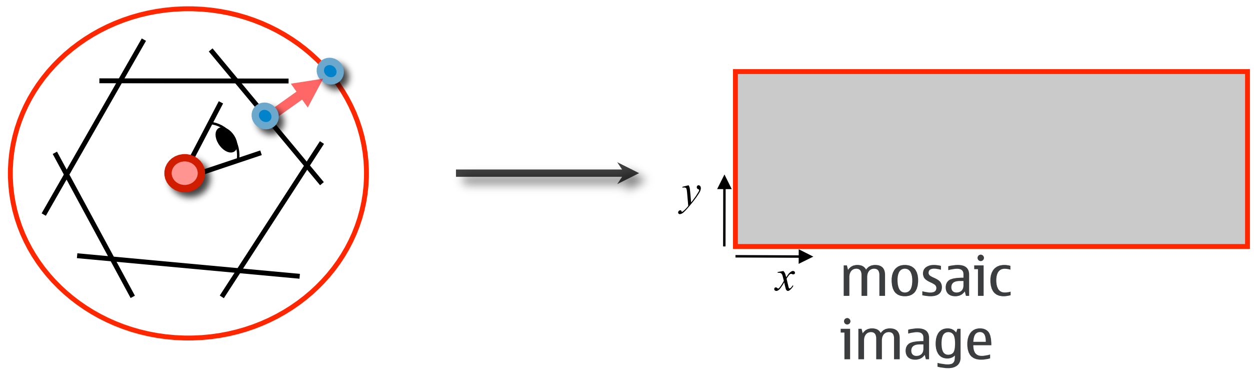
Back to 3 shots



perspective reprojection

Cylindrical panoramas

What if you want a 360° panorama?

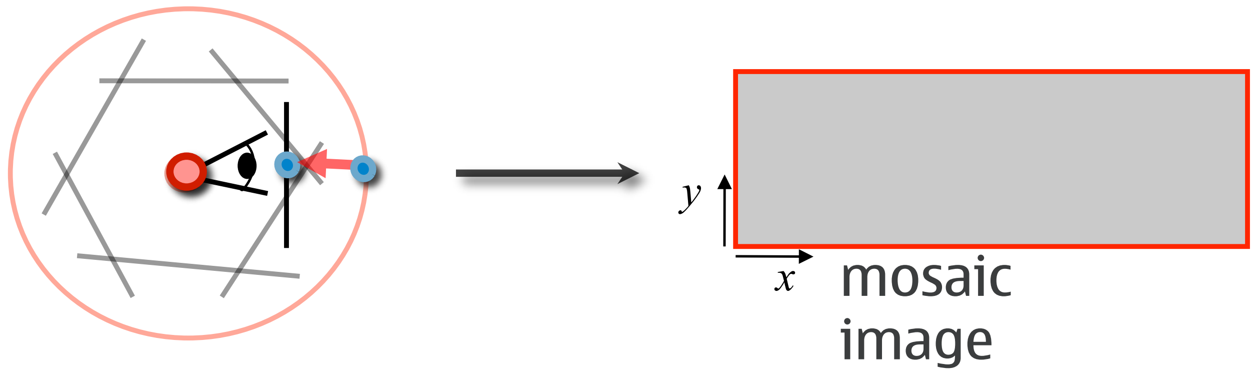


Project each image onto a cylinder

A cylindrical image is a rectangular array

Cylindrical panoramas

What if you want a 360° panorama?



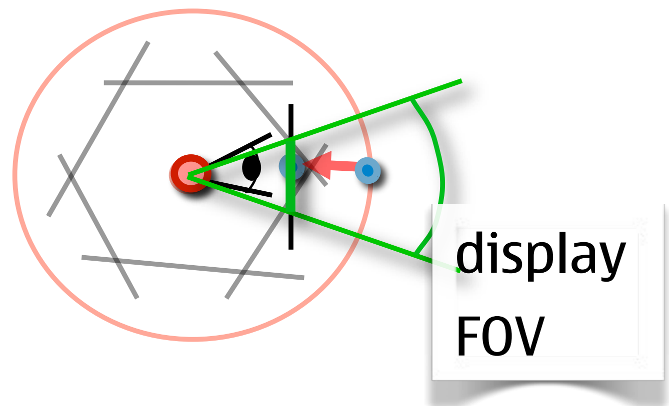
Project each image onto a cylinder

A cylindrical image is a rectangular array

To view without distortion

- reproject a portion of the cylinder onto a picture plane representing the display screen

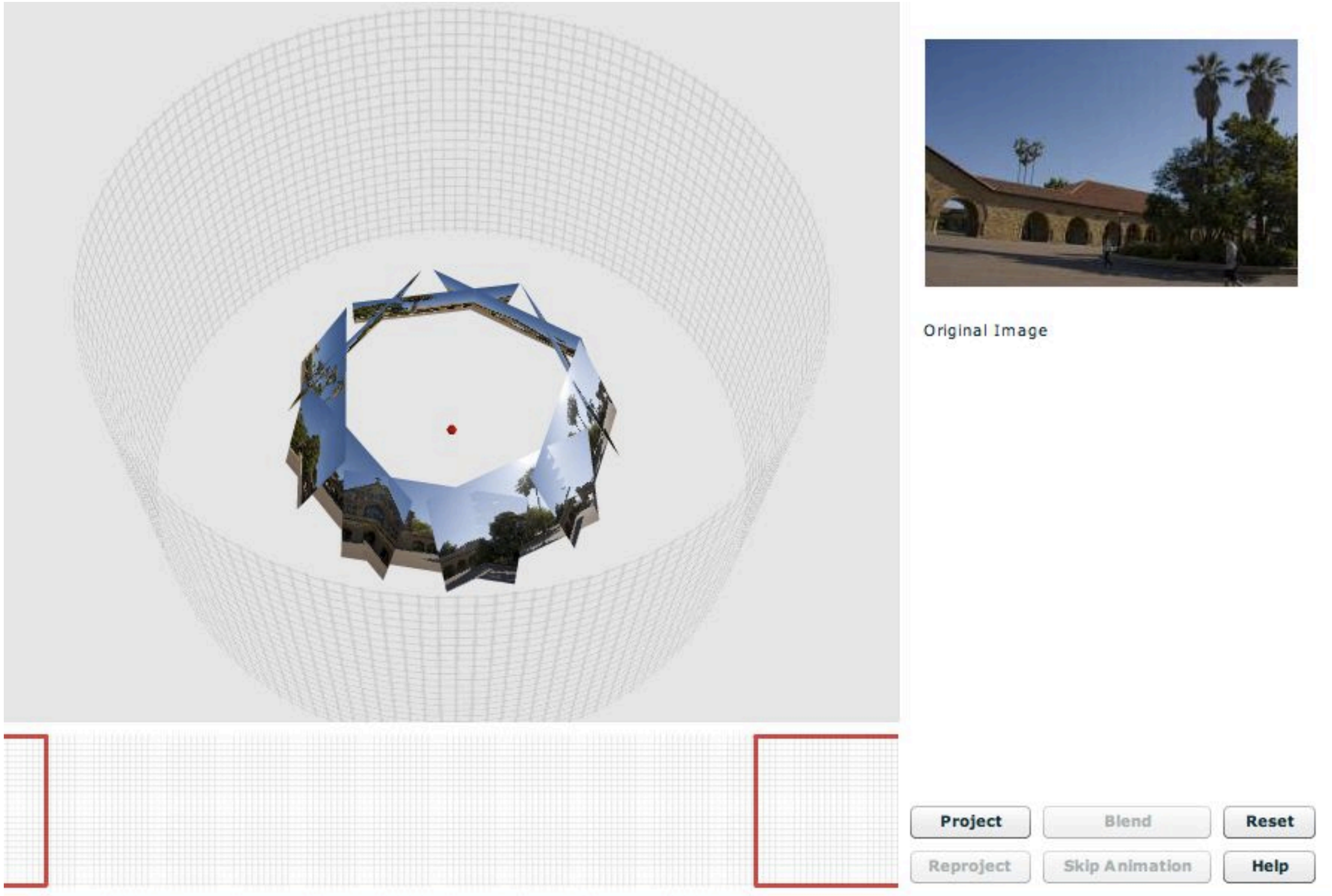
2nd reprojection to a plane for display



Imagine photographing the inside of a cylinder that is wallpapered with this panorama

- if your FOV is narrow, your photo won't be too distorted

Demo

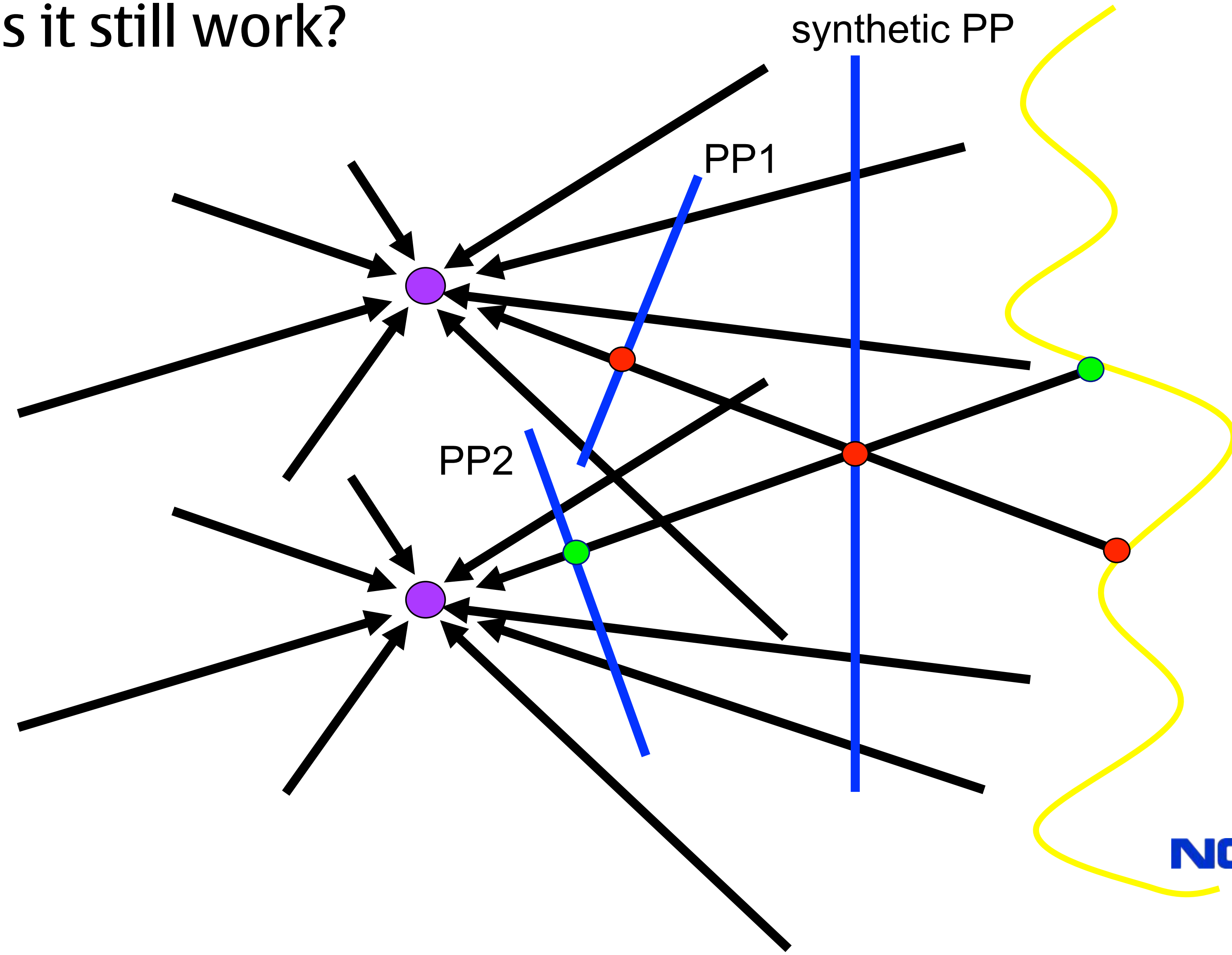


http://graphics.stanford.edu/courses/cs178/applets/projection.html

NOKIA

Changing camera center

Does it still work?



Where to rotate? Nodal point?

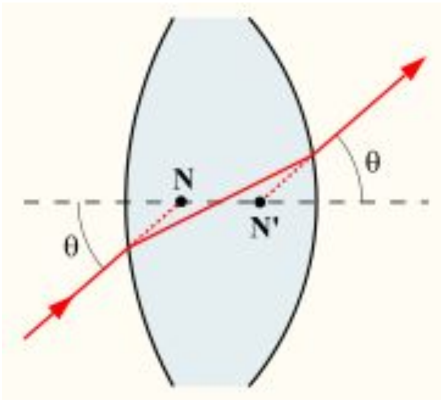
http://www.reallyrightstuff.com/pano/index.html

NOKIA

Rotate around the center of lens perspective

Many instructions say rotate around the nodal point

- wrong! http://toothwalker.org/optics/misconceptions.html#m6



Correct: the entrance pupil

- the optical image of the physical aperture stop as 'seen' through the front of the lens system
- due to the magnifying effect of the front lens, the entrance pupil's location is nearer than that of the physical aperture

The front and rear nodal points have the property that a ray aimed at one of them will be refracted by the lens such that it appears to have come from the other, and with the same angle with respect to the optical axis.

NOKIA

Test for parallax

Figure 3. Configuration to reveal the presence or absence of parallax. The subject is first placed at the left side of the frame, and subsequently at the right side after rotation of the camera about a vertical axis with the help of a panoramic tripod head.

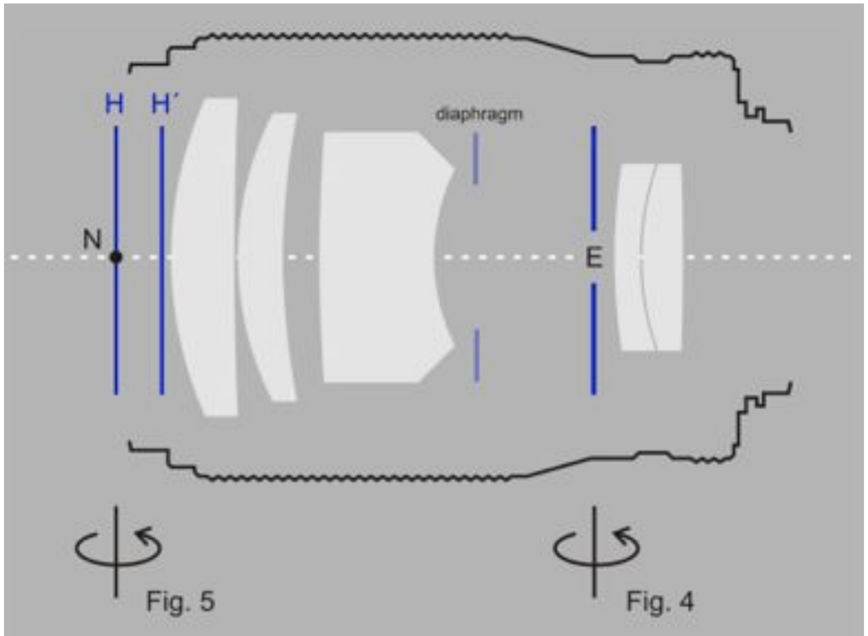


Figure 2. Diagram of a 135/2.8 lens with rotation axes through the front nodal point N and entrance pupil E.

http://toothwalker.org/optics/cop.html#stitching

NOKIA

Wrong center of rotation -> parallax

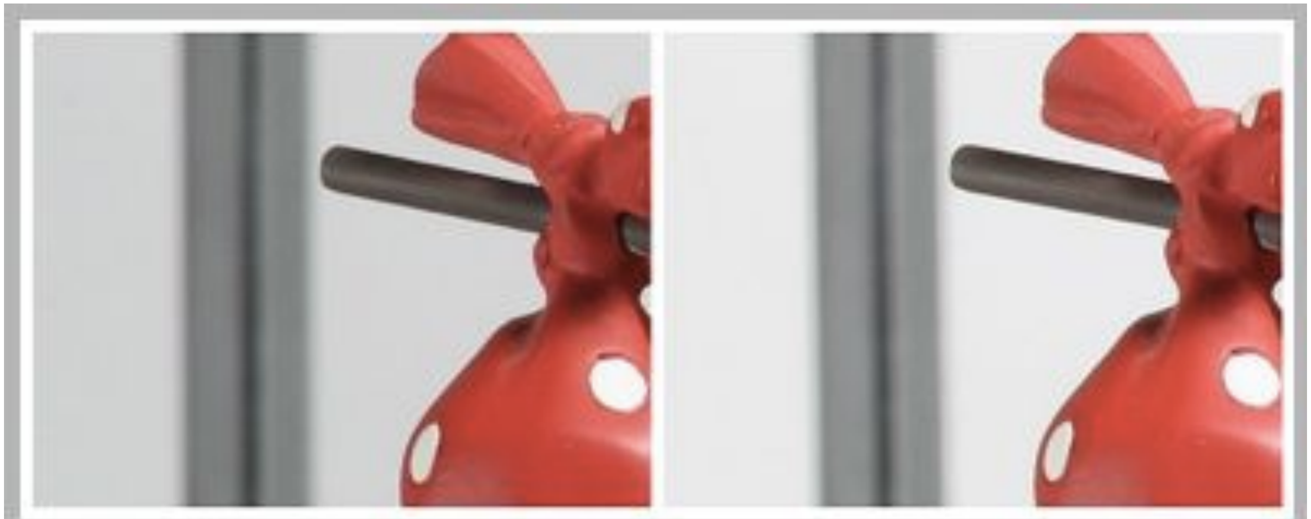


Figure 4. Rotation about an axis through the entrance pupil.

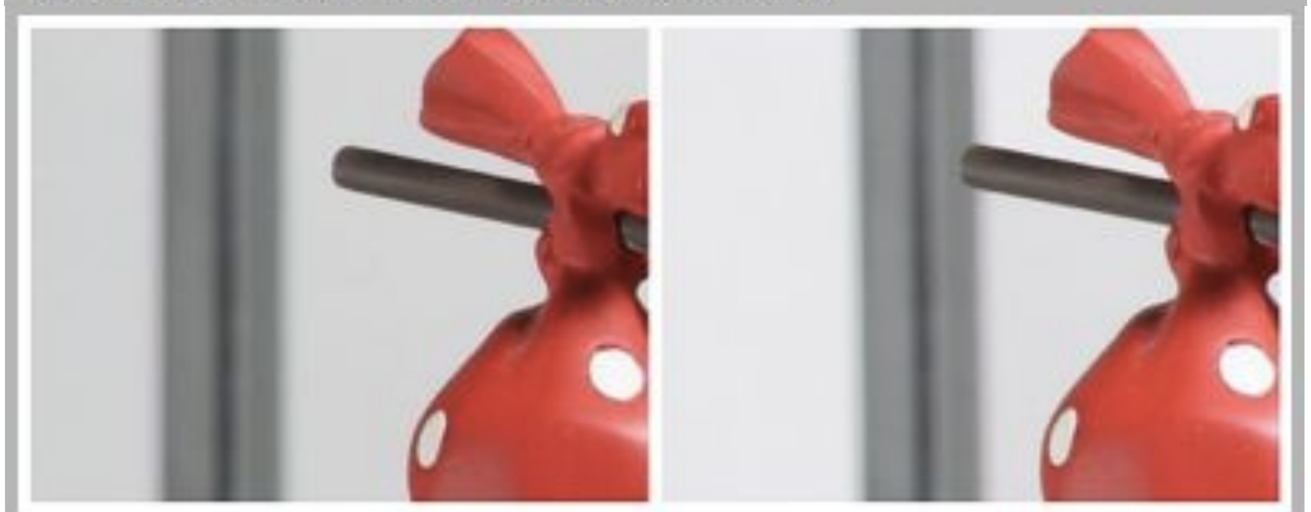
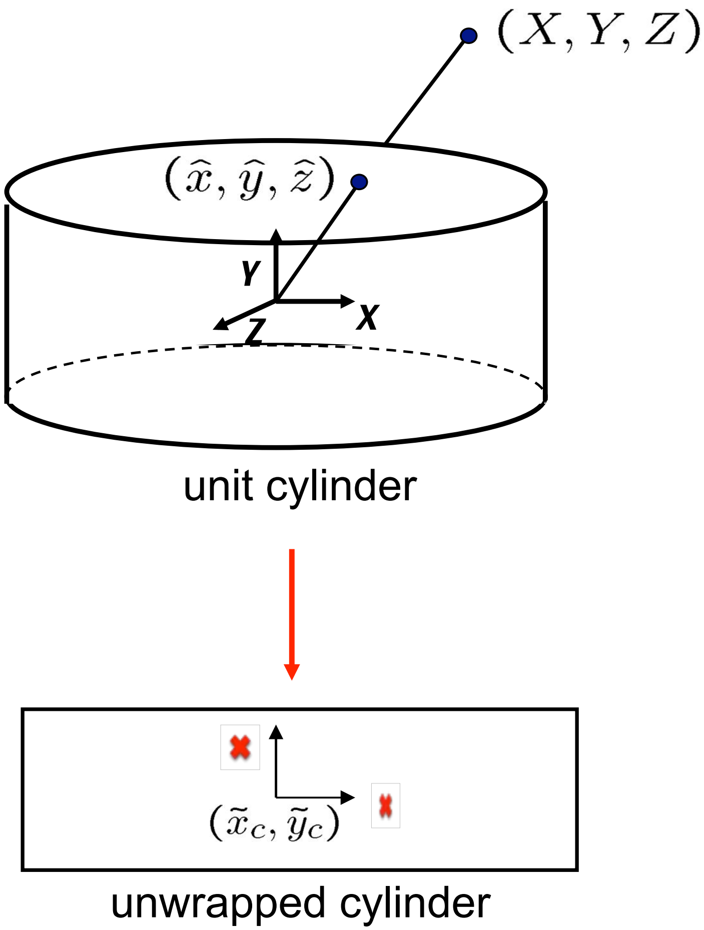


Figure 5. Rotation about an axis through the front nodal point.

