



UNIVERSITY OF
SOUTH CAROLINA

CSCE 574 ROBOTICS

Computer Vision

Slides courtesy of Professor Gregory Dudek

Alberto Quattrini Li

albertoq@cse.sc.edu



Ioannis Rekleitis

Why vision?

- Passive (emits nothing).
 - Discreet.
 - Energy efficient.
- Intuitive.
- Powerful (works well for us, right?)
- Long and short range.
- Fast.



So, what's the problem?

- How hard is vision? Why do we think it is do-able?

Problems:

- Slow.
- Data-heavy.
- Impossible.
- Mixes up many factors.



Data heavy



From GoPro HERO3+ at Barbados 2015 Field Trials

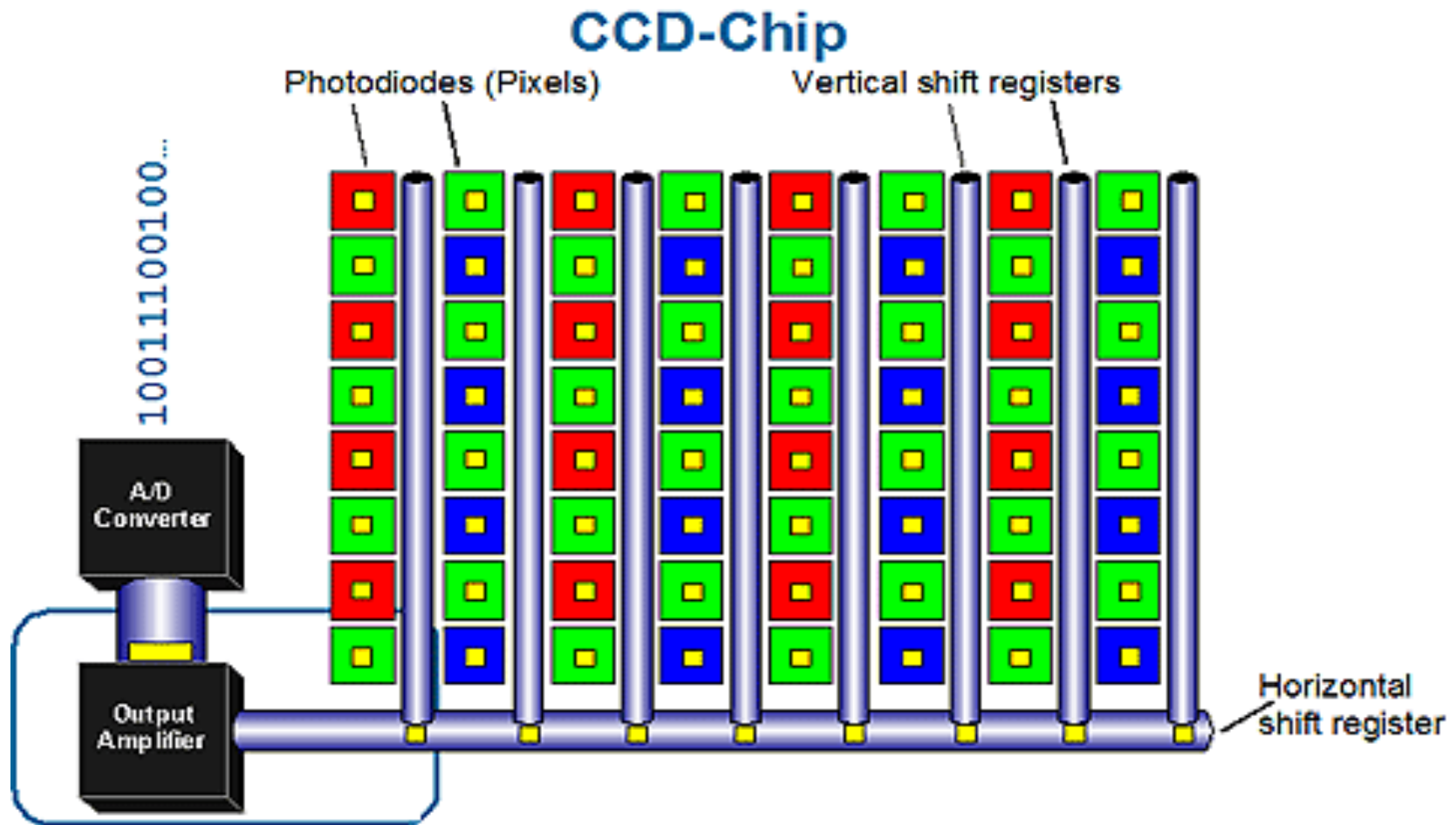
		1920											
1080	[43	43	42	40	39	...	29	29	31	33]	R
		42	41	40	39	38	...	31	32	35	37		
		⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮	⋮		
		54	57	60	62	66	...	42	43	56	46		
1080	[129	129	129	129	128	...	149	149	151	153]	G
		128	128	127	128	127	...	151	152	155	157		
		⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮	⋮		
		146	146	148	148	148	...	149	150	151	152		
1080	[146	146	146	145	146	...	166	166	168	170]	B
		145	145	144	144	145	...	168	169	172	174		
		⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮	⋮		
		159	160	160	161	162	...	165	166	165	166		