NOKIA

Cylindrical projection



- Map 3D point (X,Y,Z) onto cylinder

$$(\hat{x}, \hat{y}, \hat{z}) = \frac{1}{\sqrt{X^2+Z^2}}(X, Y, Z)$$

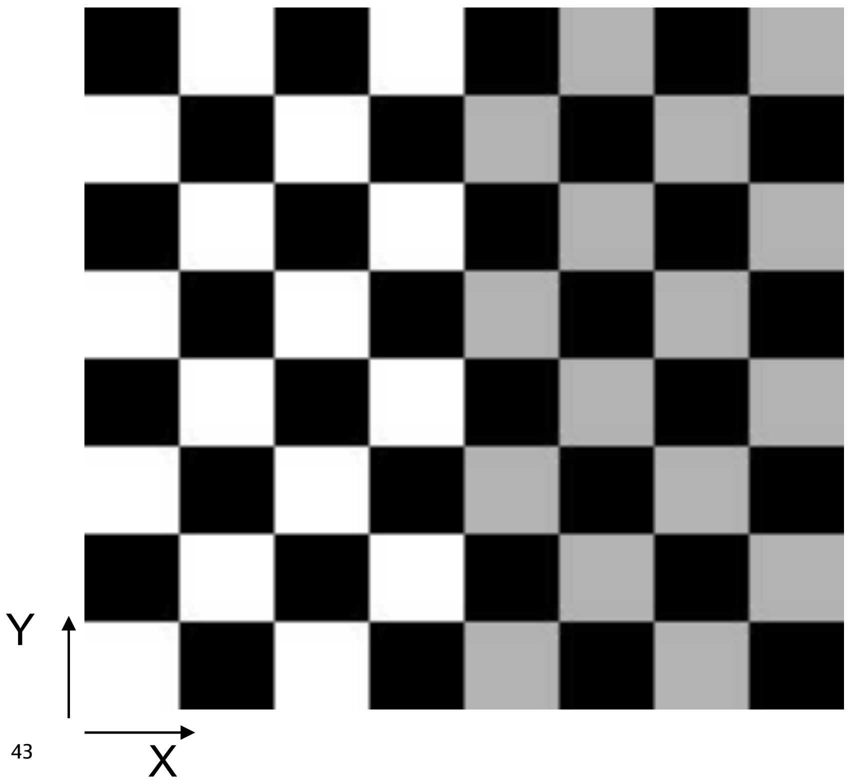
- Convert to cylindrical coordinates

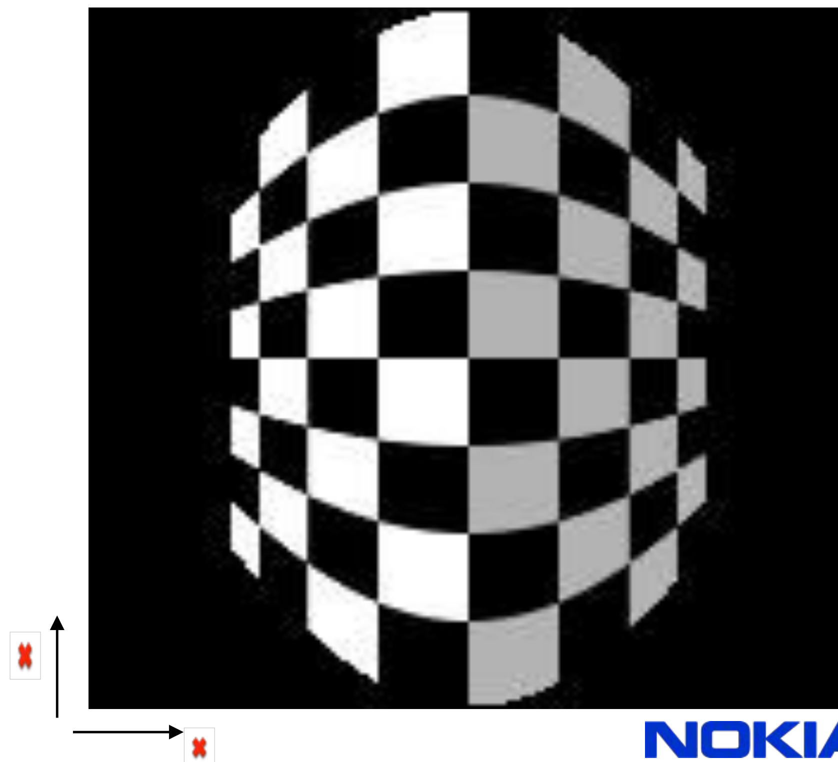
$$(sin\theta, h, cos\theta) = (\hat{x}, \hat{y}, \hat{z})$$

- Convert to cylindrical image coordinates

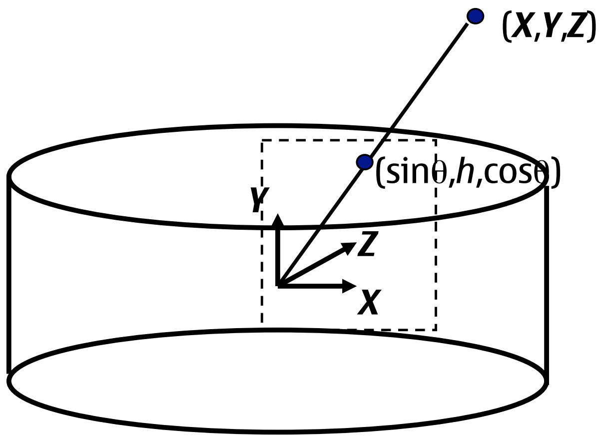
$$(\tilde{x}, \tilde{y}) = (f\theta, fh) + (\tilde{x}_c, \tilde{y}_c)$$

Cylindrical projection





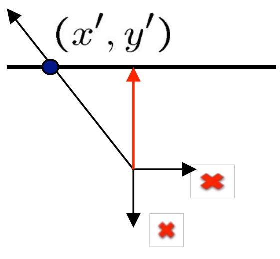
Inverse cylindrical projection



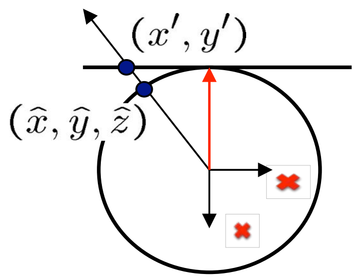
$$
\begin{aligned}
\theta &= (x_{cyl} - x_c)/f \\
h &= (y_{cyl} - y_c)/f \\
\hat{x} &= \sin\theta \\
\hat{y} &= h \\
\hat{z} &= \cos\theta
\end{aligned}
$$

NOKIA

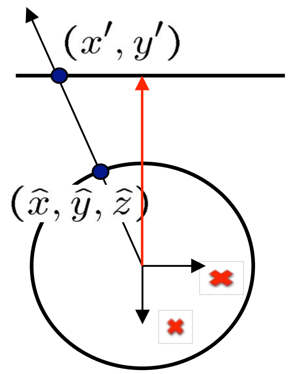
Focal length



top-down view



Focal length



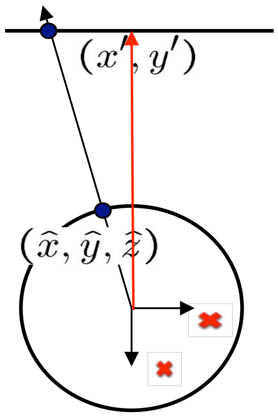


Image 384x300

f = 180 (pixels)

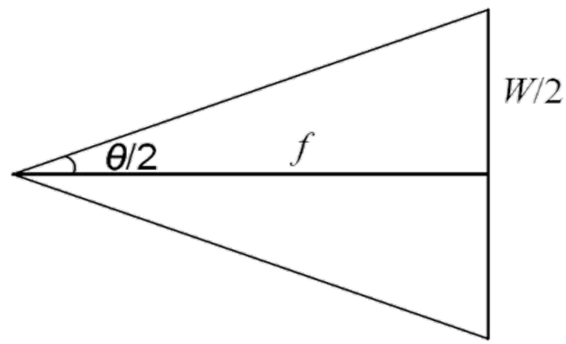
f = 280

f = 380

NOKIA

Focal length is (highly!) camera dependant

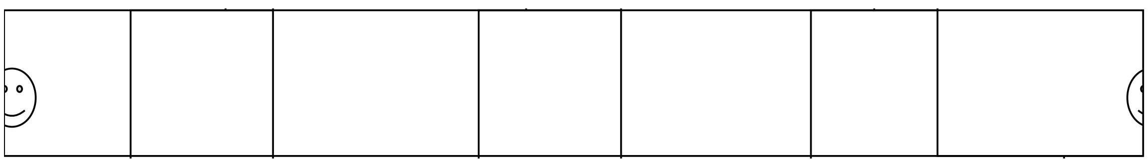
- Can get a rough estimate by measuring FOV:



- Can use the EXIF data tag
  - might not give the right thing
- Can use several images together
  - find *f* that makes them match
- Etc.

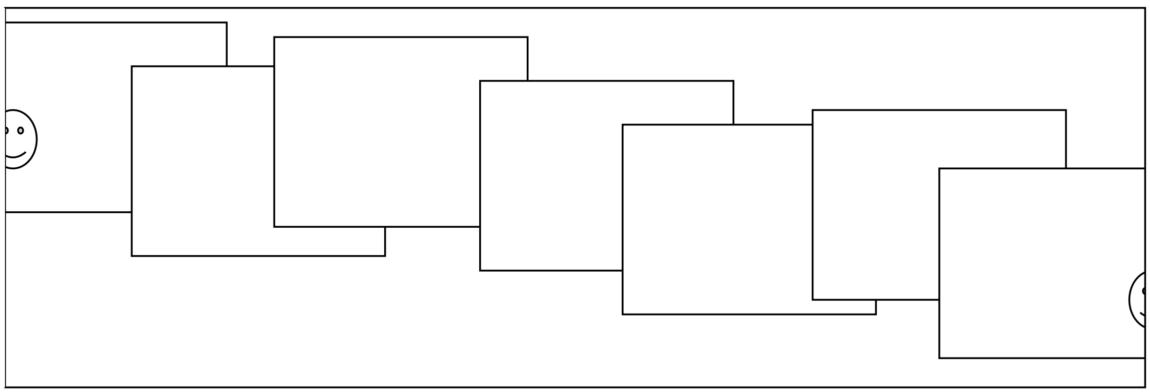
NOKIA

Assembling the panorama



Stitch pairs together, blend, then crop

Problem: Drift



Vertical Error accumulation

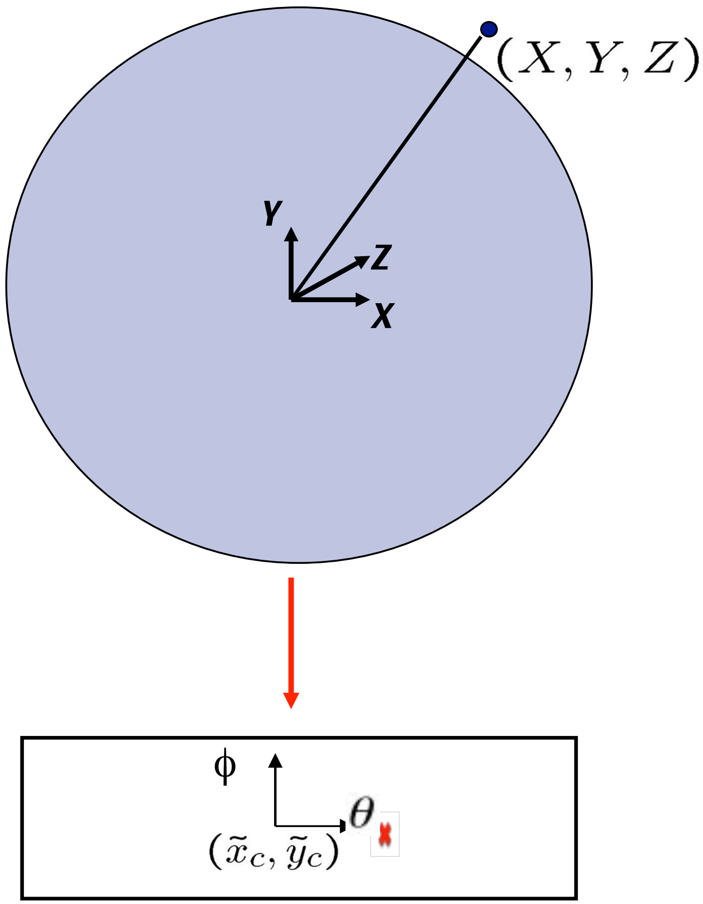
- small (vertical) errors accumulate over time
- apply correction so that sum = 0 (for 360° pan.)

Horizontal Error accumulation

- can reuse first/last image to find the right panorama radius

NOKIA

Spherical projection



unwrapped sphere

- Map 3D point (X,Y,Z) onto sphere

$$(\hat{x},\hat{y},\hat{z}) = \frac{1}{\sqrt{X^2+Y^2+Z^2}}(X,Y,Z)$$

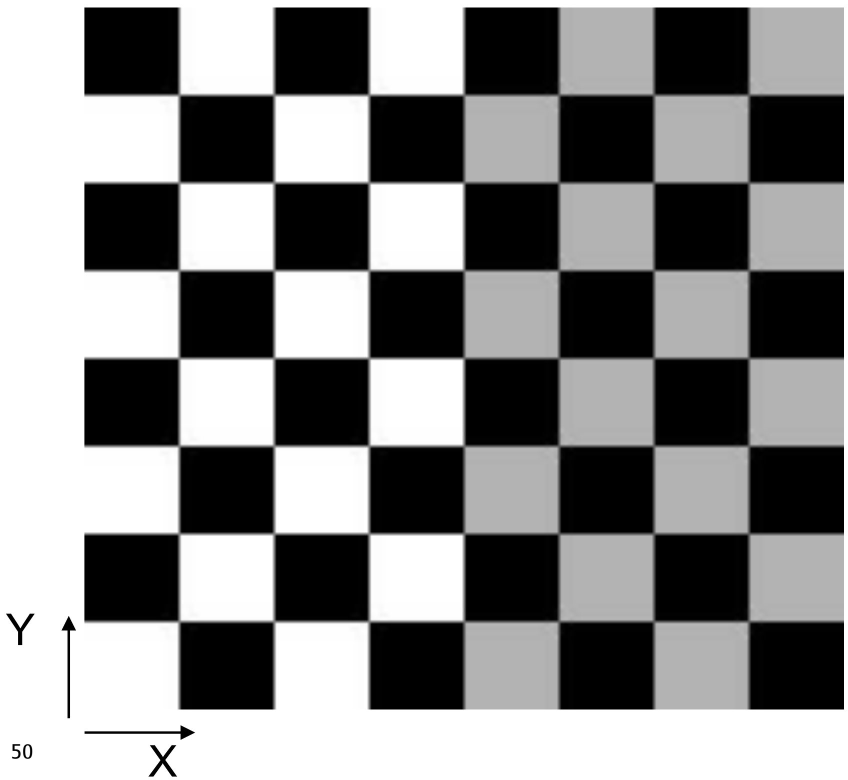
- Convert to spherical coordinates

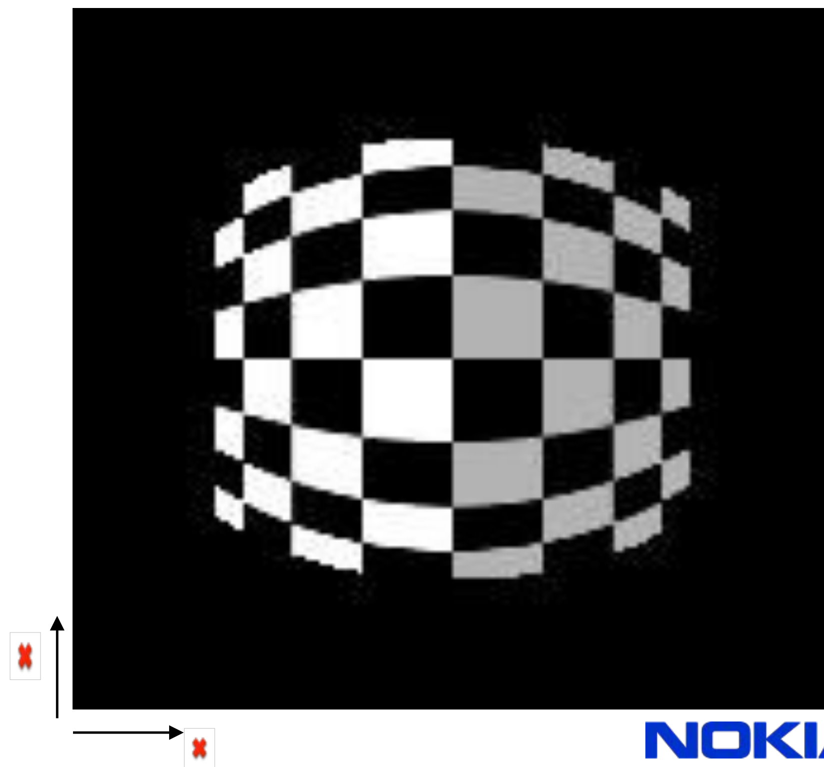
$$(\sin\theta \cos\phi, \sin\phi, \cos\theta \cos\phi) = (\hat{x},\hat{y},\hat{z})$$

- Convert to spherical image coordinates

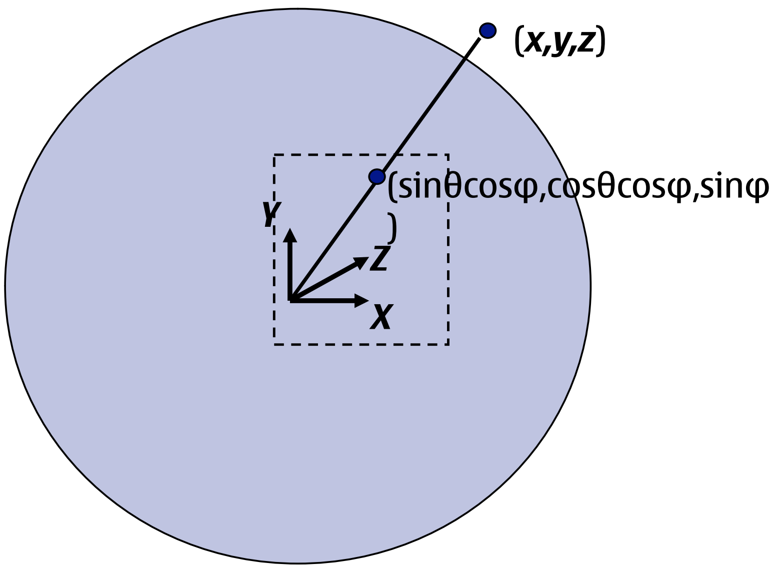
$$(\tilde{x},\tilde{y}) = (f\theta, f\phi) + (\tilde{x}_c,\tilde{y}_c)$$

Spherical Projection





Inverse Spherical projection



$$\theta = (x_{sph} - x_c)/f$$

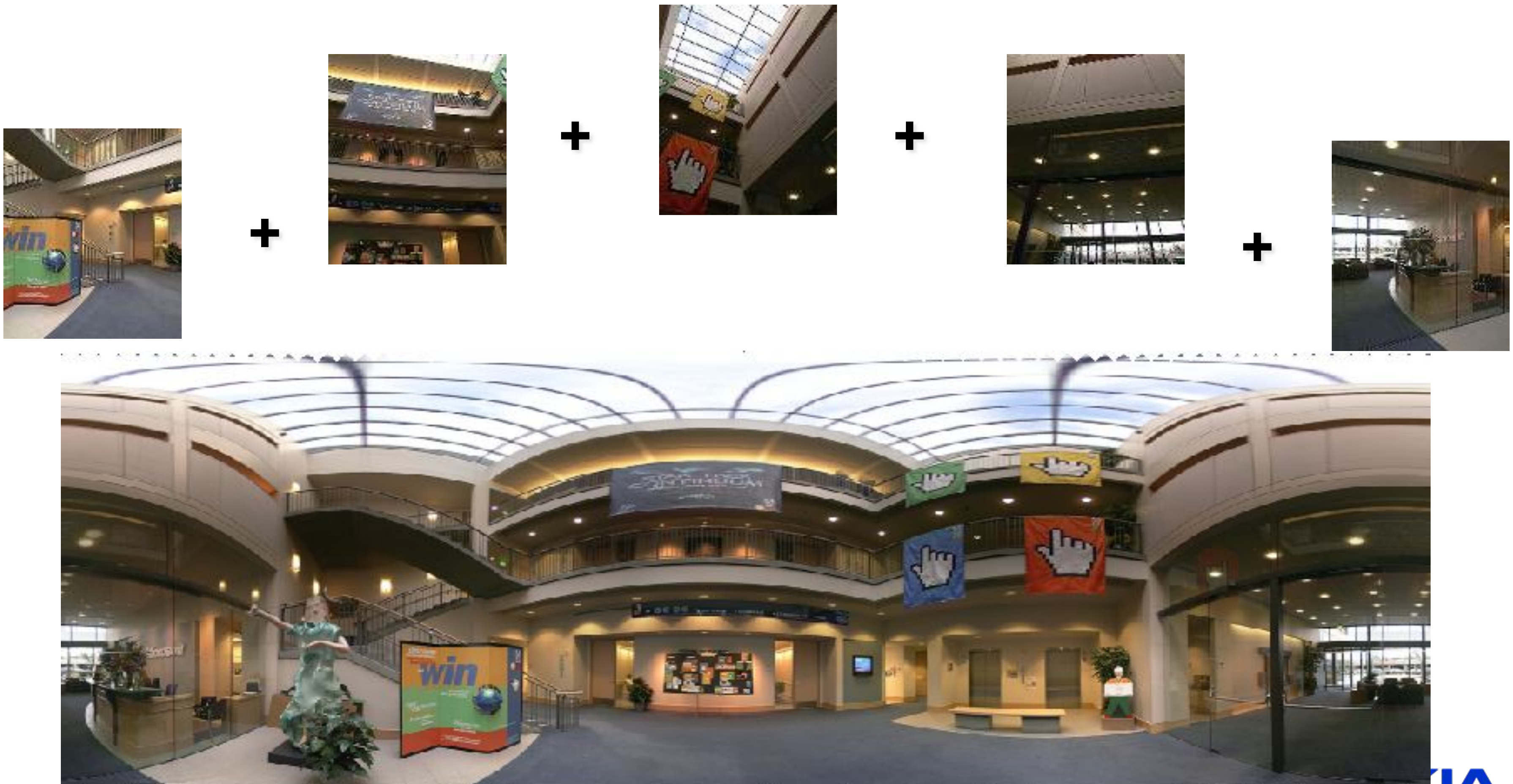
$$\varphi = (y_{sph} - y_c)/f$$

$$\hat{x} = \sin\theta \cos\varphi$$

$$\hat{y} = \sin\varphi$$

$$\hat{z} = \cos\theta \cos\varphi$$

Full-view Panorama

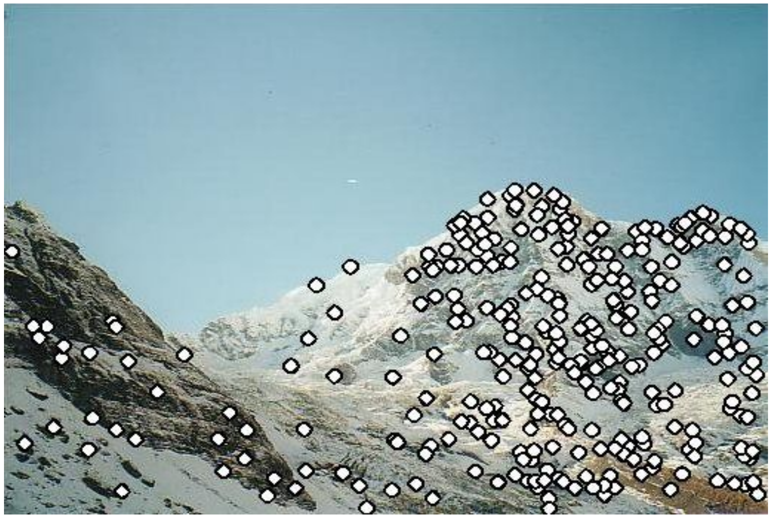


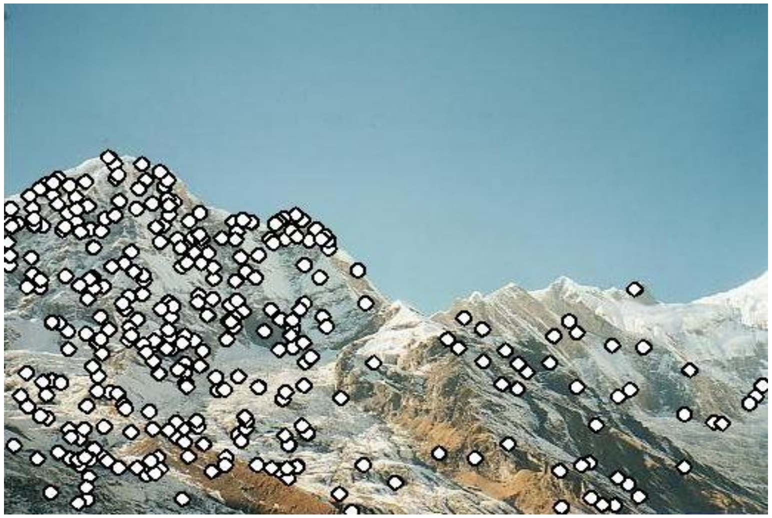
Building a Panorama

 M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003

We need to match (align) images

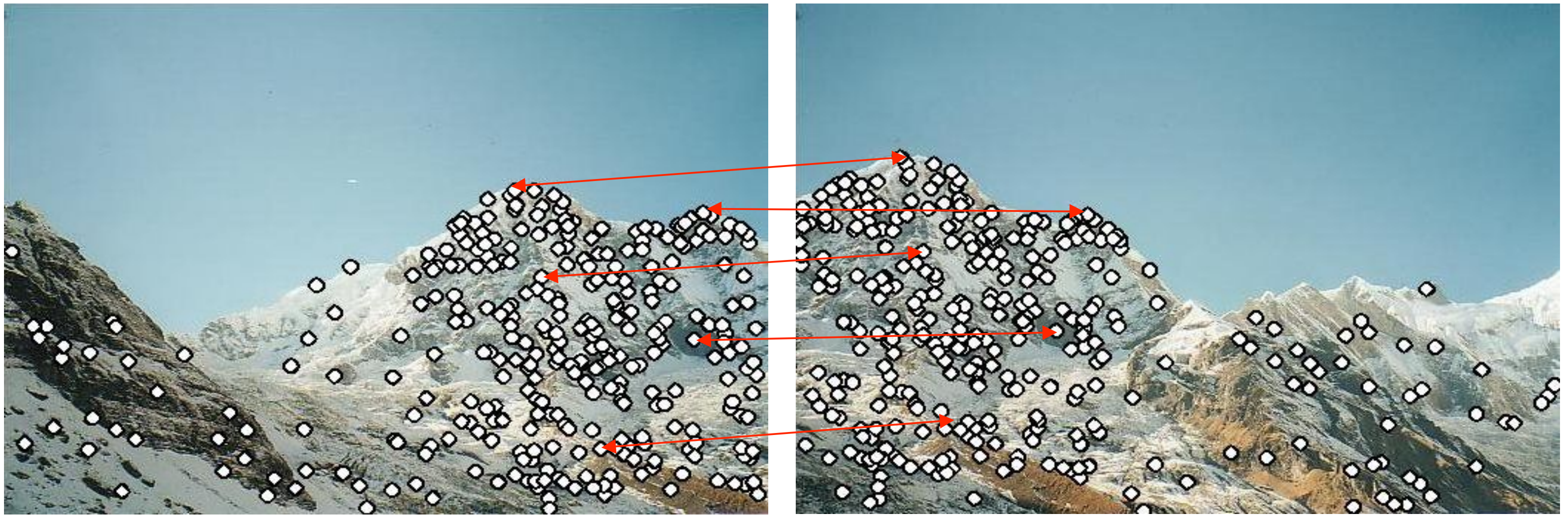
Detect feature points in both images





NOKIA

Find corresponding pairs



Use these pairs to align images

Matching with Features

Problem 1:

- Detect the *same* point *independently* in both images

no chance to match!

We need a repeatable detector

NOKIA

Matching with Features

Problem 2:

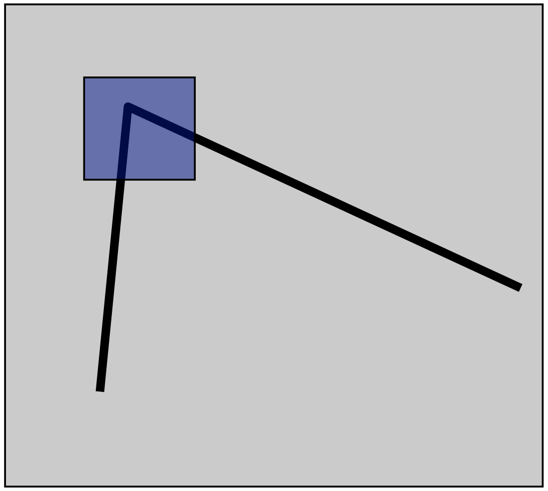
- For each point correctly recognize the corresponding one

We need a reliable and distinctive descriptor

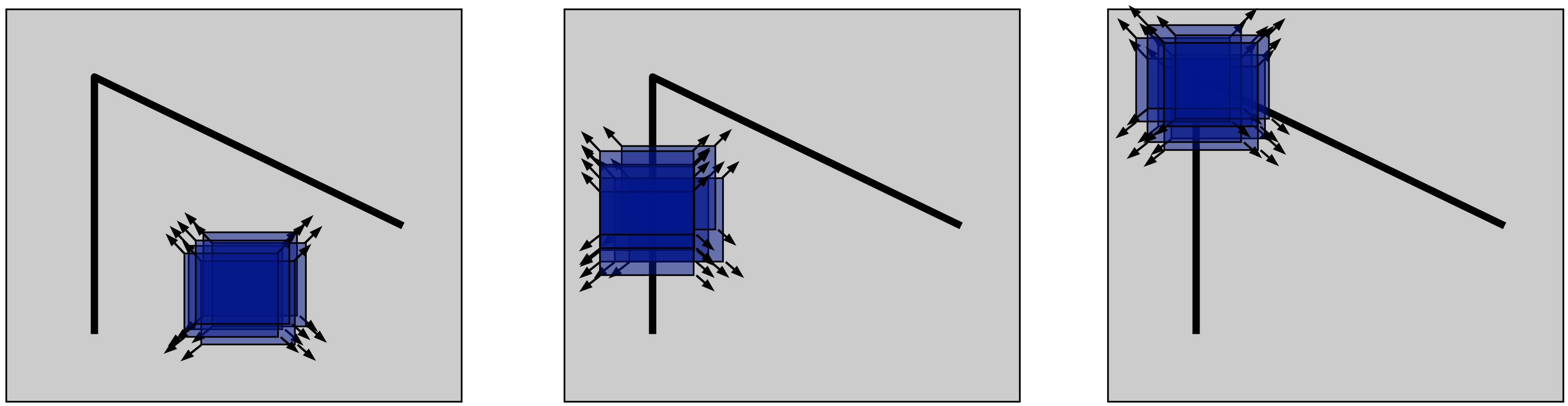
Harris Corners: The Basic Idea

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

Window-averaged change of intensity for the shift [*u*,*v*]:

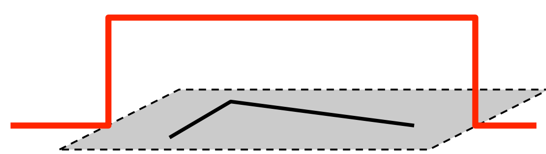
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

Window function

Shifted intensity

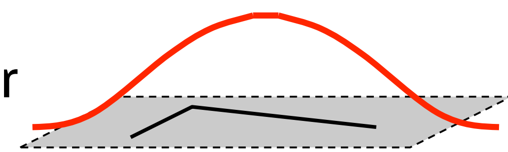
Intensity

Window function *w(x,y)* =



1 in window, 0 outside

or



Gaussian

Harris Detector: Mathematics

Expanding *E(u,v)* in a 2nd order Taylor series expansion, we have, for small shifts $[u,v]$, a *bilinear* approximation:

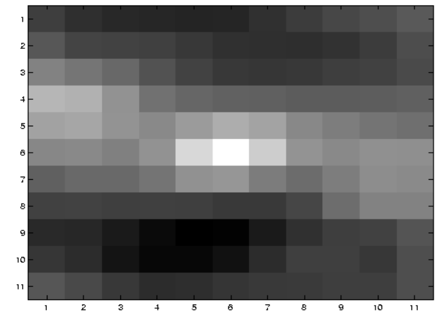
$$E(u,v) \cong [u,v] \; M \; \begin{bmatrix} u \\ v \end{bmatrix}$$

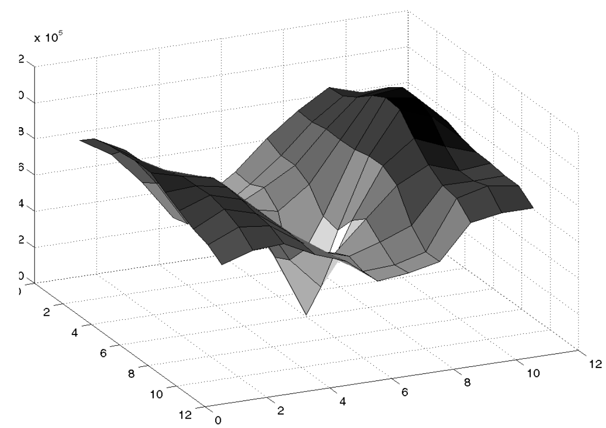
where *M* is a 2×2 matrix computed from image derivatives:

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

NOKIA

Eigenvalues λ_1, λ_2 of M at different locations

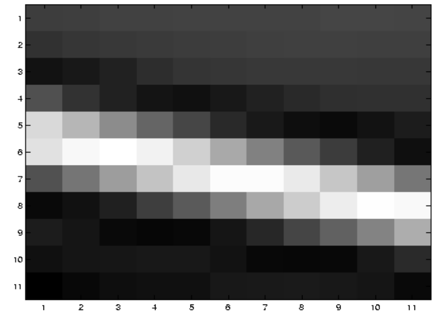


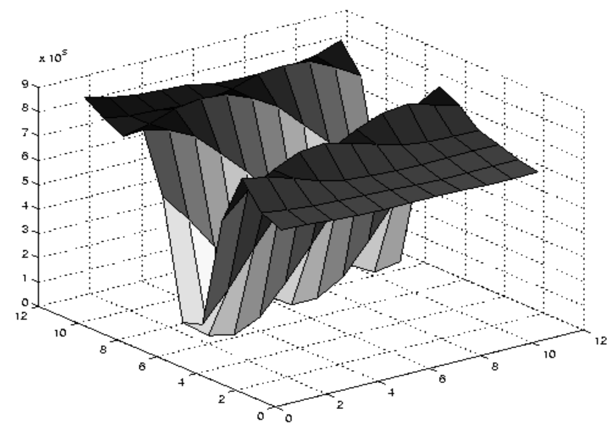


λ_1 and λ_2 are large

NOKIA

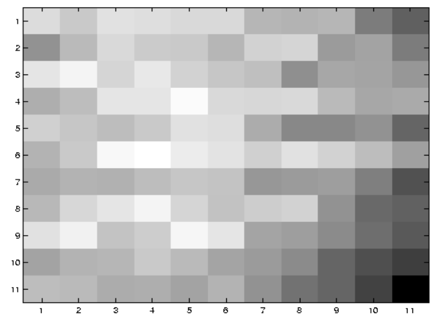
Eigenvalues λ_1, λ_2 of M at different locations

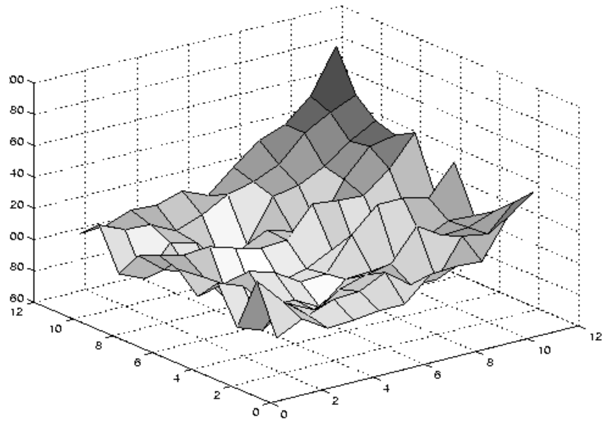




large λ_1, small λ_2

Eigenvalues λ_1, λ_2 of M at different locations



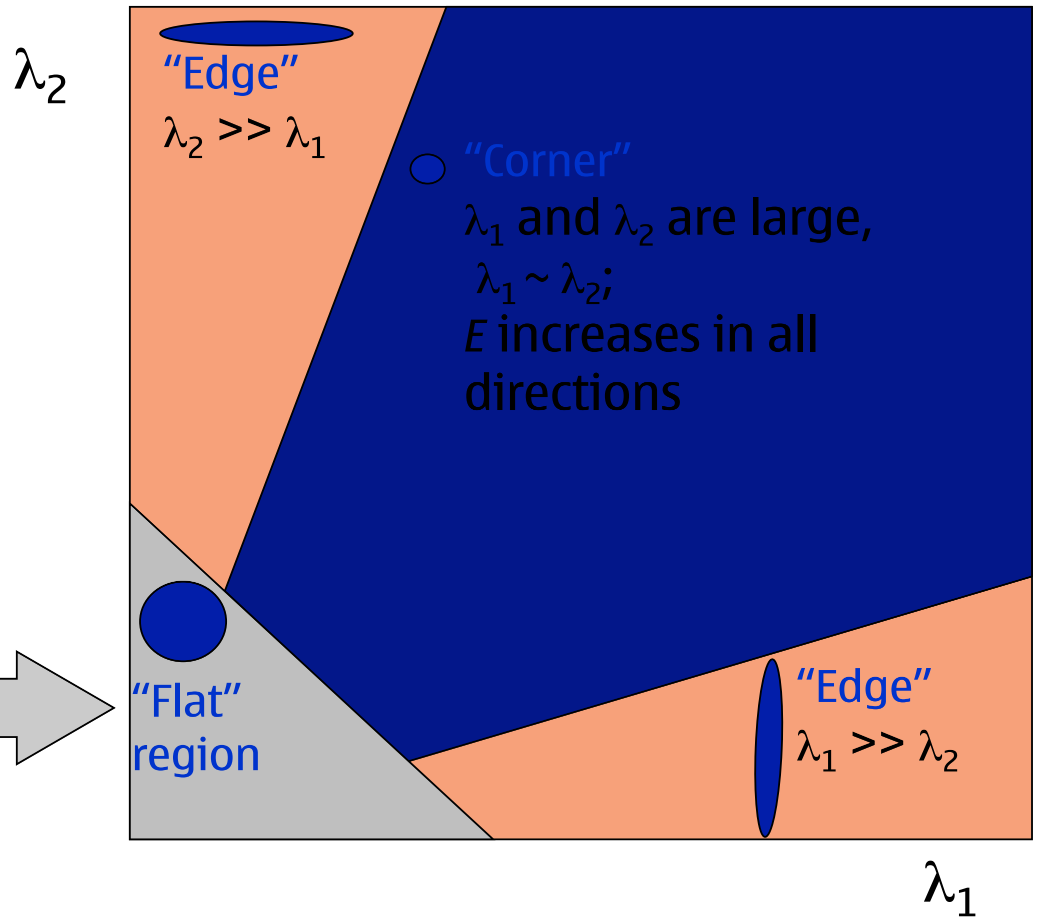


small λ_1, small λ_2

Harris Detector: Mathematics

Classification of image points using eigenvalues of *M*:

λ_1 and λ_2 are small;
E is almost constant in all directions



Harris Detector: Mathematics

Measure of corner response:

$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

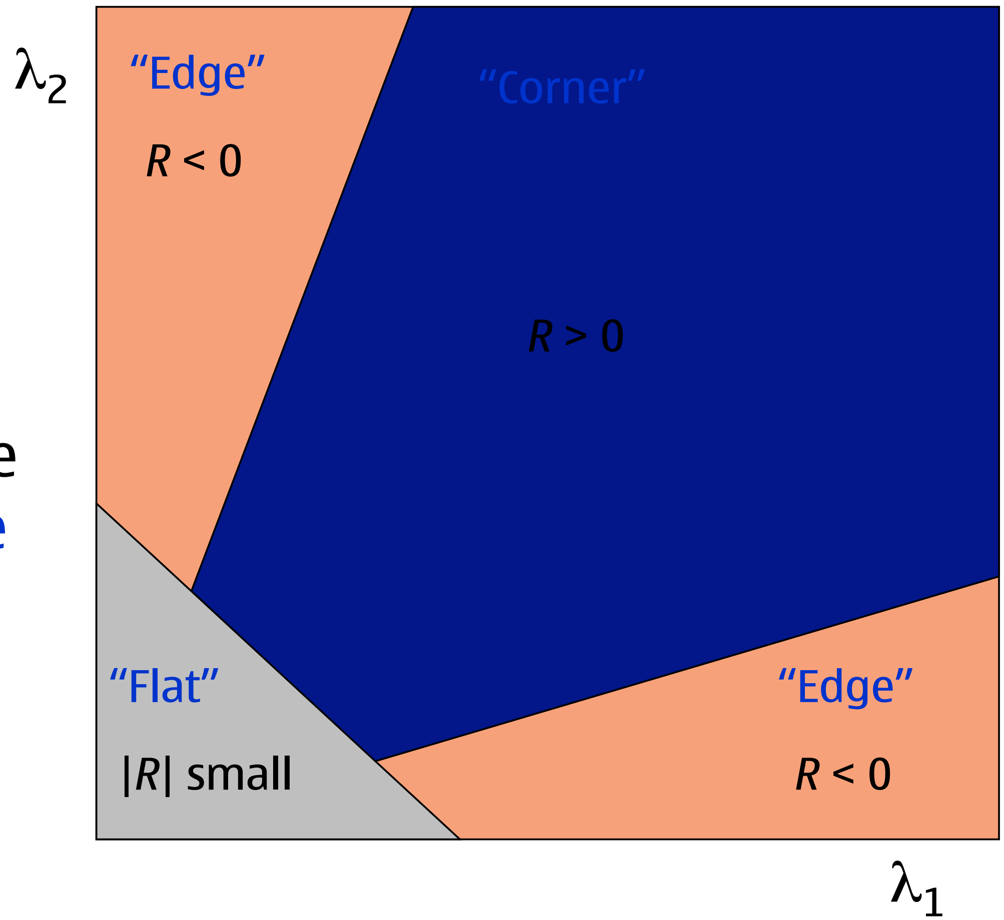
$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

(k – empirical constant, k = 0.04-0.06)

NOKIA

Harris Detector: Mathematics

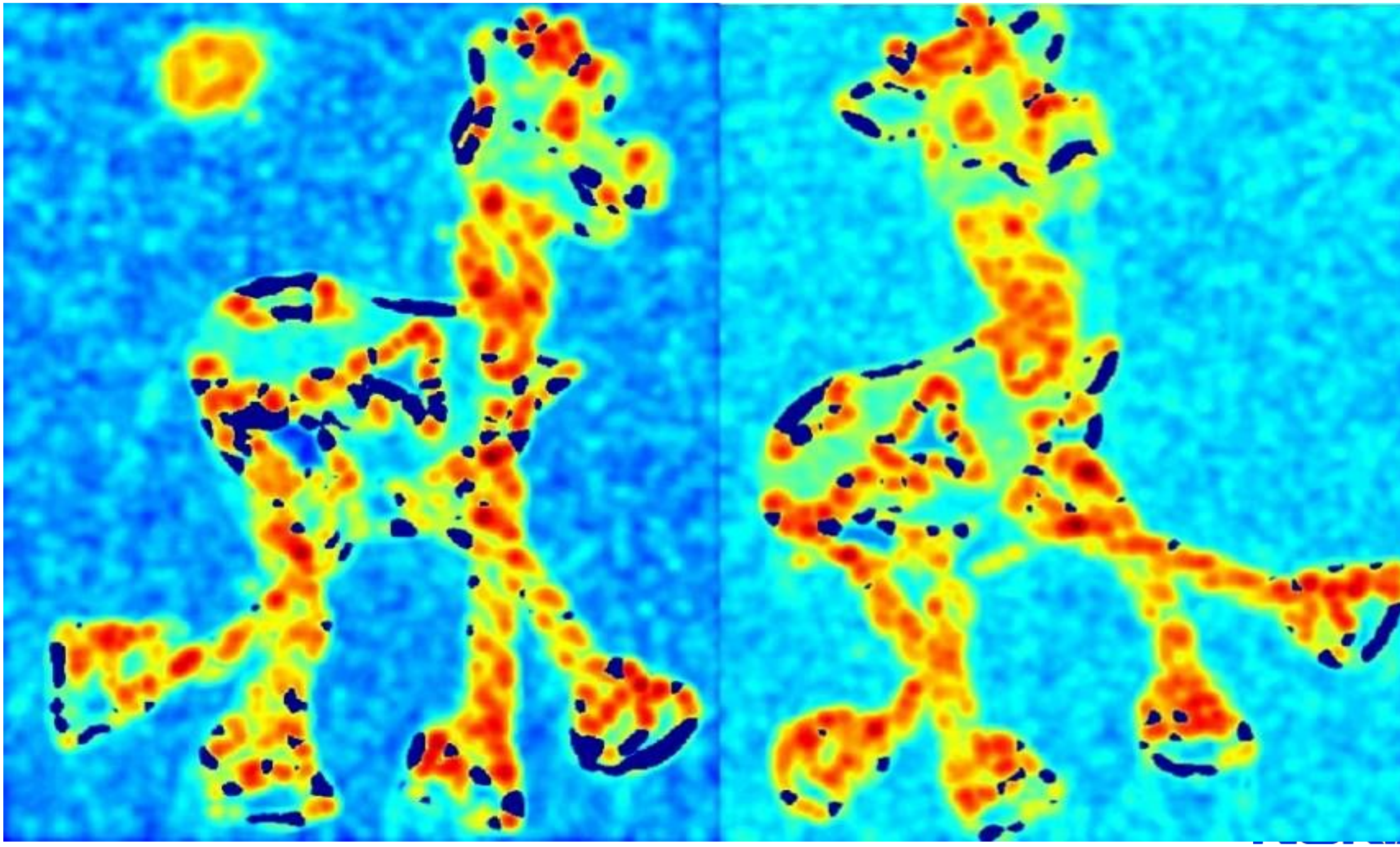
- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region



Harris Detector: Workflow

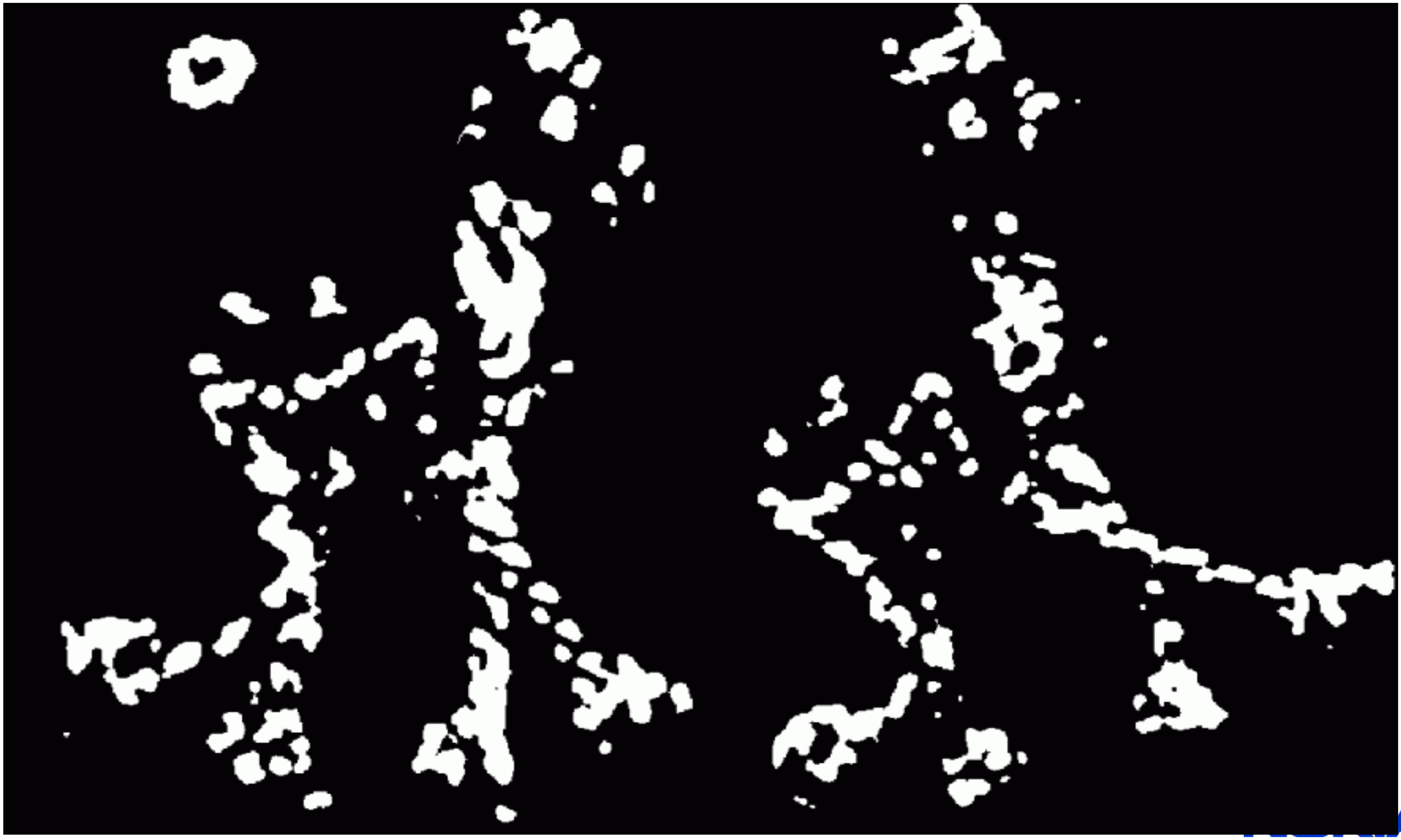
Harris Detector: Workflow

Compute corner response R



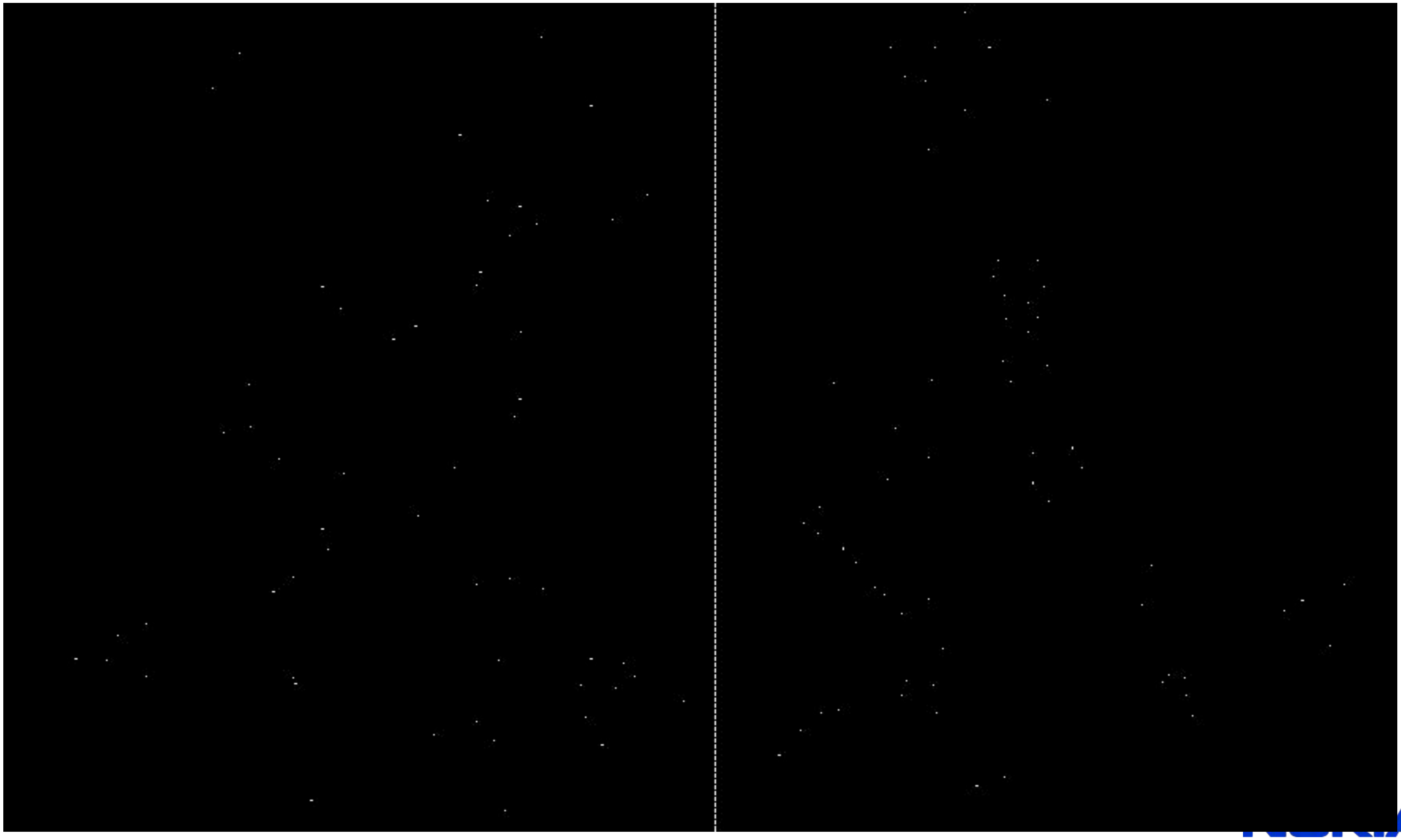
Harris Detector: Workflow

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: Summary

Average intensity change in direction $[u,v]$ can be expressed as a bilinear form:

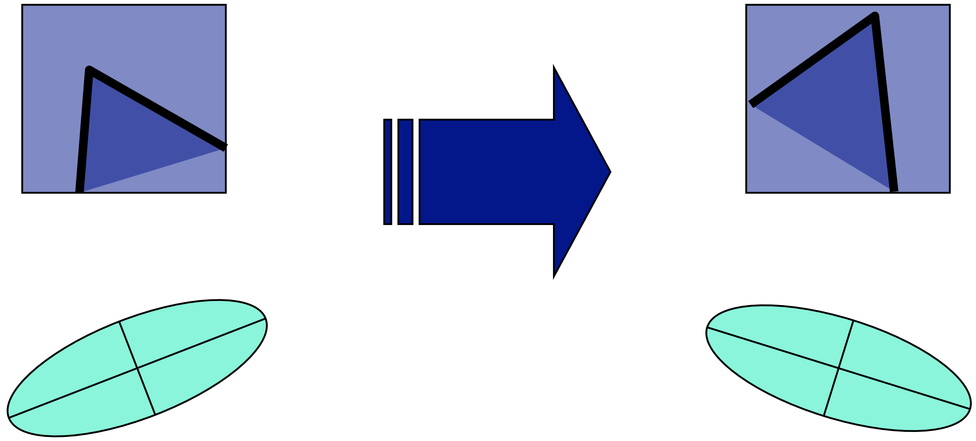
$$E(u,v) \cong [u,v] \; M \; \begin{bmatrix} u \\ v \end{bmatrix}$$

Describe a point in terms of eigenvalues of M: *measure of corner response*

$$R = \lambda_1 \lambda_2 - k\left(\lambda_1 + \lambda_2\right)^2$$

A good (corner) point should have a *large intensity change* in *all directions*, i.e., R should be large positive

Harris Detector: Invariant to rotation



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

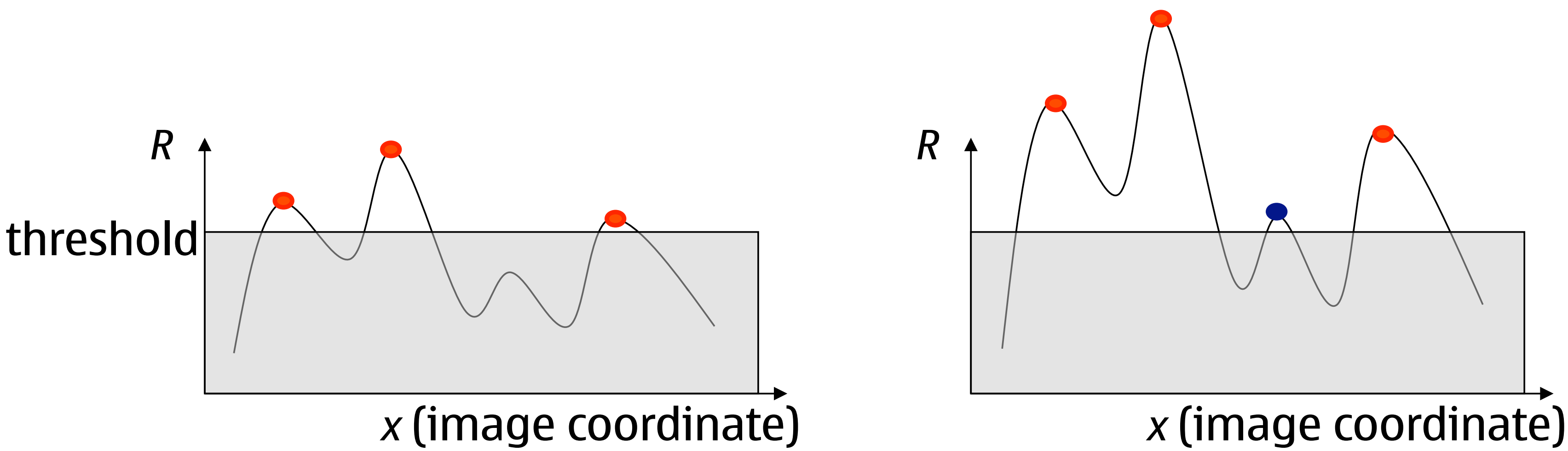
Corner response R is invariant to image rotation

Harris Detector: ~Invariant to intensite change

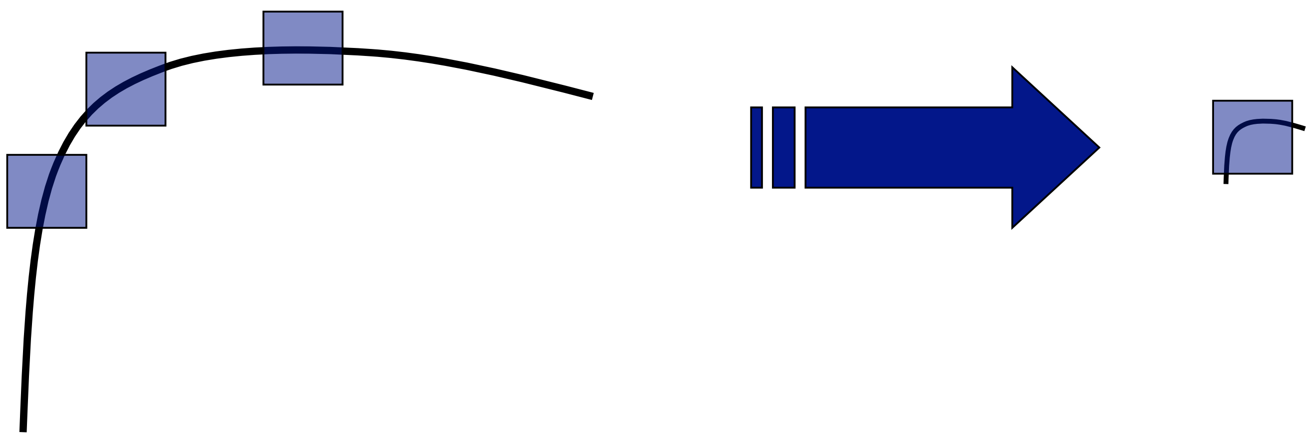
Partial invariance

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

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



Harris Detector: ***Not*** invariant to image scale!