Aliasing

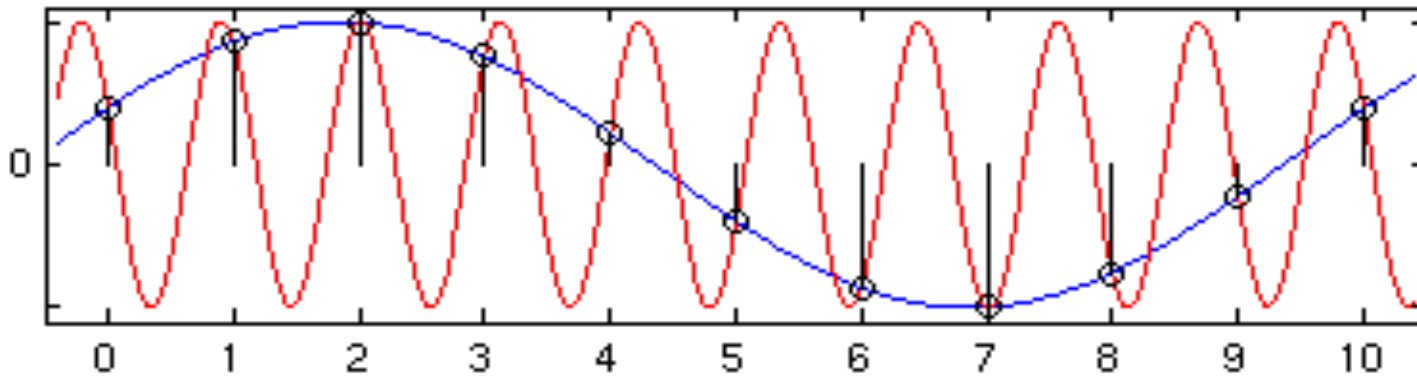
- Images are not actually continuous.
- The sampling (and hardware) issues lead to a few other minor problems.



Aliasing



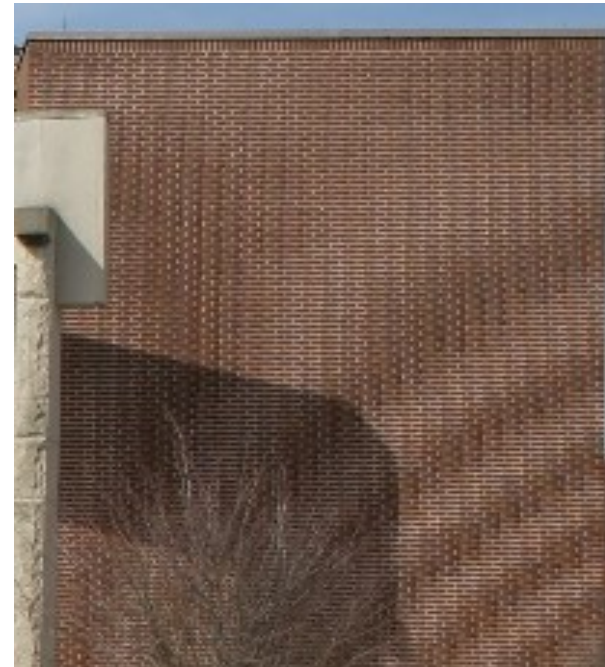
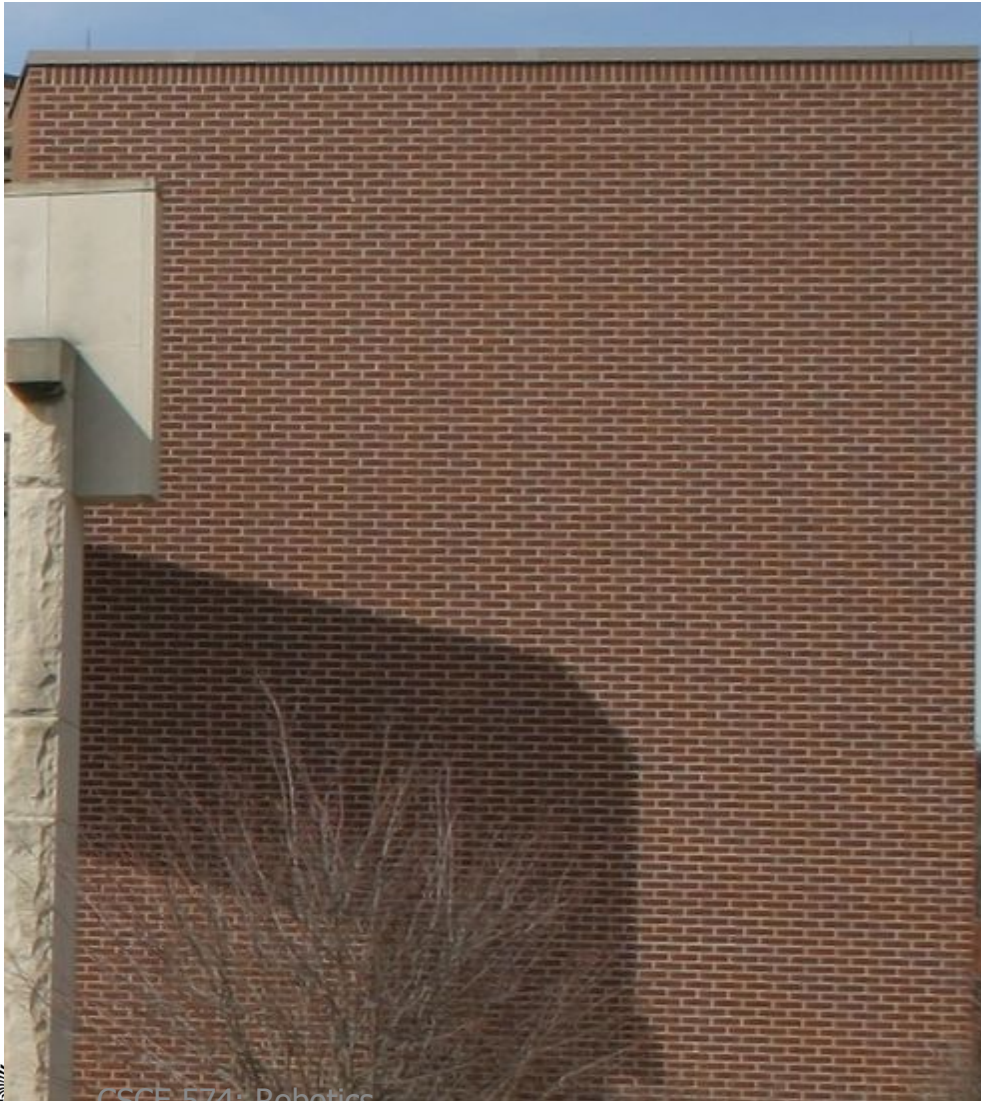
Aliasing