All points will be classified as edges

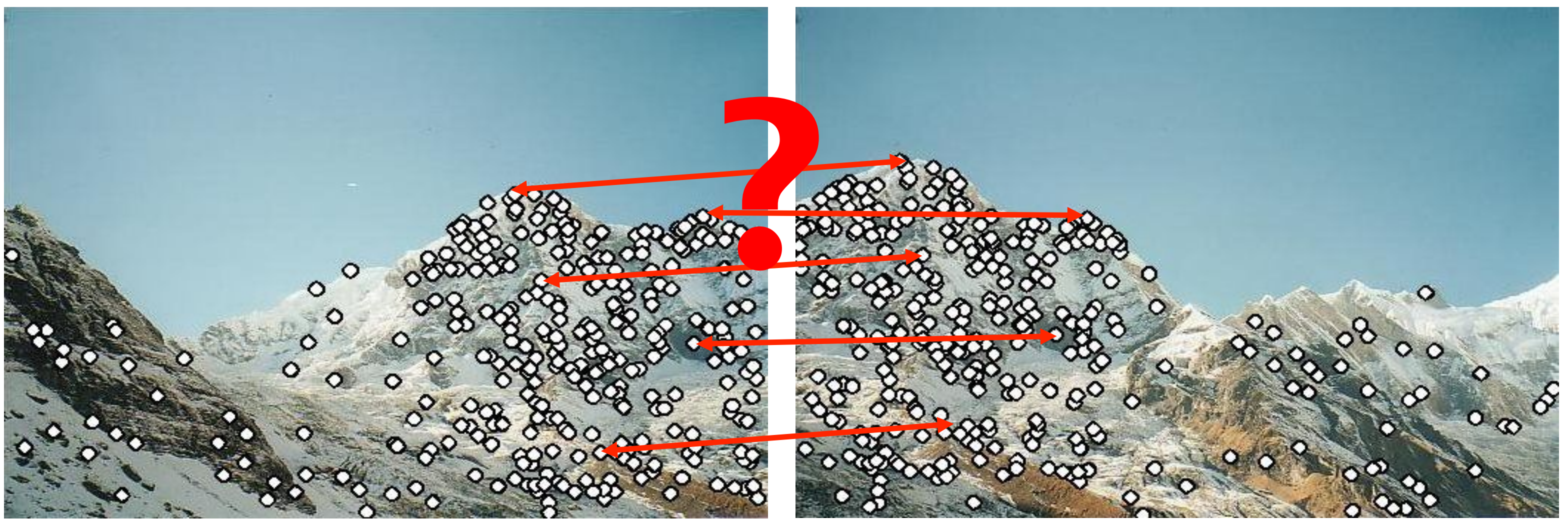
Corner !

Point Descriptors

We know how to detect points

Next question:

How to match them?



Point descriptor should be:

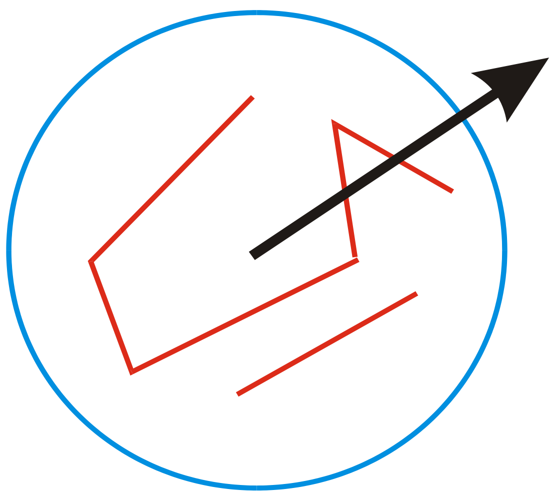
1. Invariant
2. Distinctive

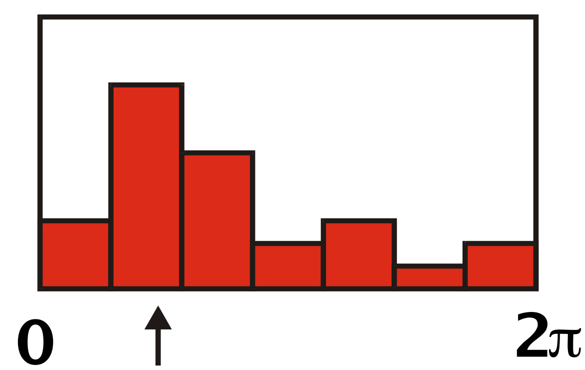
NOKIA

SIFT – Scale Invariant Feature Transform

Descriptor overview:

- Determine scale (by maximizing DoG in scale and in space), local orientation as the dominant gradient direction.
  Use this scale and orientation to make all further computations invariant to scale and rotation.

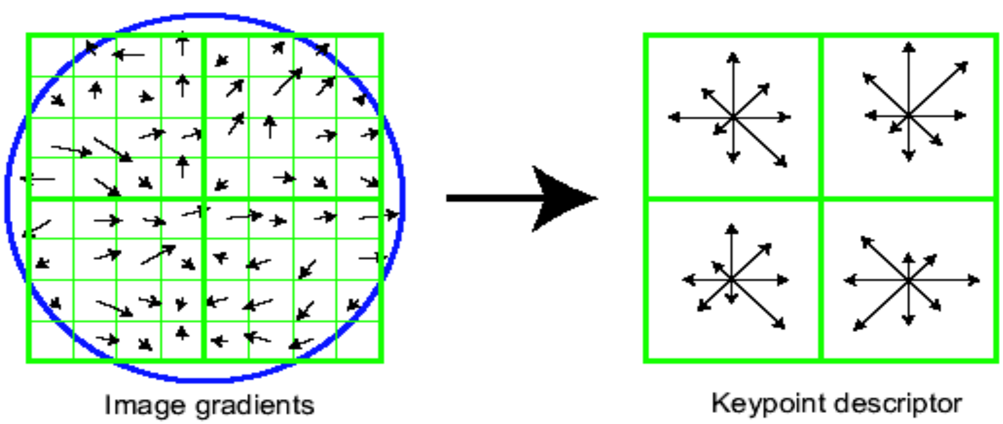




SIFT – Scale Invariant Feature Transform

Descriptor overview:

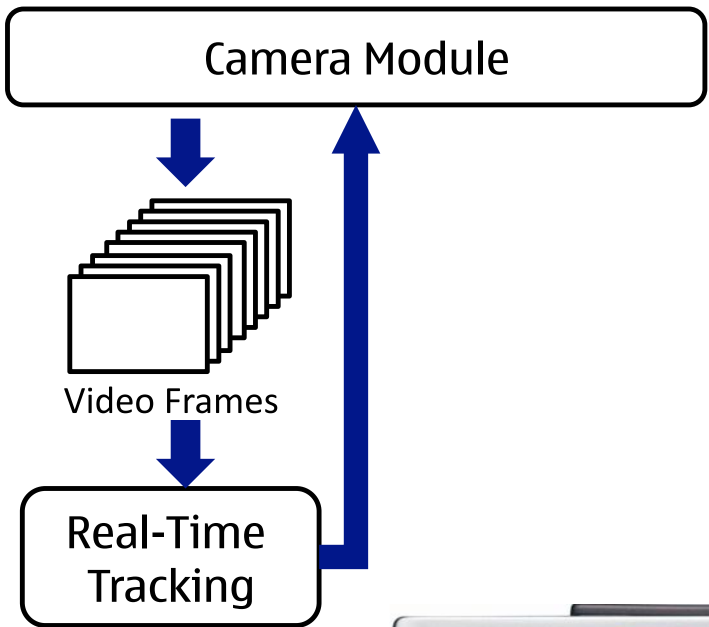
- Determine scale (by maximizing DoG in scale and in space), local orientation as the dominant gradient direction. Use this scale and orientation to make all further computations invariant to scale and rotation.
- Compute gradient orientation histograms of several small windows (128 values for each point)
- Normalize the descriptor to make it invariant to intensity change



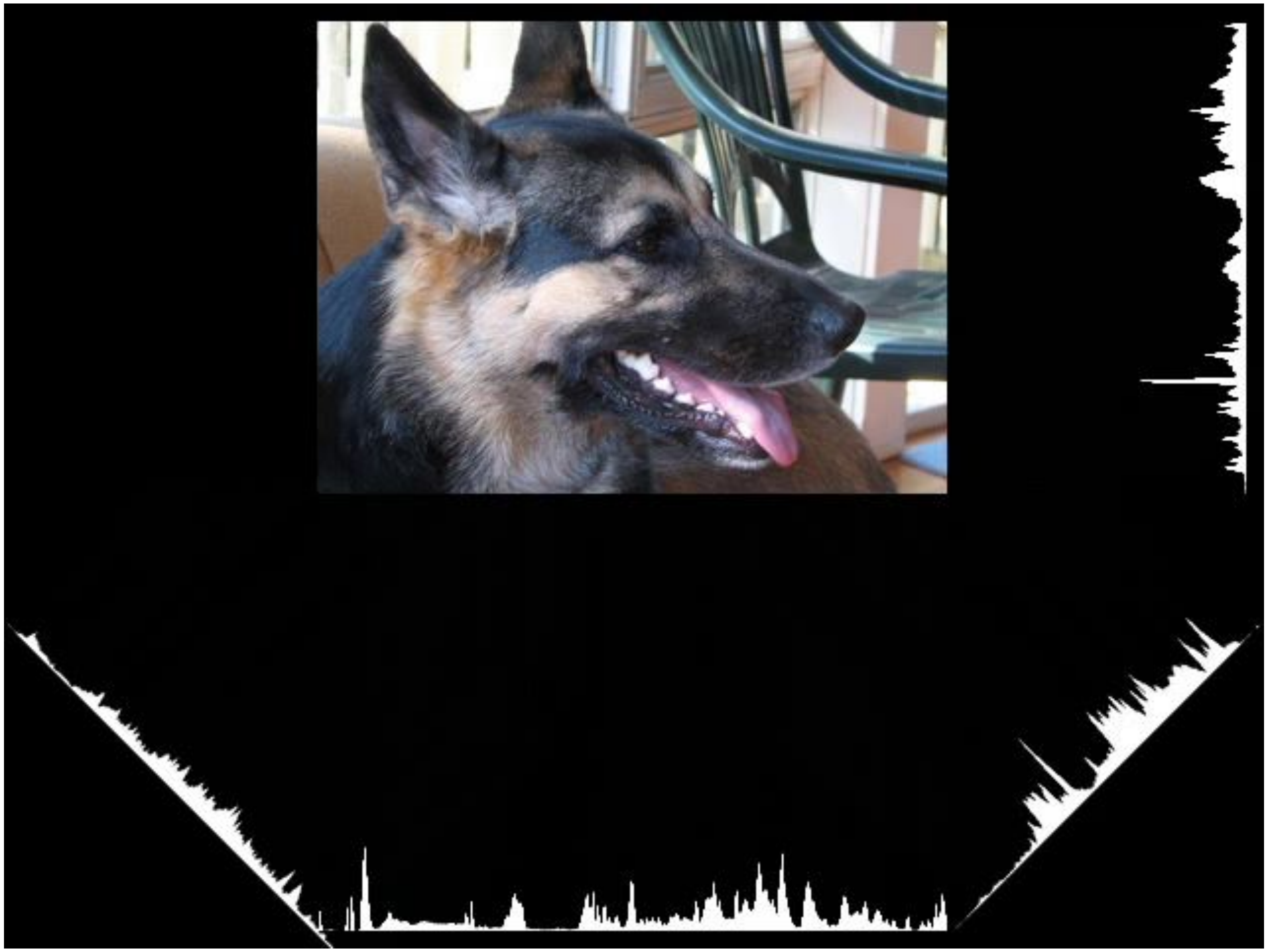
D. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints" IJCV 2004

Figure 12: The training images for two objects are shown on the left. These can be recognized in a cluttered image with extensive occlusion, shown in the middle. The results of recognition are shown on the right. A parallelogram is drawn around each recognized object showing the boundaries of the original training image under the affi ne transformation solved for during recognition. Smaller squares indicate the keypoints that were used for recognition.

Registration in practice: tracking



Viewfinder alignment for tracking

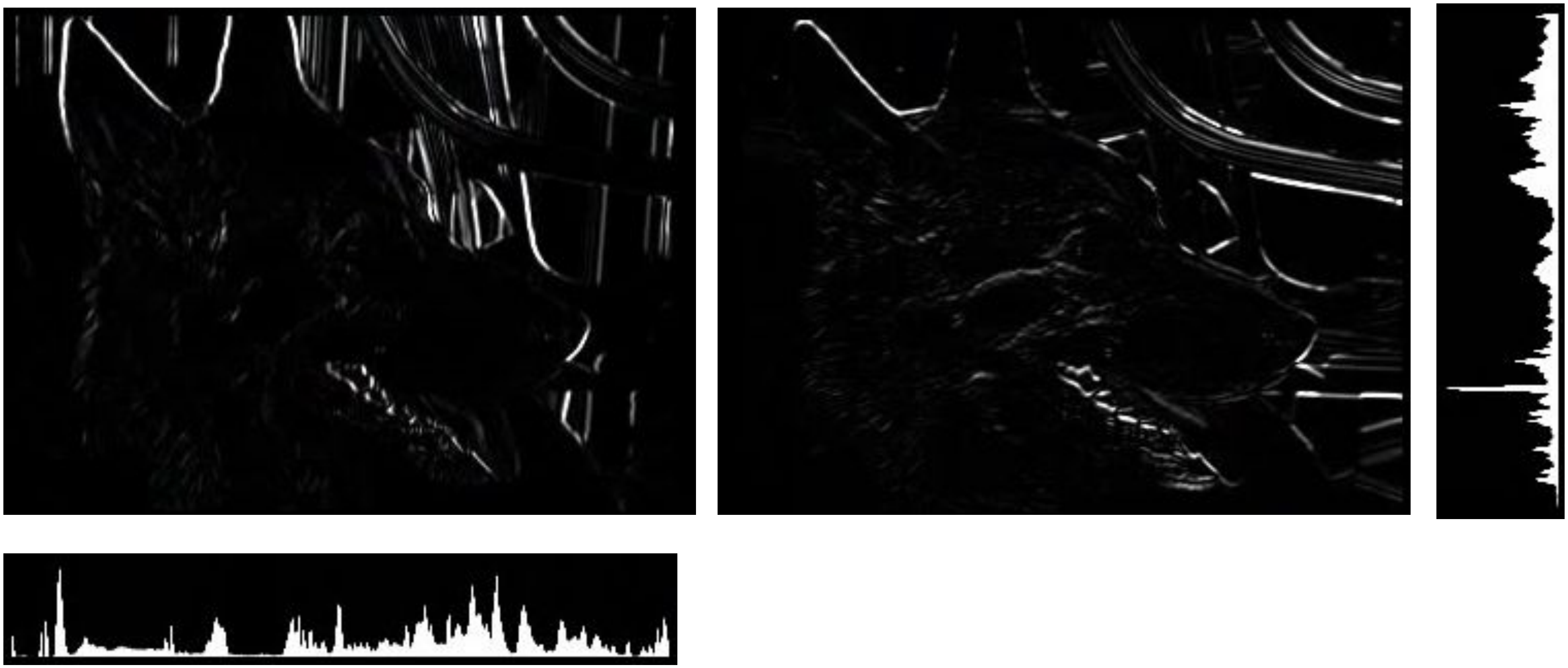


Andrew Adams, Natasha Gelfand, Kari Pulli
Viewfinder Alignment
Eurographics 2008

http://graphics.stanford.edu/papers/viewfinderalignment/

NOKIA

Project gradients along columns and rows



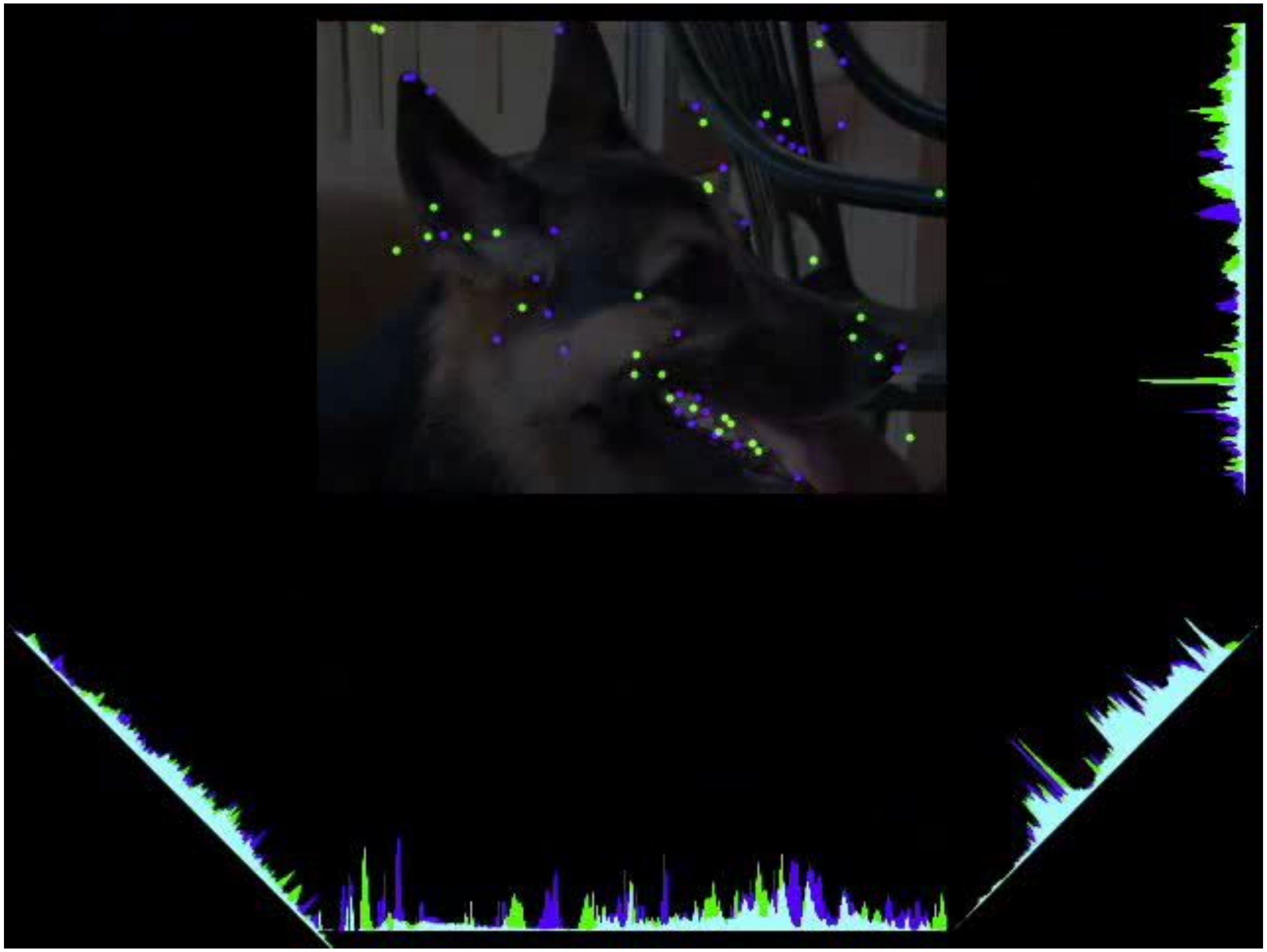
NOKIA

... diagonal gradients along diagonals ...