- To avoid: $f_{\text{sampling}} > 2F_{\text{max}}$
 - Nyquist Rate



Aliasing: Moiré Patterns



Ill-posed

- What a camera does to the 3d world...

Shigeo Fukuda



squeezes away one dimension

<http://www.psychologie.tu-dresden.de/i1/kaw/diverses> Material/www.illusionworks.com/html/art_of_shigeo_fukuda.html



Ill-posed

- What a camera does to the 3d world...

Shigeo Fukuda



<http://www.psychologie.tu-dresden.de/i1/kaw/diverses> Material/www.illusionworks.com/html/art_of_shigeo_fukuda.html

Ill-posed

- In trying to extract 3d structure from 2d images, vision is an *ill-posed* problem.



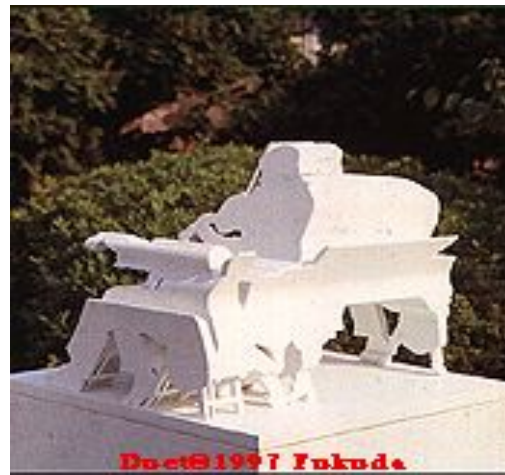
Ill-posed

- In trying to extract 3d structure from 2d images, vision is an *ill-posed* problem.



Ill-posed

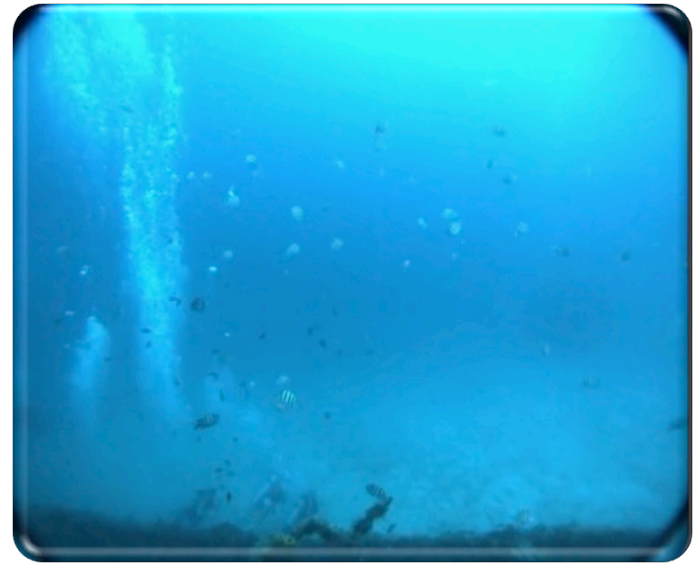
- In trying to extract 3d structure from 2d images, vision is an *ill-posed* problem.



- An image isn't enough to disambiguate the many possible 3d worlds that could have produced it.

Difficult scenarios

- In certain settings, such as the underwater, robotic vision is particularly challenging
 - Different lighting conditions
 - Color loss
 - Hazing and blur
 - Texture loss



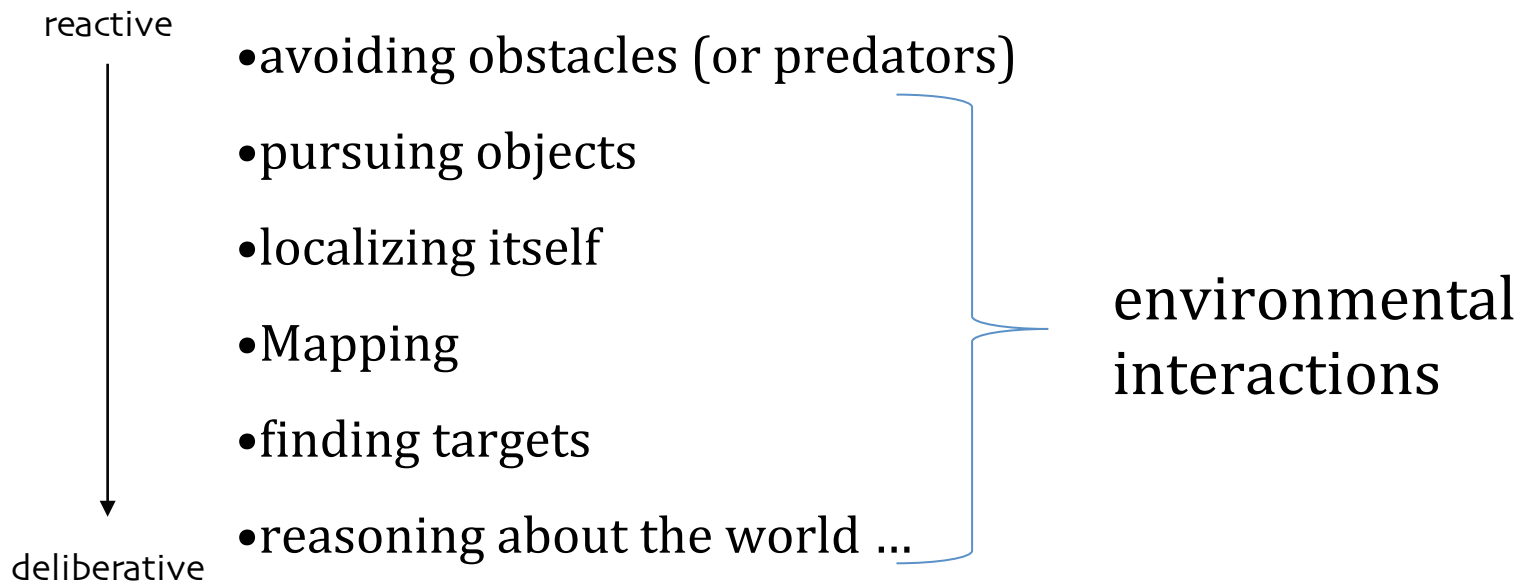
What does a robot need ?