... and find corners

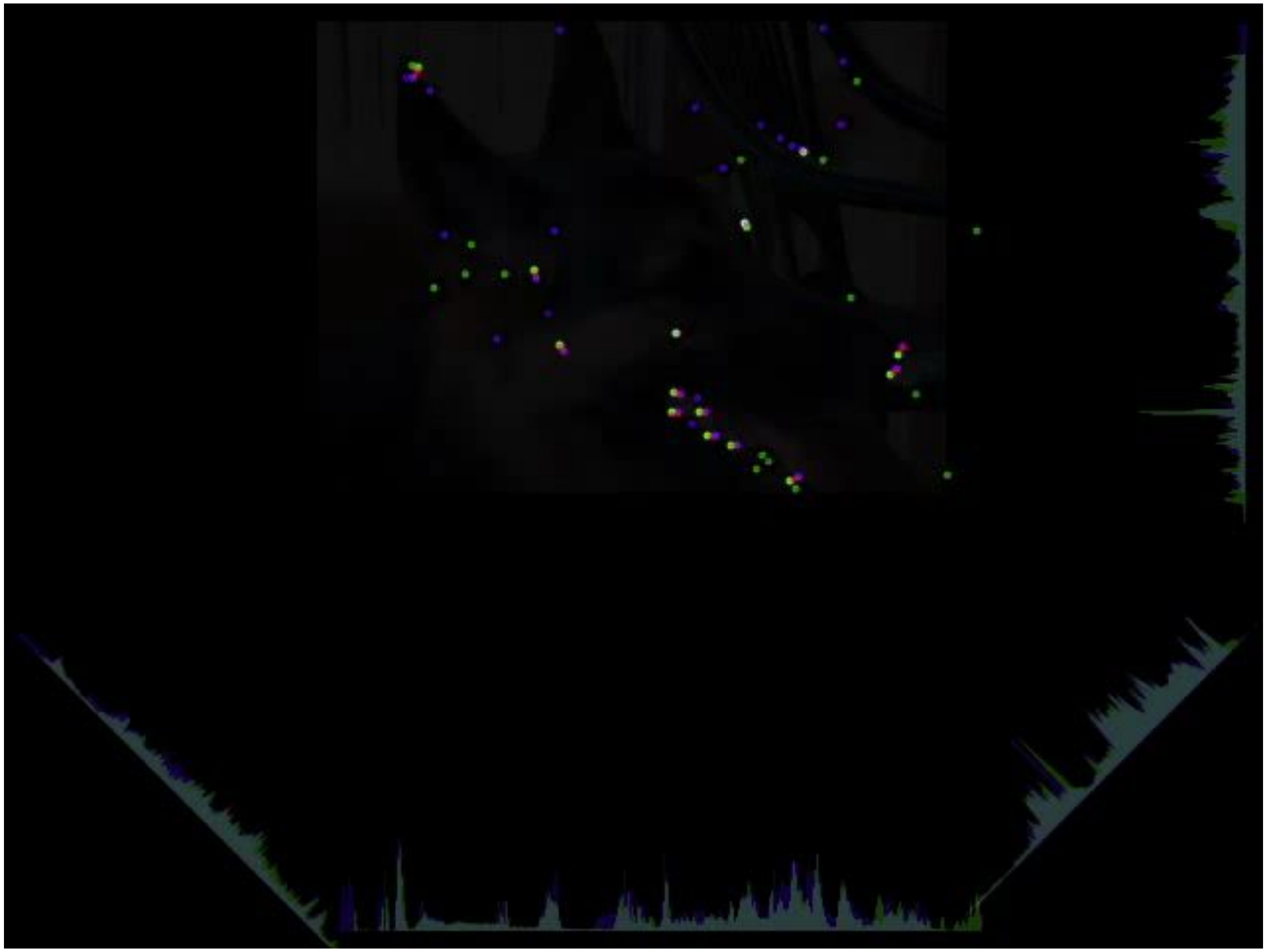
NOKIA

Overlap and match the gradient projections and determine translation

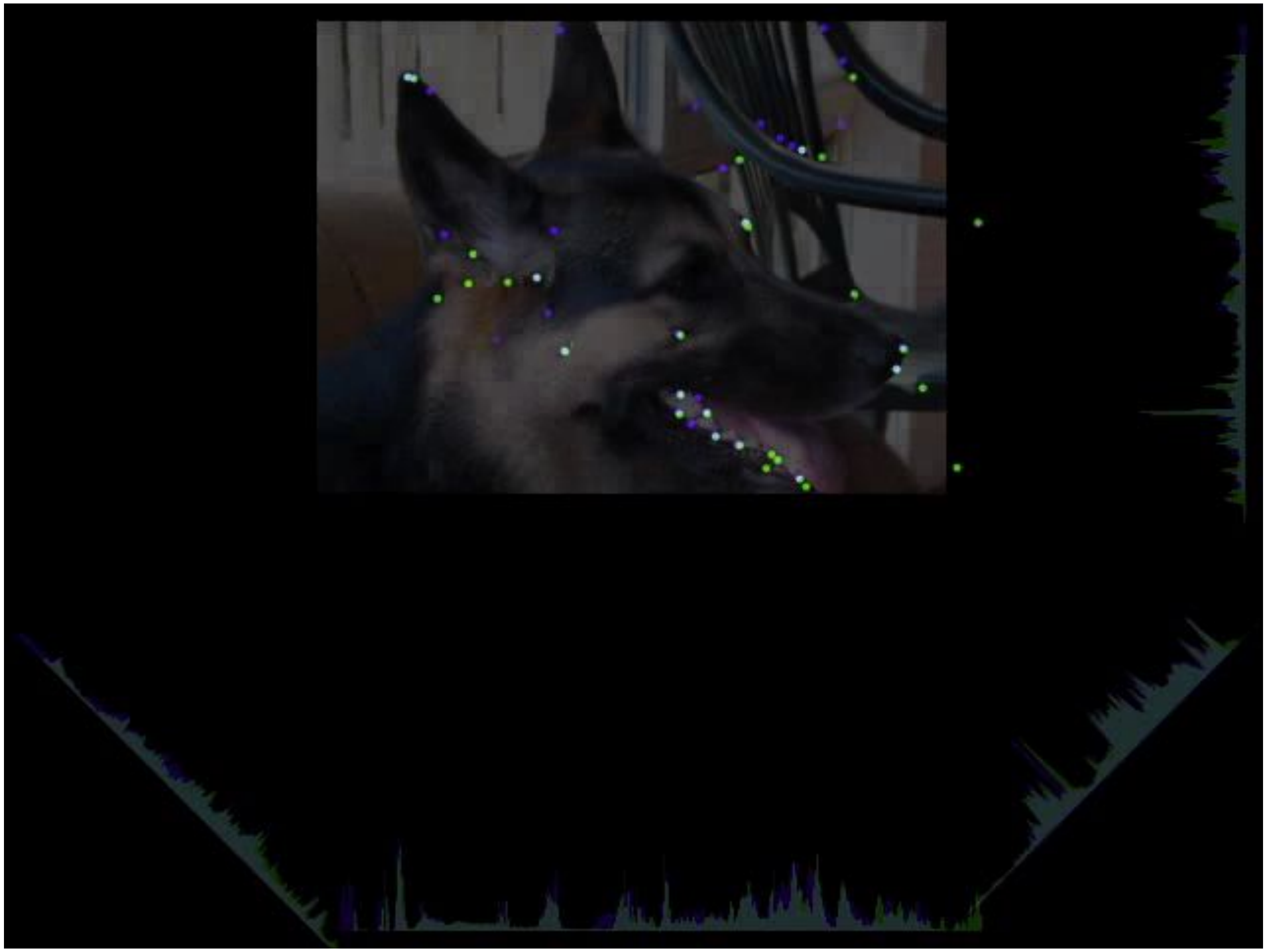


NOKIA

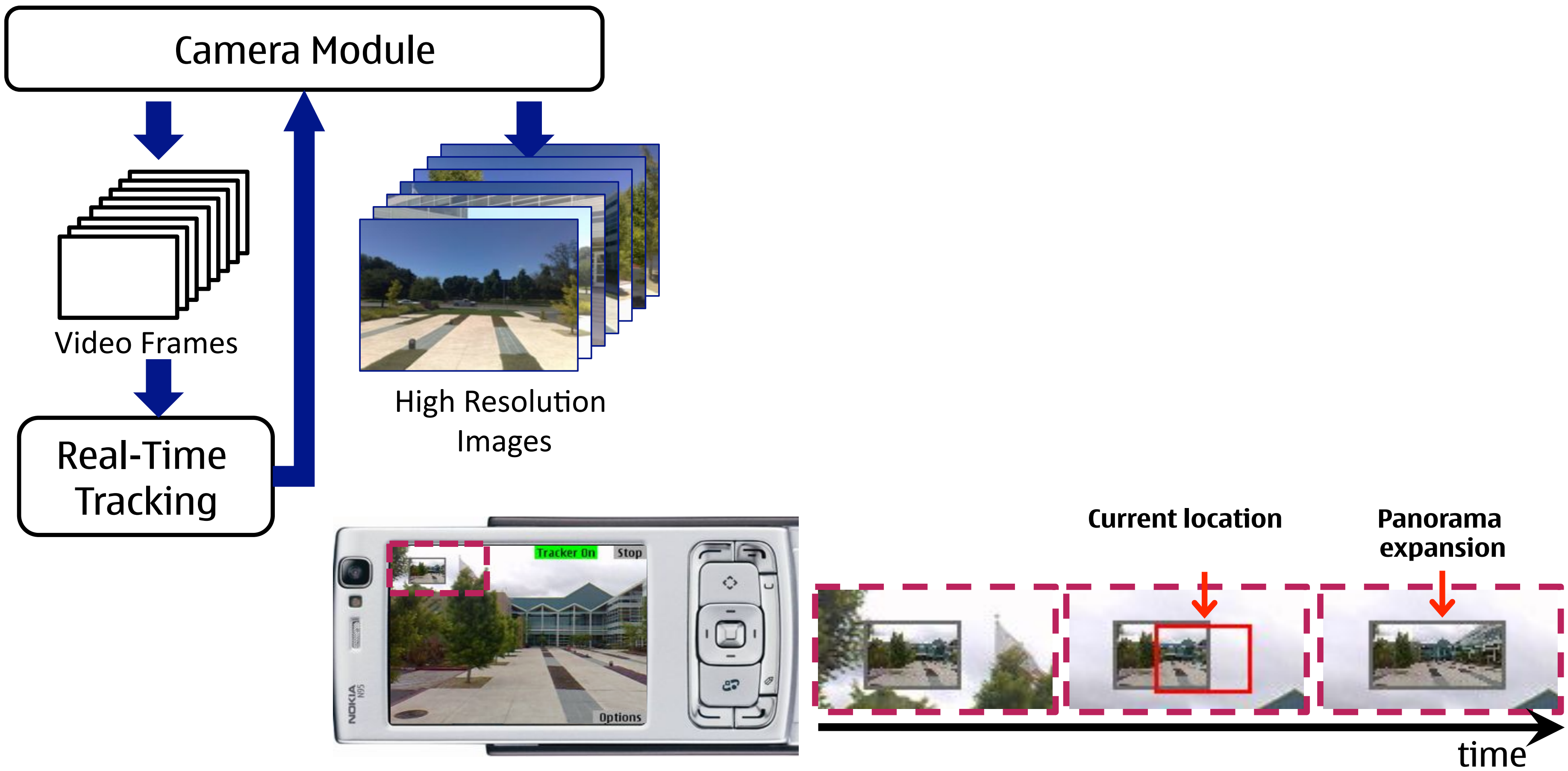
Apply the best translation to corners



Match corners, refine translation & rotation



System Overview



System Overview

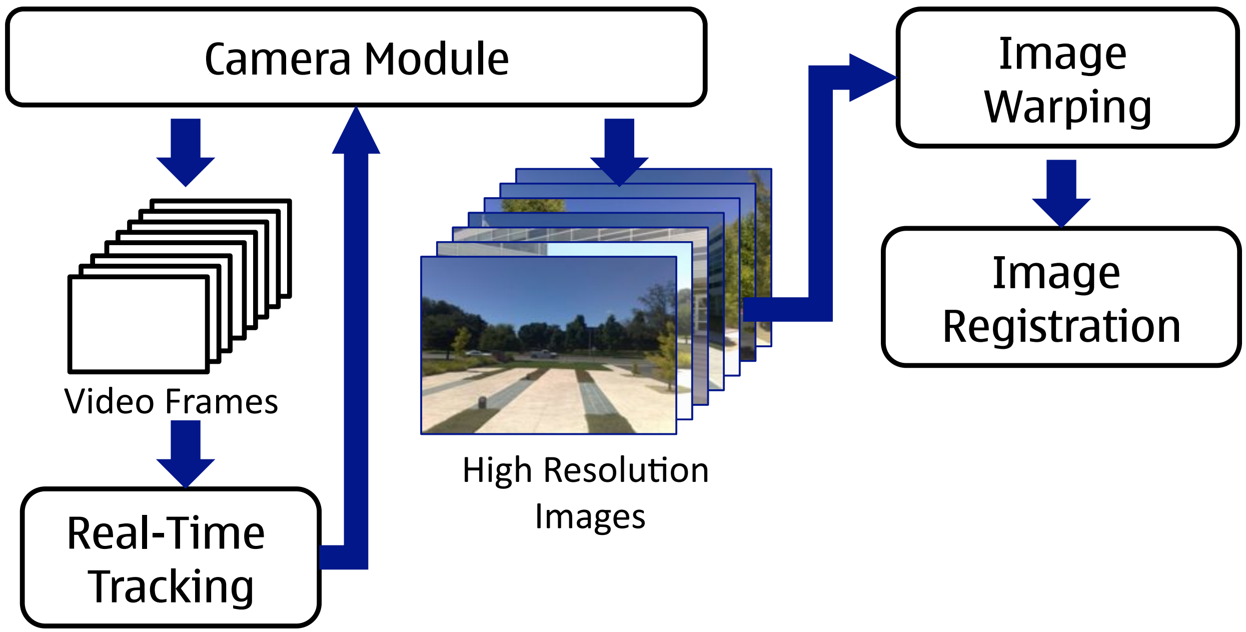


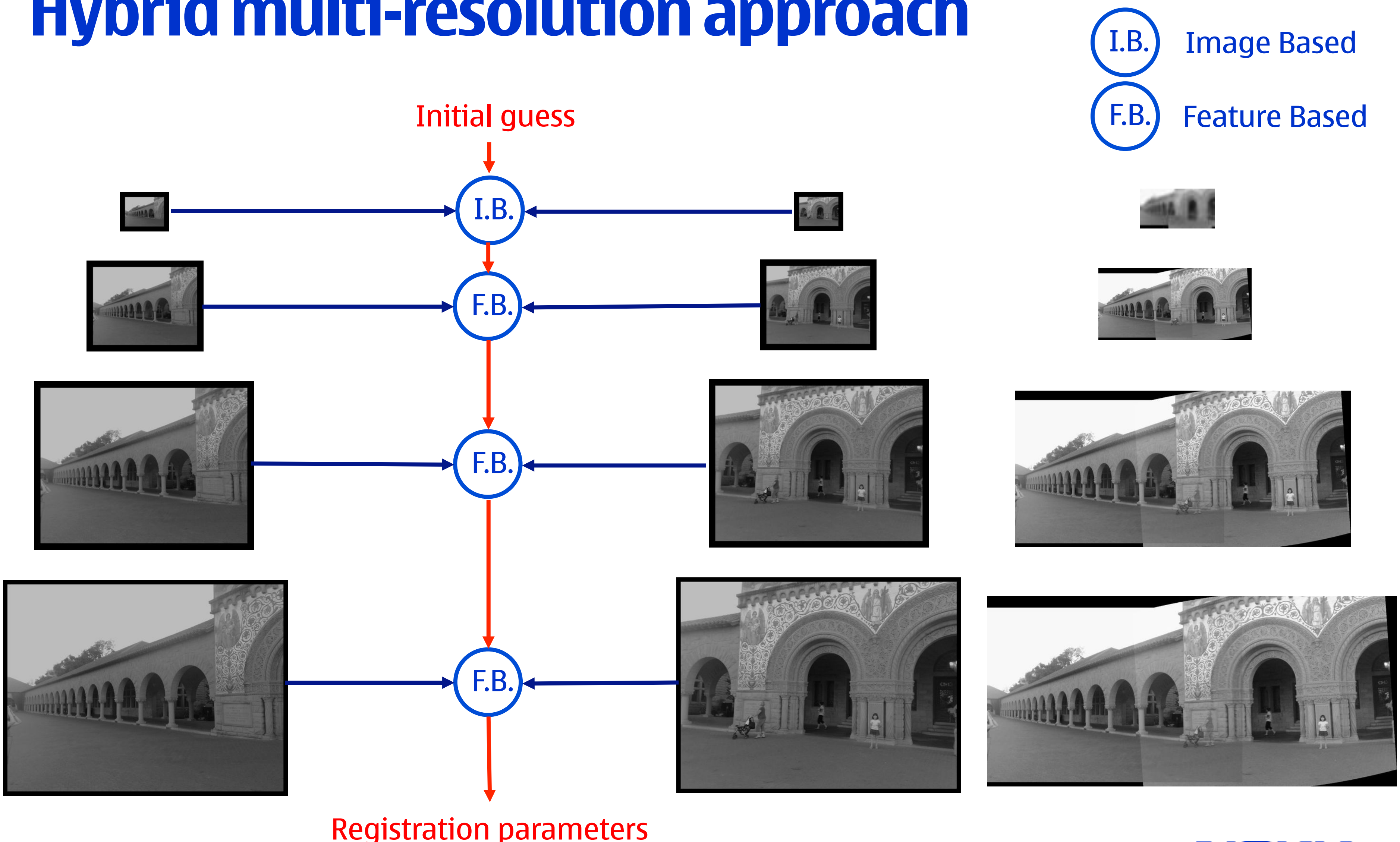
Image registration

Requirements

- Robust to illumination changes
- Robust to object motion in the scene
- Low computational complexity

Registration example (including large illumination changes)

Hybrid multi-resolution approach



K. Pulli, M. Tico, Y. Xiong, X. Wang, C-K. Liang, "Panoramic Imaging System for Camera Phones", ICCE 2010

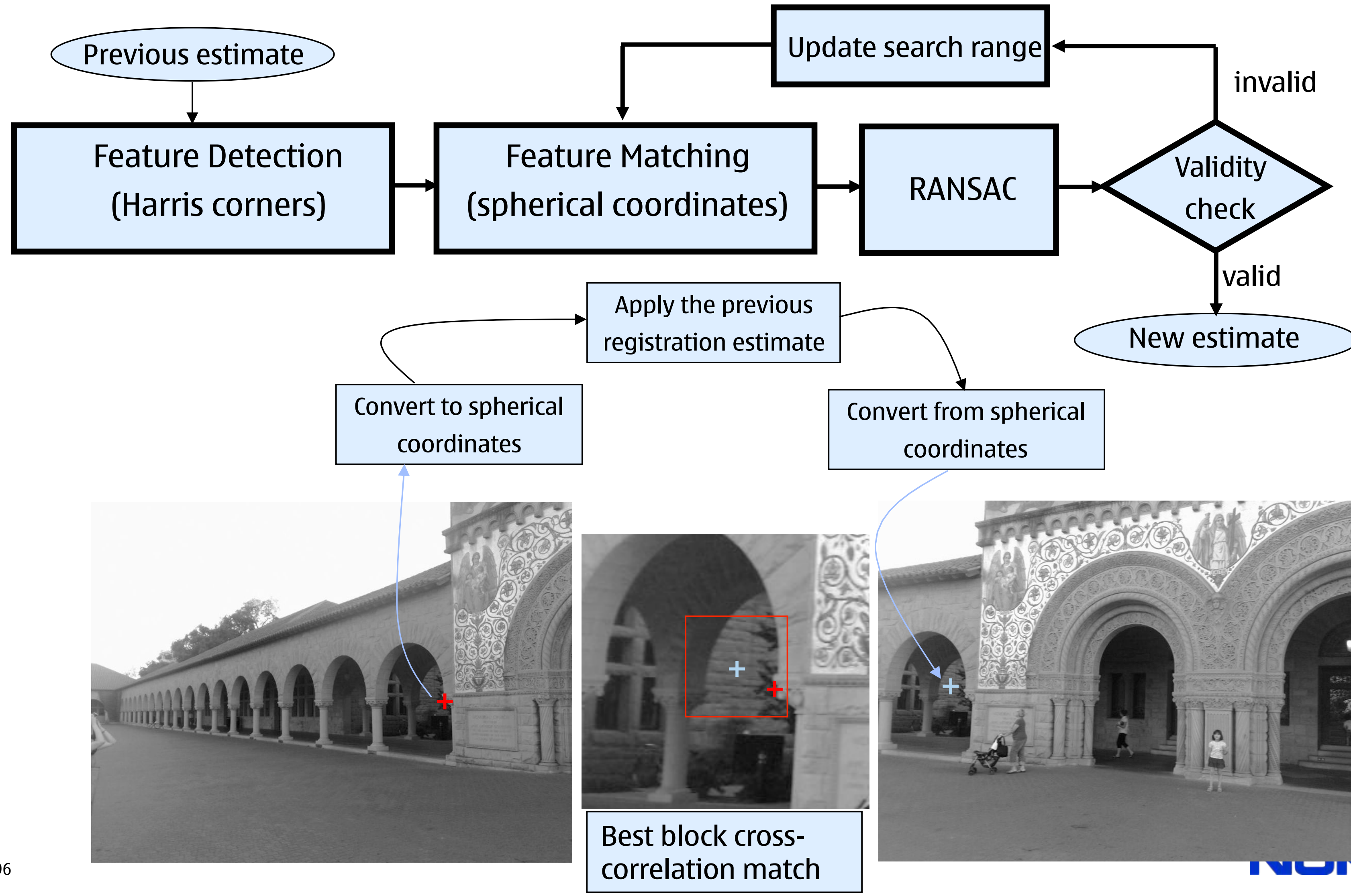
Progression of multi-resolution registration