doesn't need a full interpretation of available images

“This is Prof. X in his office offering me a cup of iced tea.”

does need information about what to do...

“Run Away!!”



Key problems

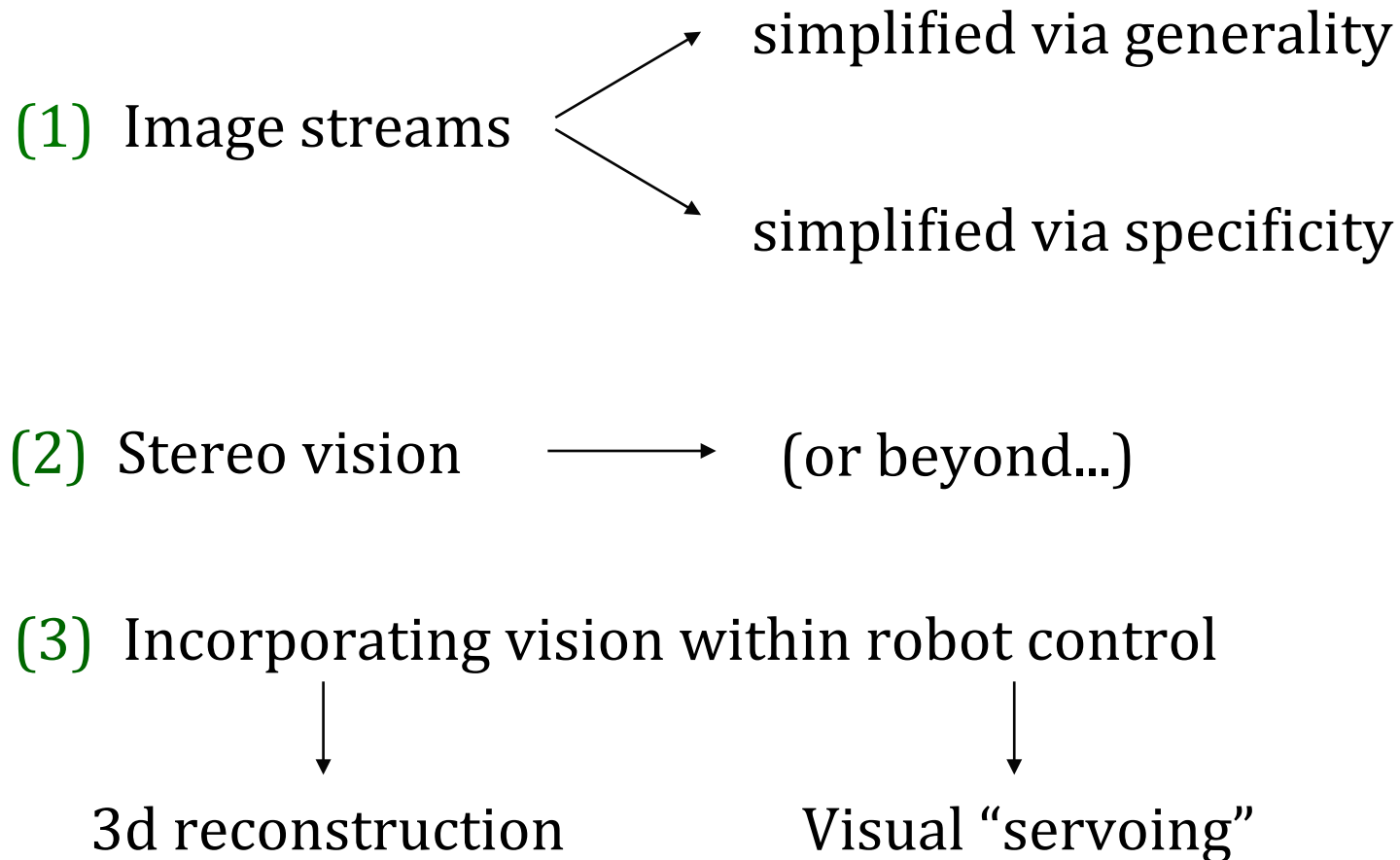
- Recognition:
 - What is that thing in the picture?
 - What are all the things in the image?
- Scene interpretation
 - Describe the image?
- Scene “reconstruction”:
 - What is the 3-dimensional layout of the scene?
 - What are the physical parameters that gave rise to the image?
 - What is a description of the scene?

Notion of an “inverse problem.”

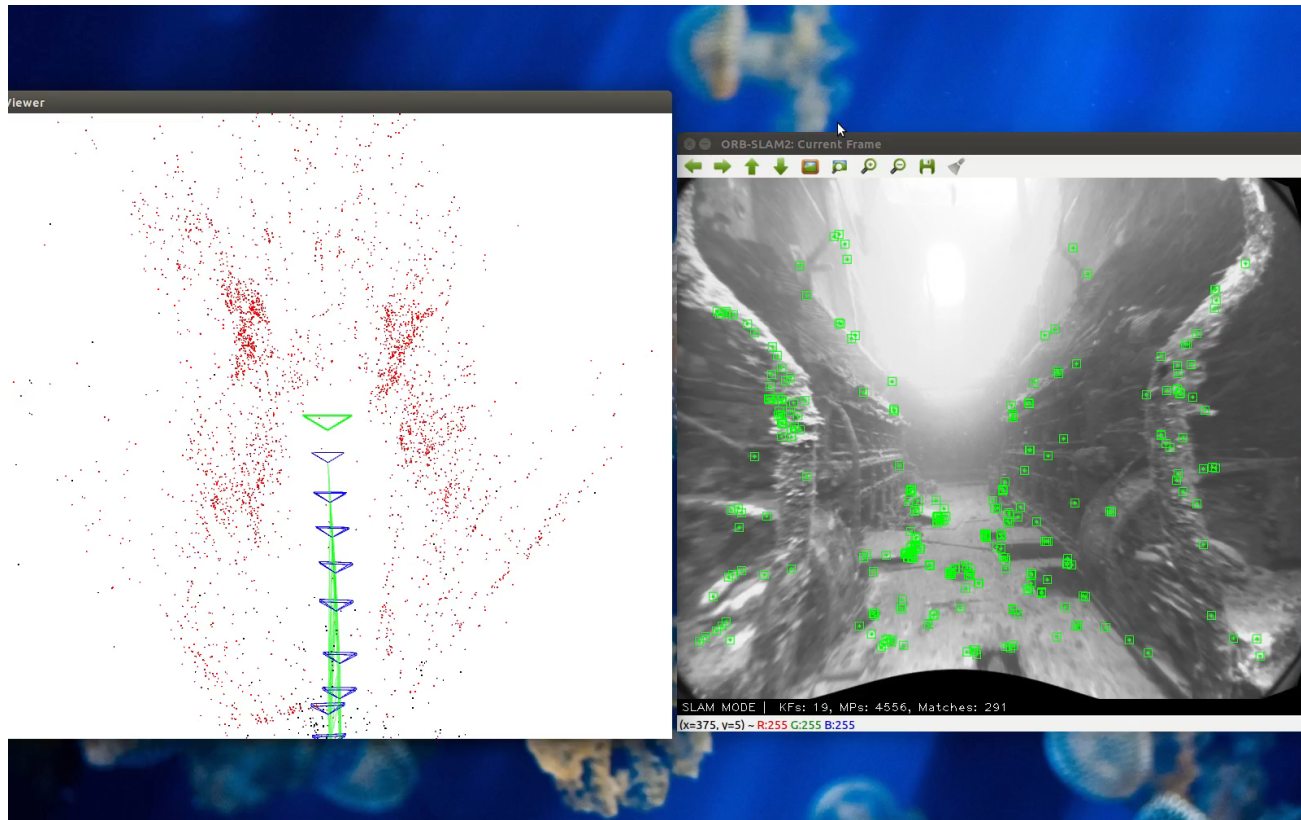


Robot vision sampler

A brief overview of robotic vision processing...