Actual
size

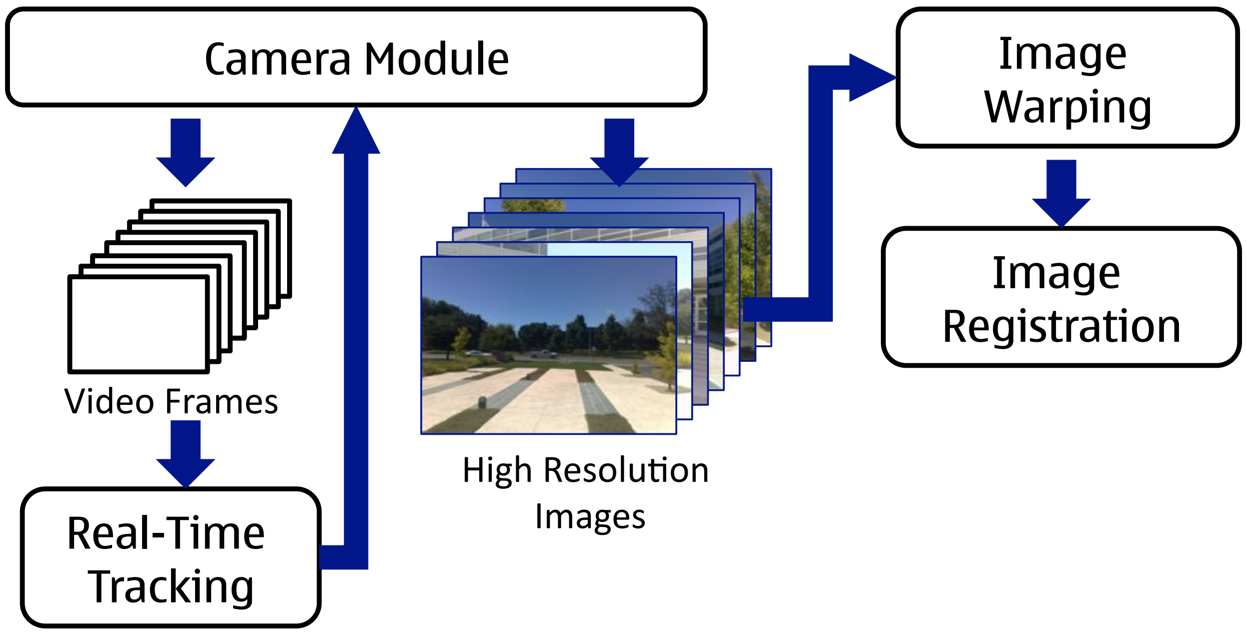
Applied
to hi-res

NOKIA

Feature-based registration



System overview



NOKIA

System overview

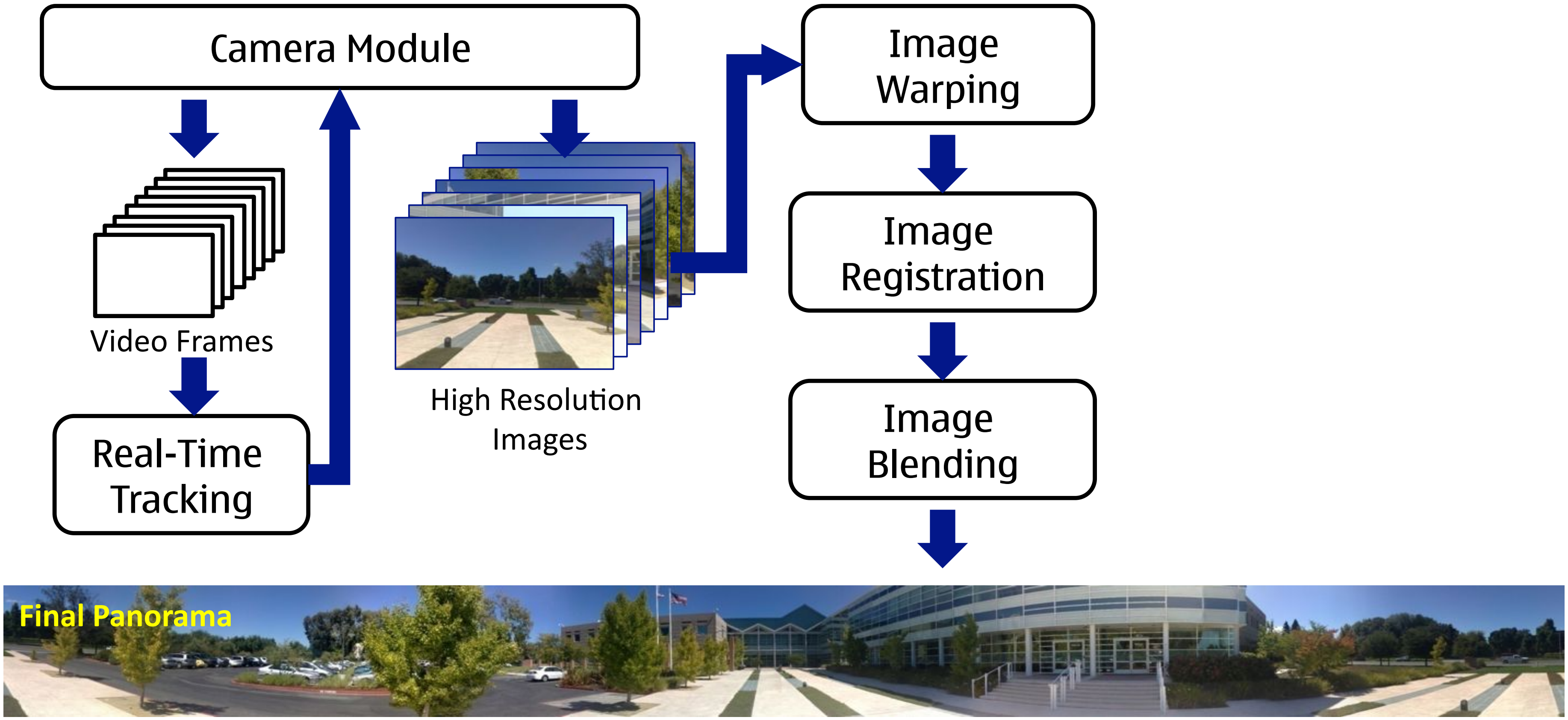


Photo by Marius Tico

Image blending

Directly averaging the overlapped pixels results in ghosting artifacts

- Moving objects, errors in registration, parallax, etc.

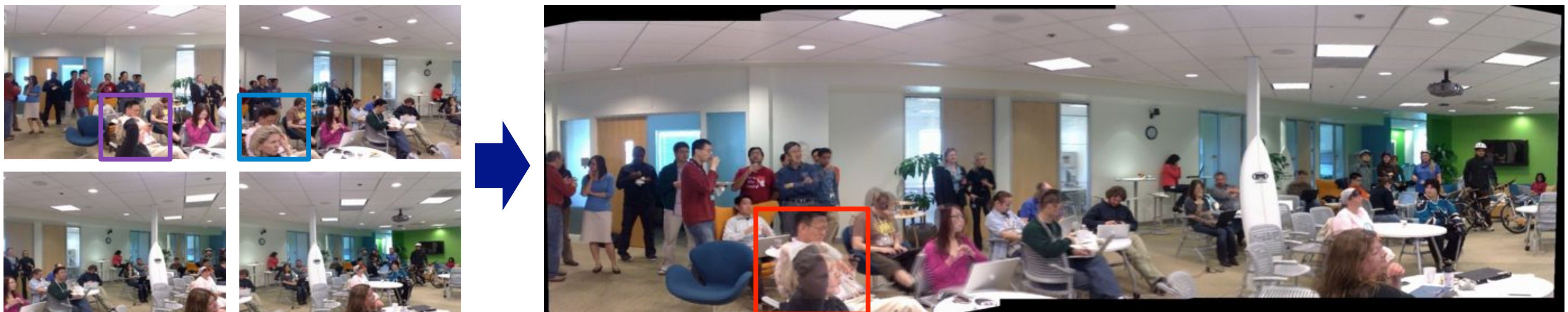
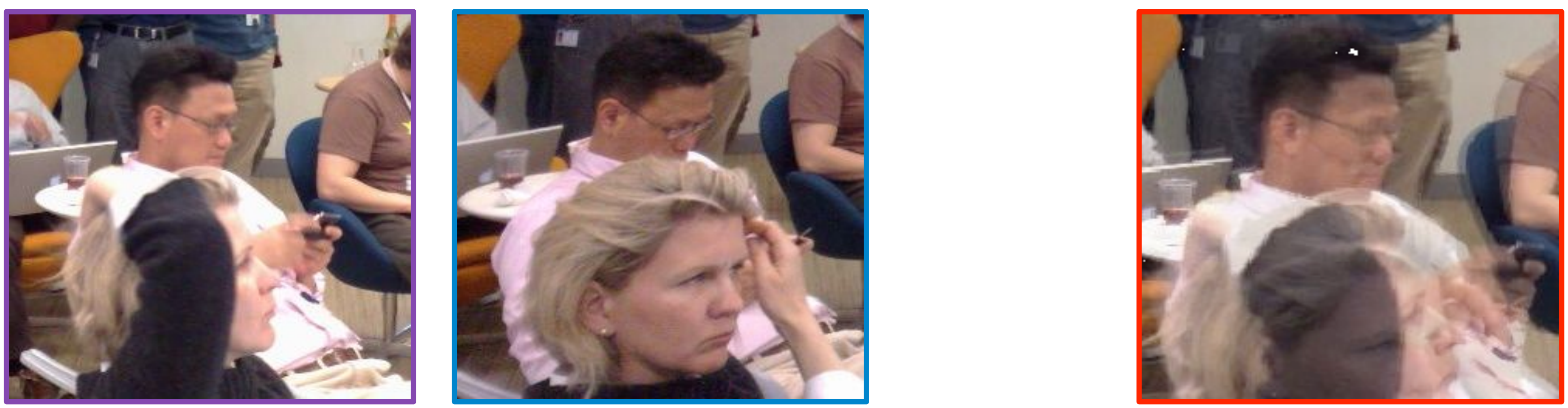


Photo by Chia-Kai Liang

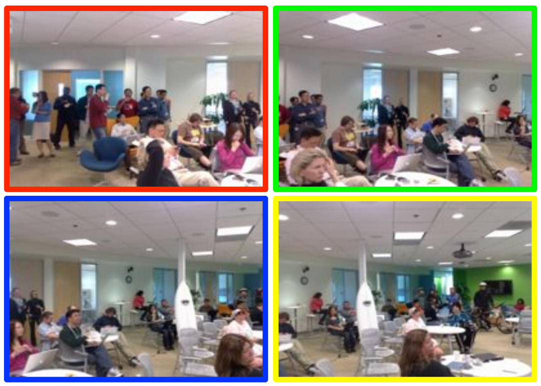


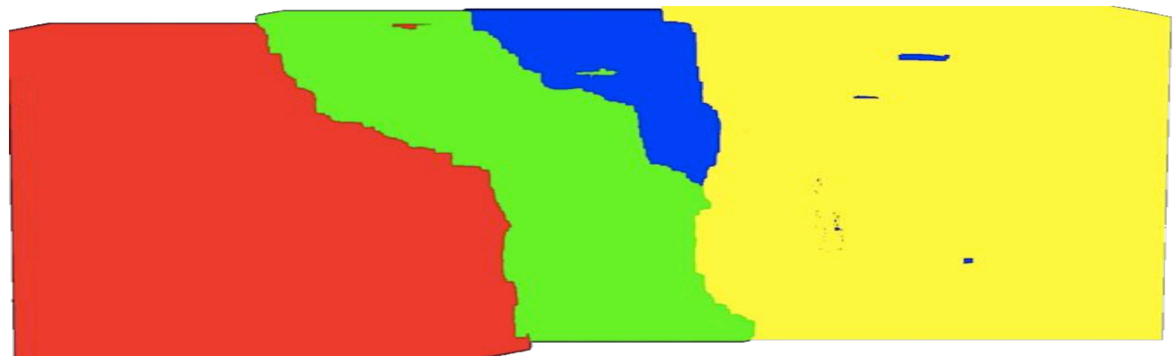
NOKIA

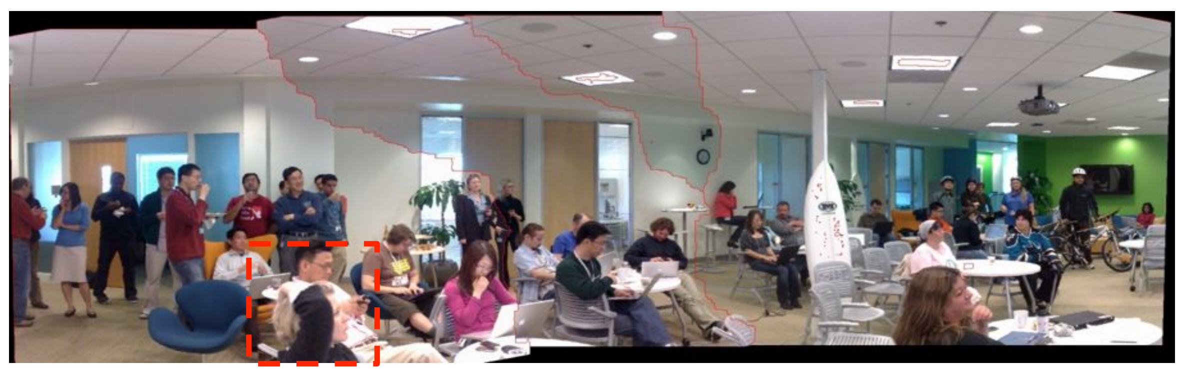
Solution: Image labeling

Assign one input image each output pixel

- Optimal assignment can be found by graph cut [Agarwala et al. 2004]





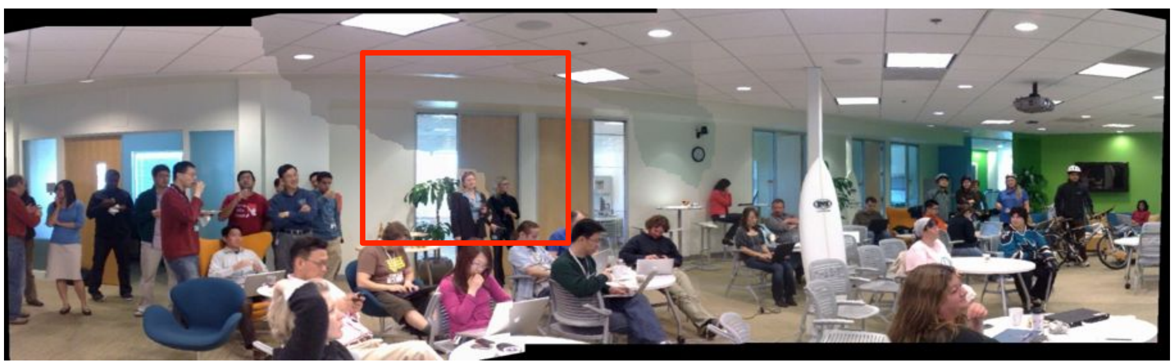


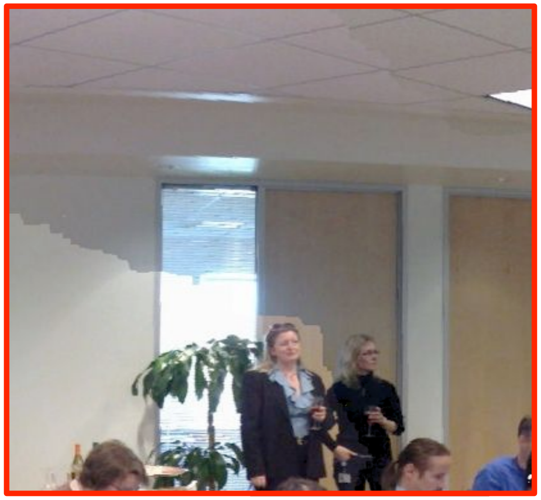
NOKIA

New artifacts

Inconsistency between pixels from different input images

- Different exposure/white balance settings
- Photometric distortions (e.g., vignetting)