3d reconstruction



Visual Servoing



Computer vision algorithms

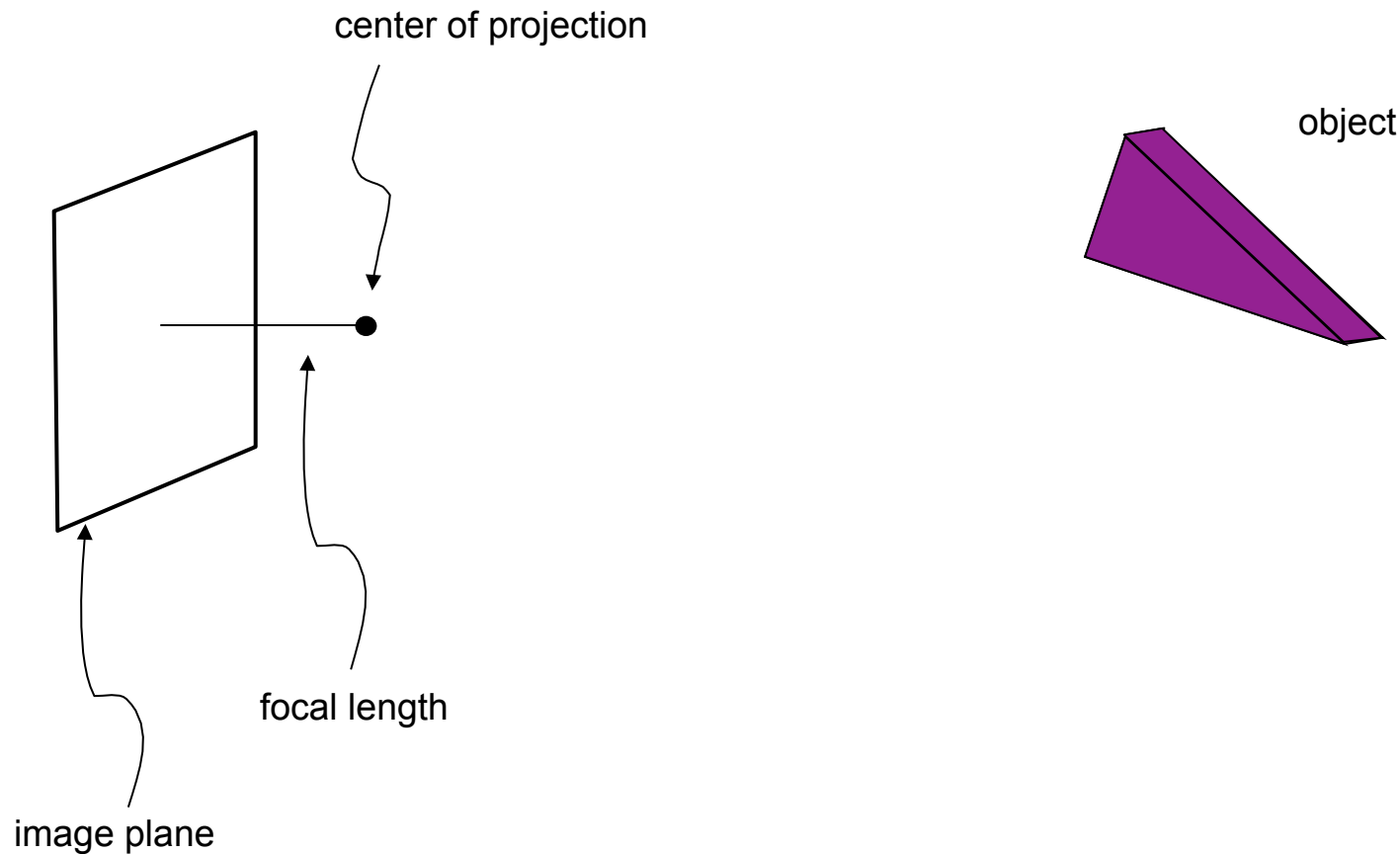
- Image processing
- Geometric computer vision
- Semantic computer vision

- It is fundamental first to understand image formation

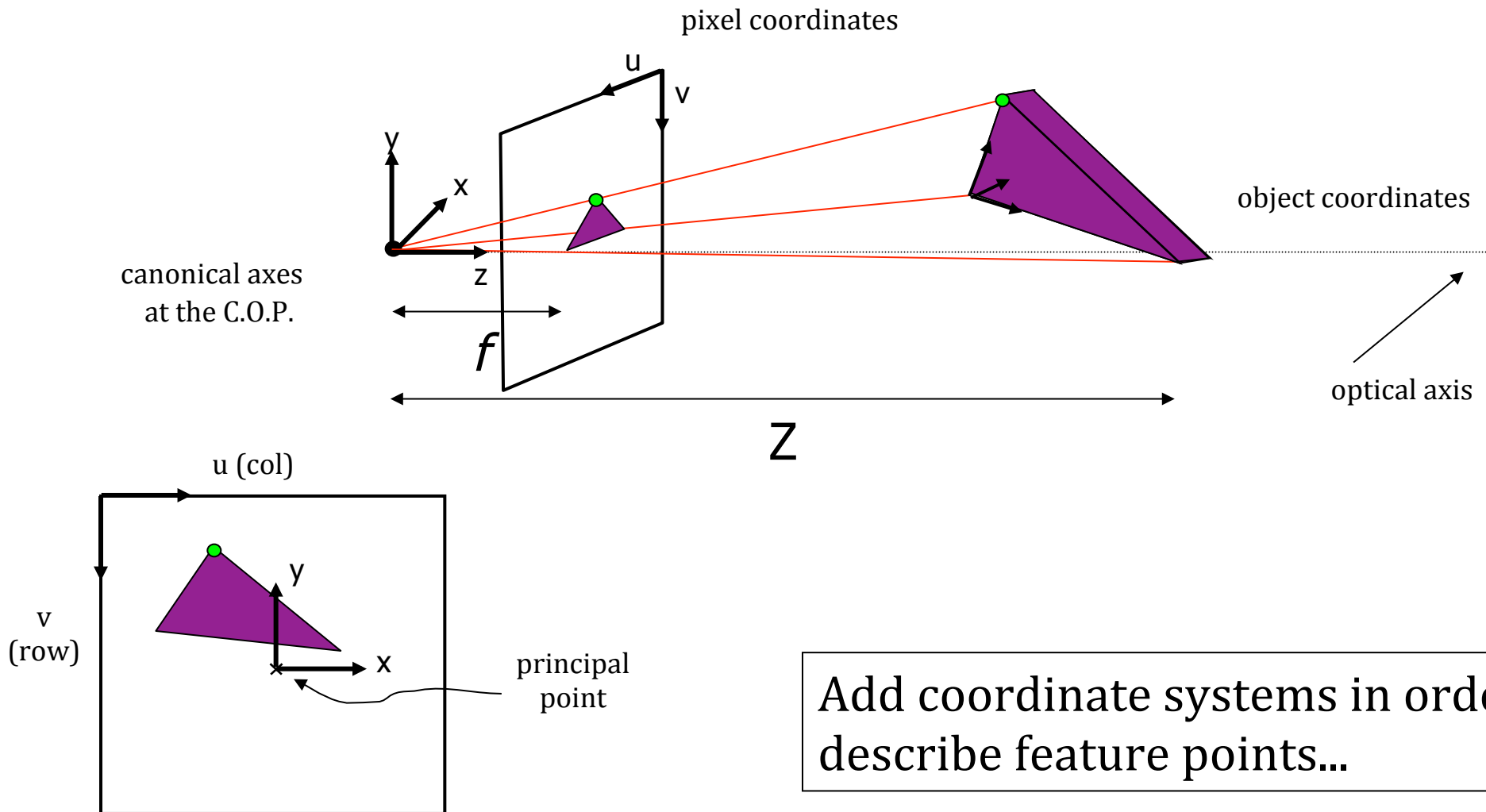


Camera Geometry

3D \rightarrow 2D transformation: perspective projection



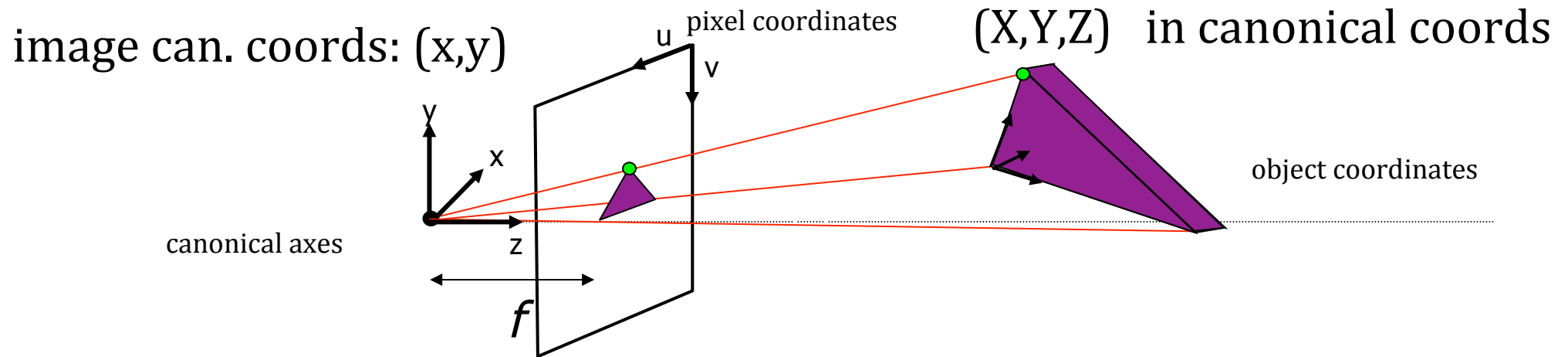
Coordinate Systems



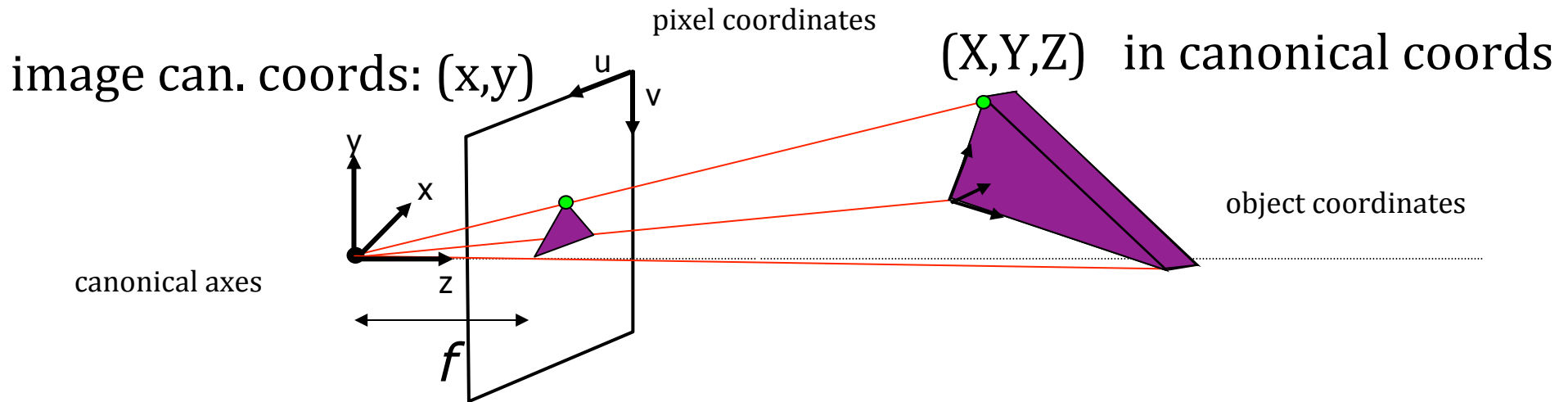
Add coordinate systems in order to describe feature points...



Coordinate Systems



From 3d to 2d



$$x = \frac{fX}{Z}$$

$$y = \frac{fY}{Z}$$

a nonlinear transformation

goal: to recover information about (X,Y,Z) from (x,y)

Camera Calibration

- Camera Model

- $[u \ v \ 1]$ Pixel coords

- $[x_w \ y_w \ z_w \ 1]^T$ World coords

$$z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A \begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