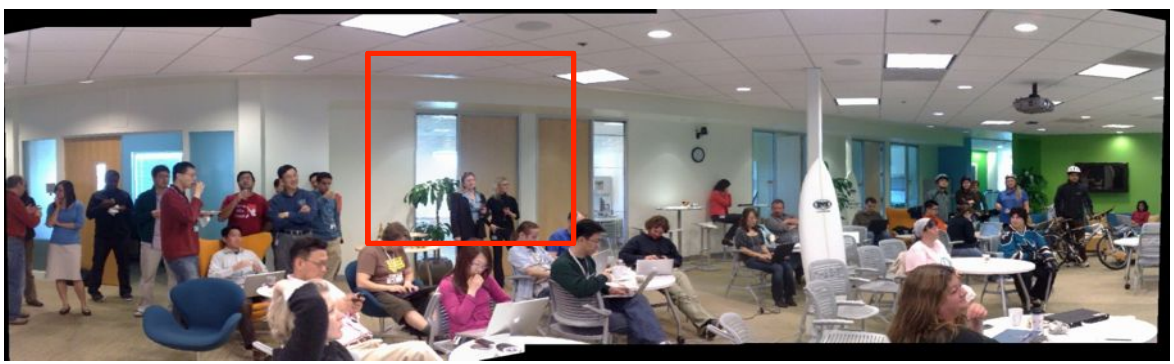
NOKIA

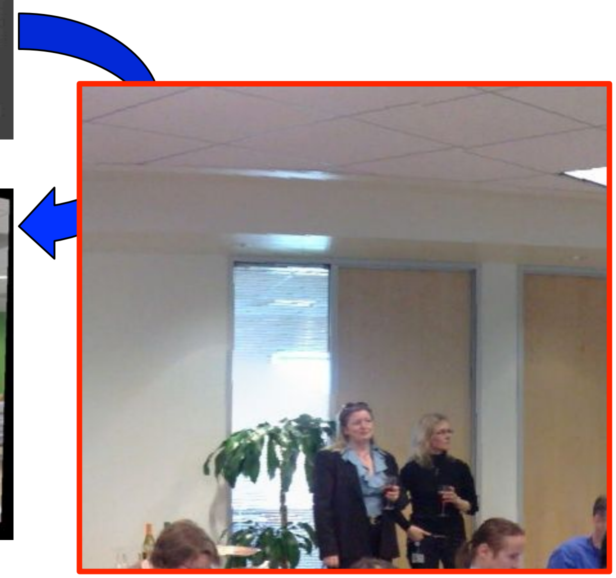
Solution: Poisson blending

Copy the gradient field from the input image

Reconstruct the final image by solving a Poisson equation

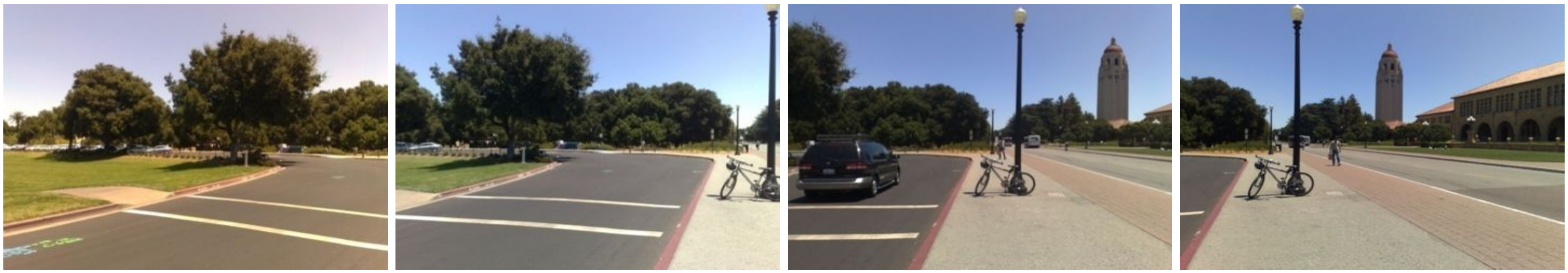
Combined gradient field



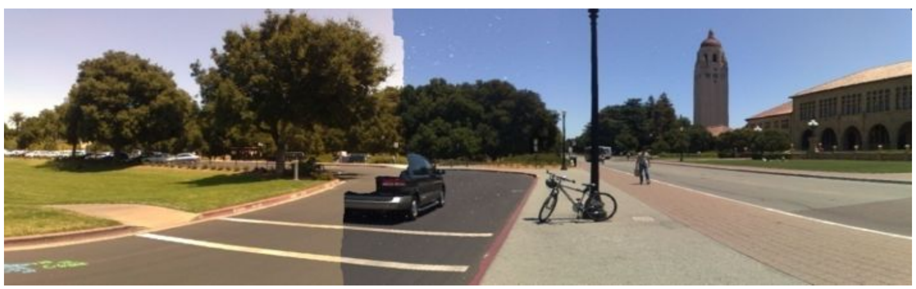


NOKIA

Seam finding gets difficult when colors differ



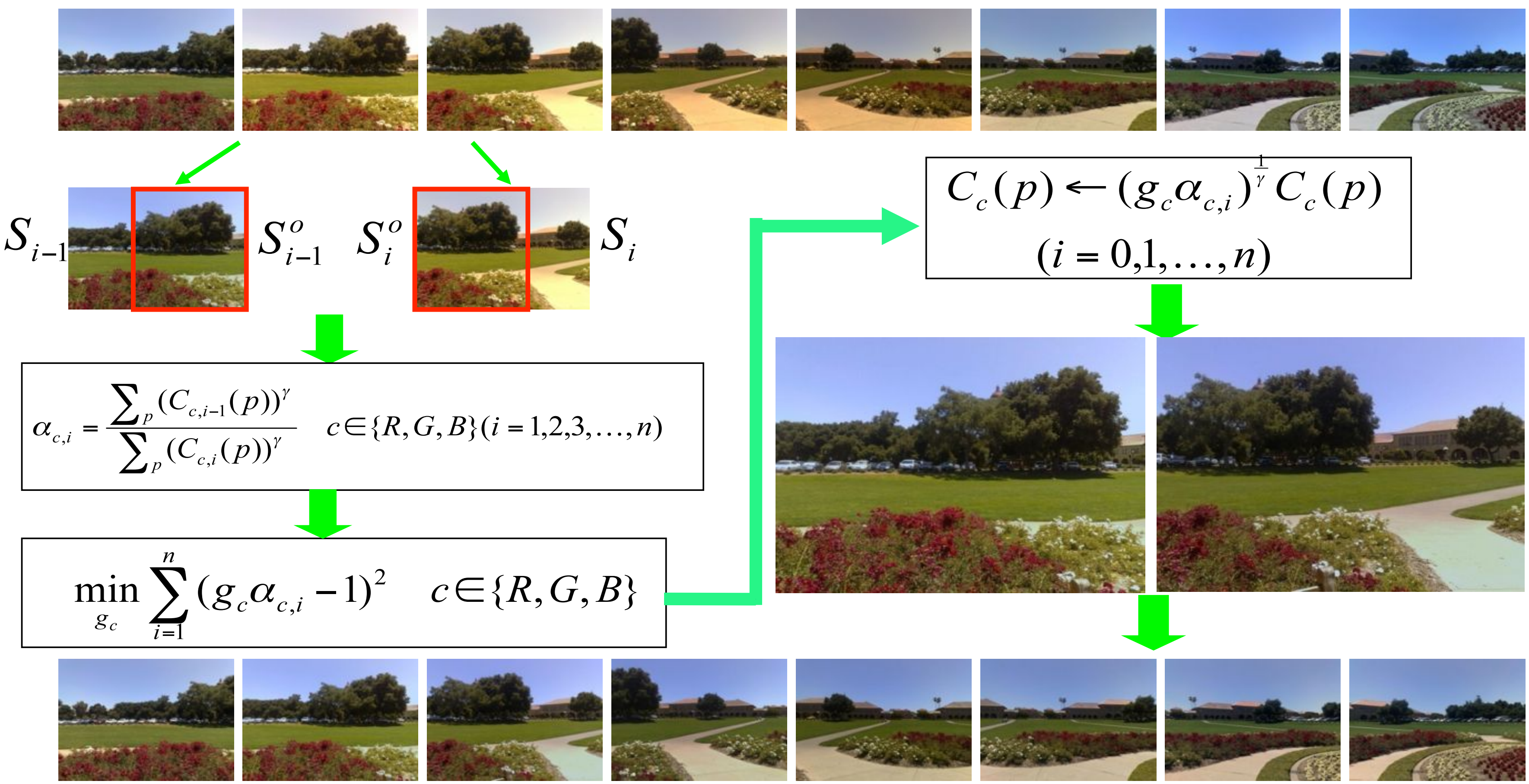
No color correction



With color correction

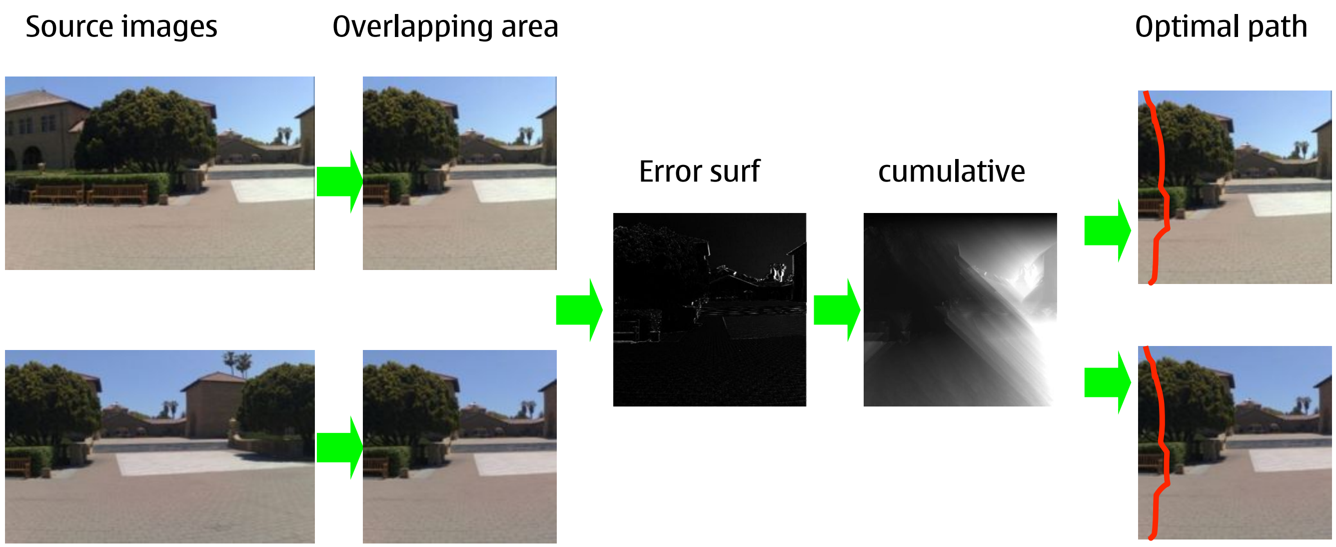
Y. Xiong, K. Pulli, “Fast Panorama Stitching on Mobile Devices”, ICCE 2010

Color correction in linearized RGB



NOKIA

Dynamic programming finds good cuts fast



- Error surface $$e = (I_c^o - S_c^o)^2$$

  Overlapping area in the current composite image

  Overlapping area in the current source image

- Cumulative minimum error surface

$$E(w,h) = e(w,h) + \min(E(w-1,h-1), E(w,h-1), E(w+1,h-1))$$

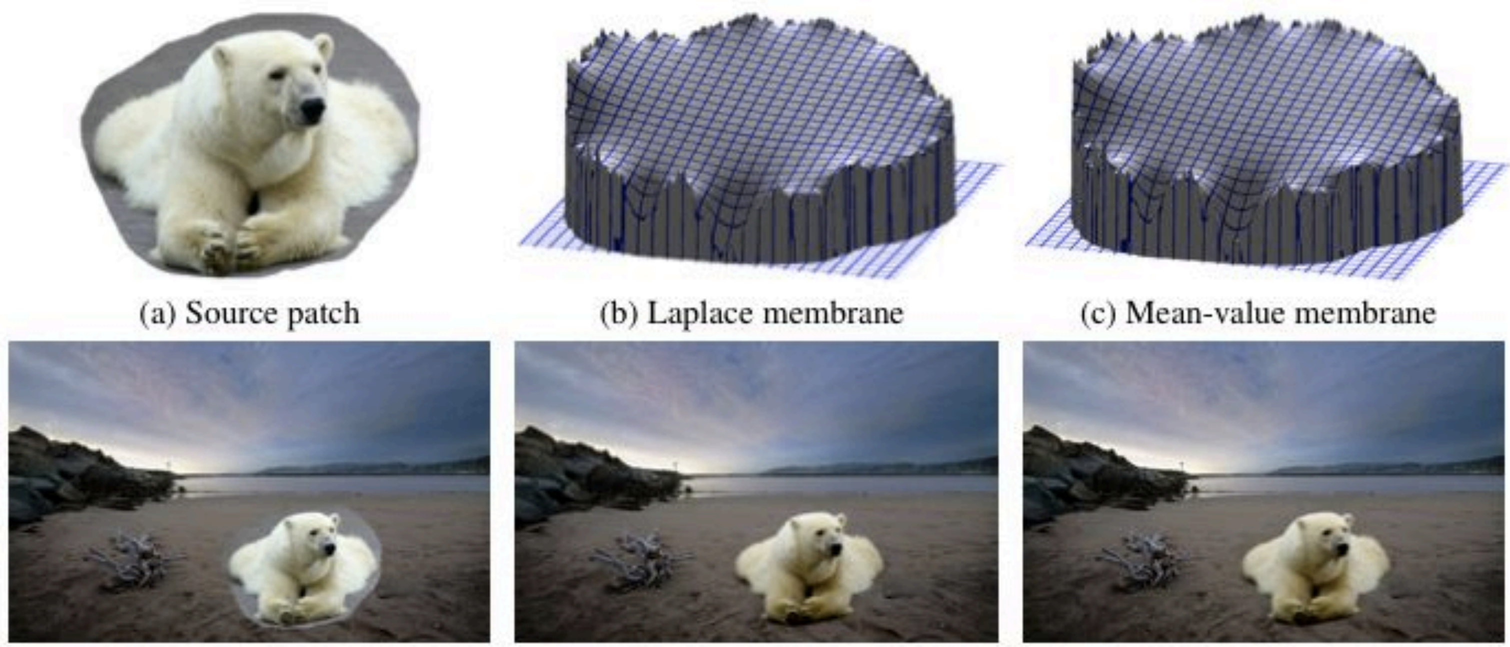
NOKIA

Fast “Poisson” blending

SIGGRAPH 2009

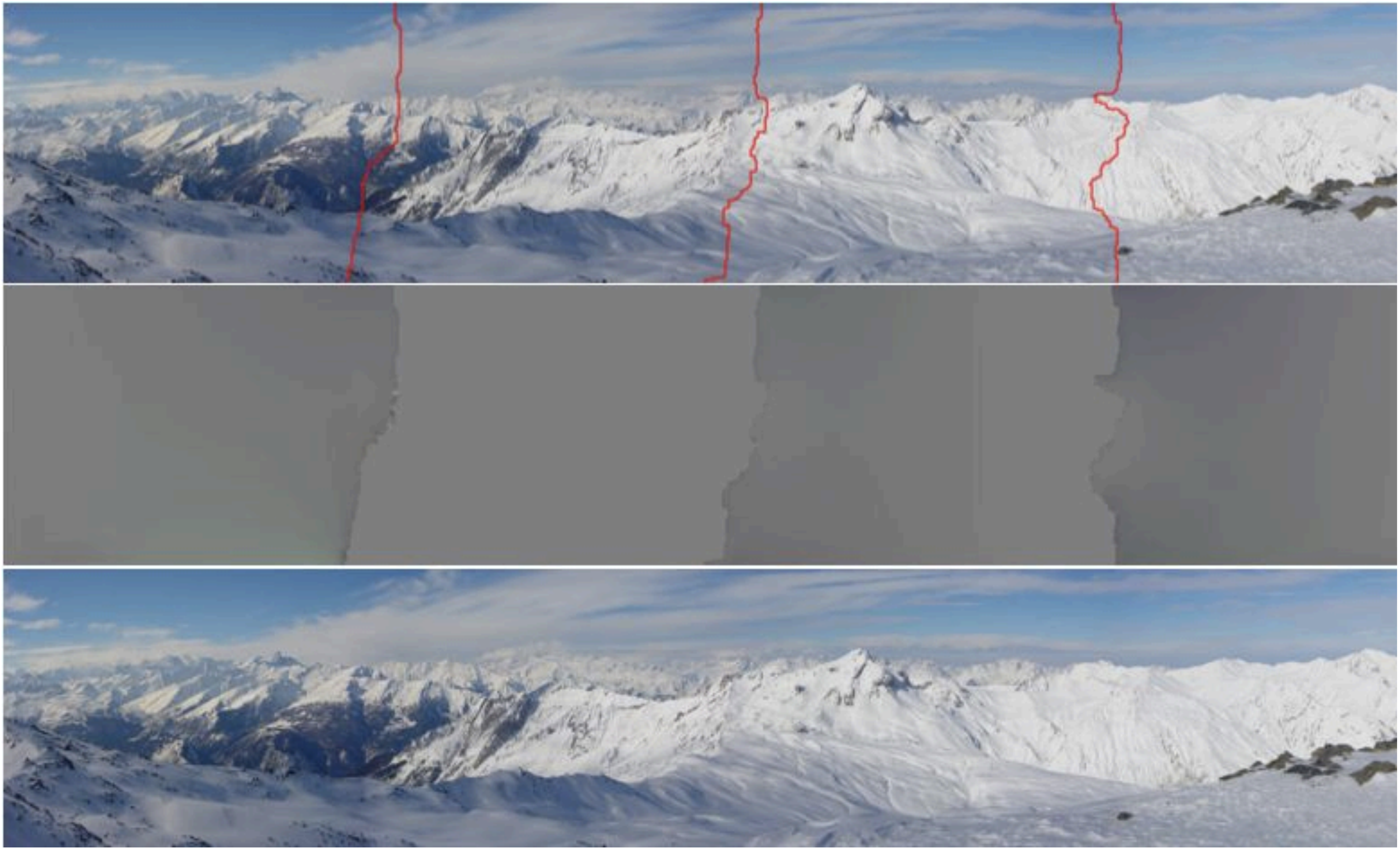
Coordinates for Instant Image Cloning

Zeev Farbman, Hebrew University
Gil Hoffer, Tel Aviv University
Yaron Lipman, Princeton University
Daniel Cohen-Or, Tel Aviv University
Dani Lischinski, Hebrew University

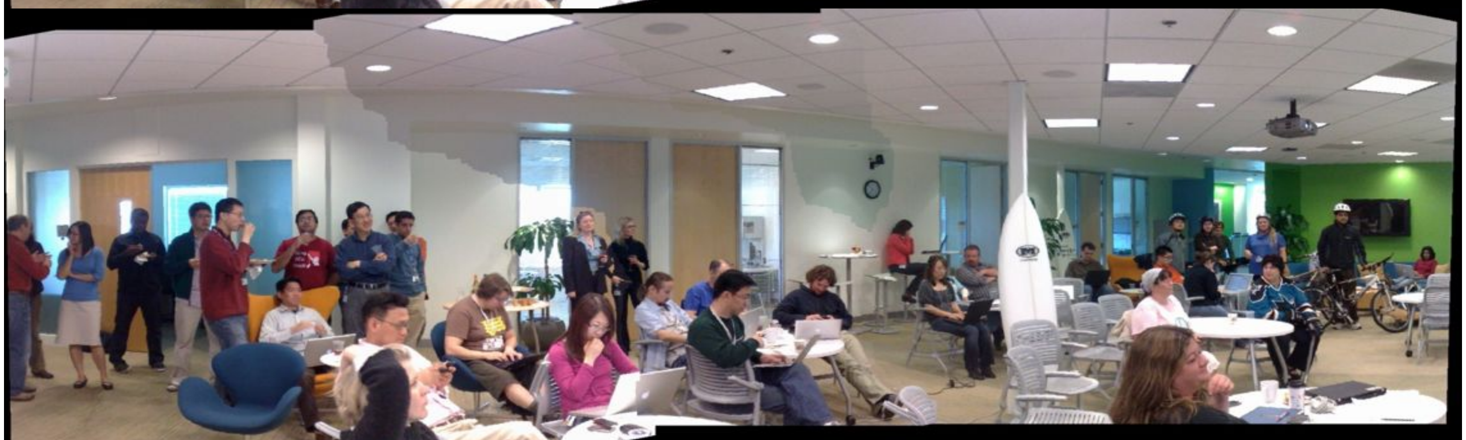


(a) Source patch (b) Laplace membrane (c) Mean-value membrane

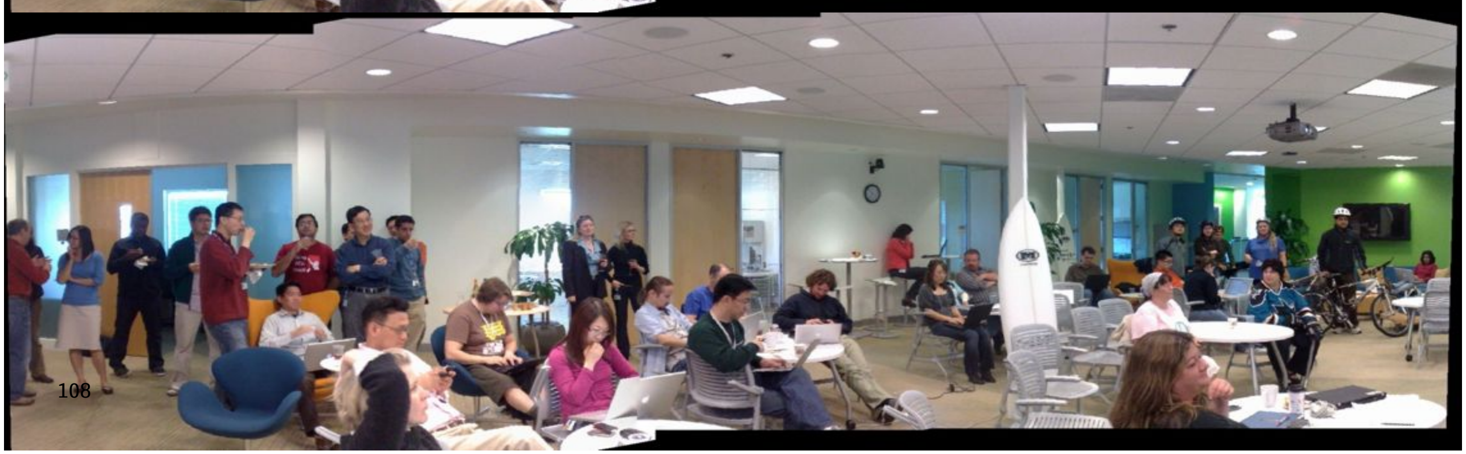
(d) Target image (e) Poisson cloning (f) Mean-value cloning



Alpha blending



After labeling



Poisson blending

System Overview

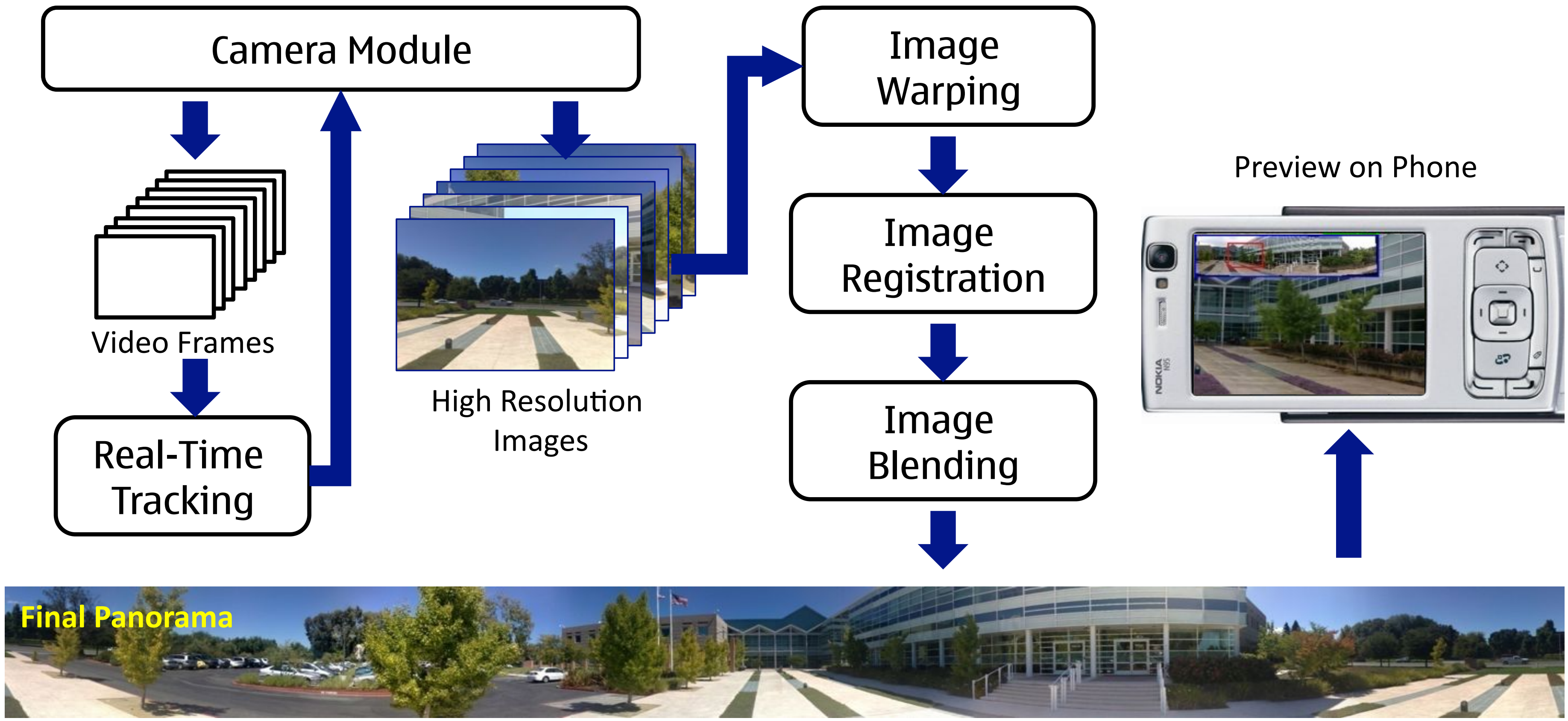


Photo by Marius Tico

Panorama Visualization

Trivial method:

- Show the whole panorama on the screen
- Zooming and panning

No Projection Method is Optimal

Zoom

Spherical Projection

Perspective Projection

Solution: Interpolate the Projection Coordinates

Weights are determined by a sigmoid function of zoom factor

Slide credits

Fredo Durand
Alyosha Efros
Bill Freeman
Marc Levoy
Chia-Kai Liang
Steve Seitz
Rick Szeliski
Marius Tico
Yingen Xiong
...