- Intrinsic Parameters

- $\alpha_x = f \cdot m_x, \alpha_y = f \cdot m_y$ focal lengths in pixels

- γ skew coefficient

- u_0, v_0 focal point

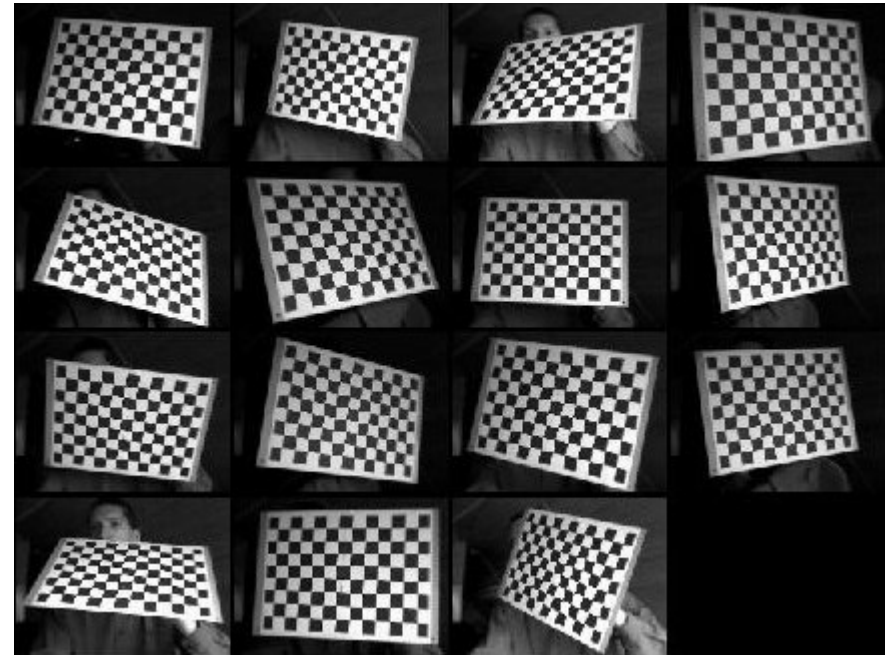
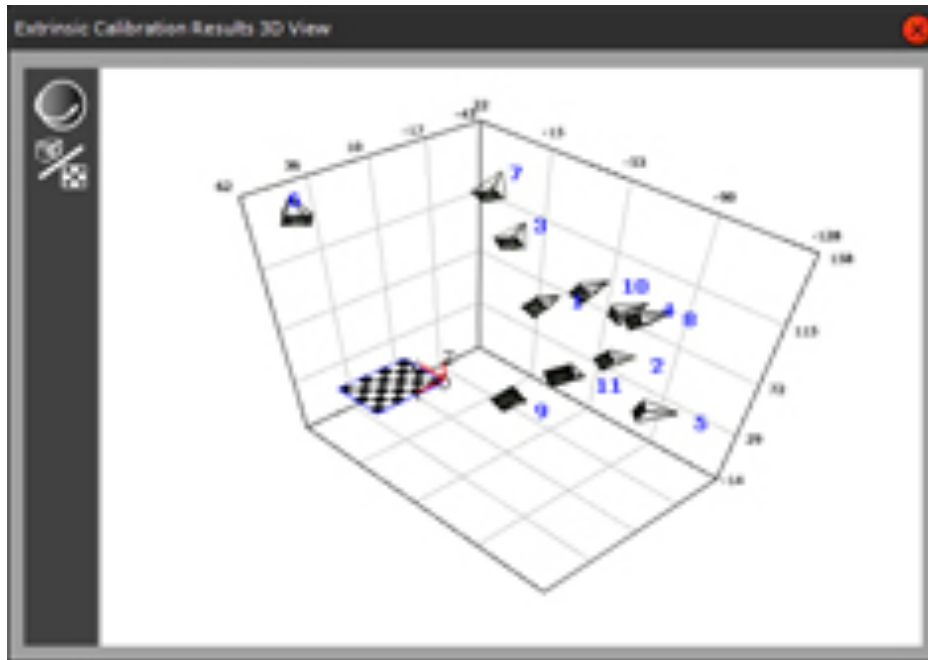
$$A = \begin{bmatrix} \alpha_x & \gamma & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Extrinsic Parameters

- $[R \ T]$ Rotation and Translation



Camera Calibration



Existing packages in MATLAB, OpenCV, etc

Rectified Image Sample

Unrectified



Rectified



From Clearpath Husky Axis M1013 camera

Rectified Image Sample

Unrectified



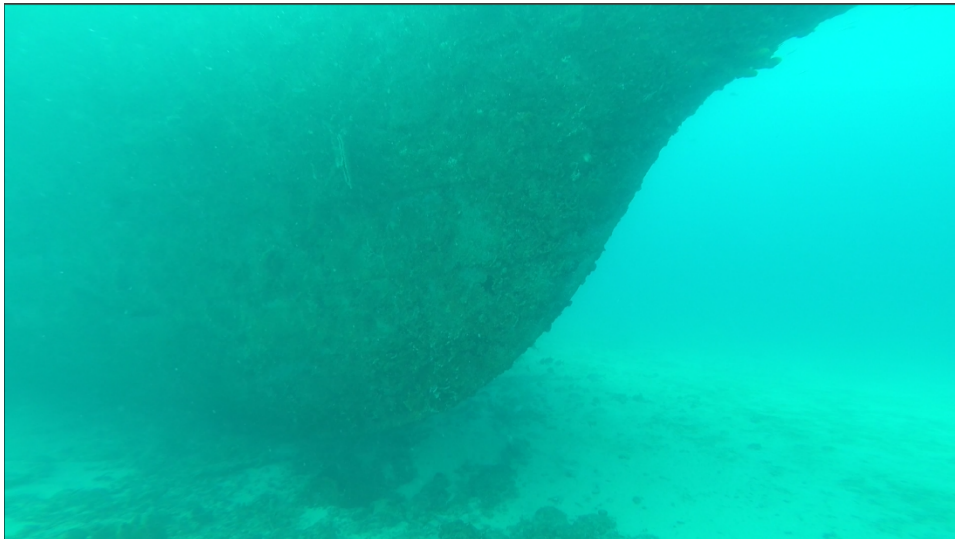
Rectified



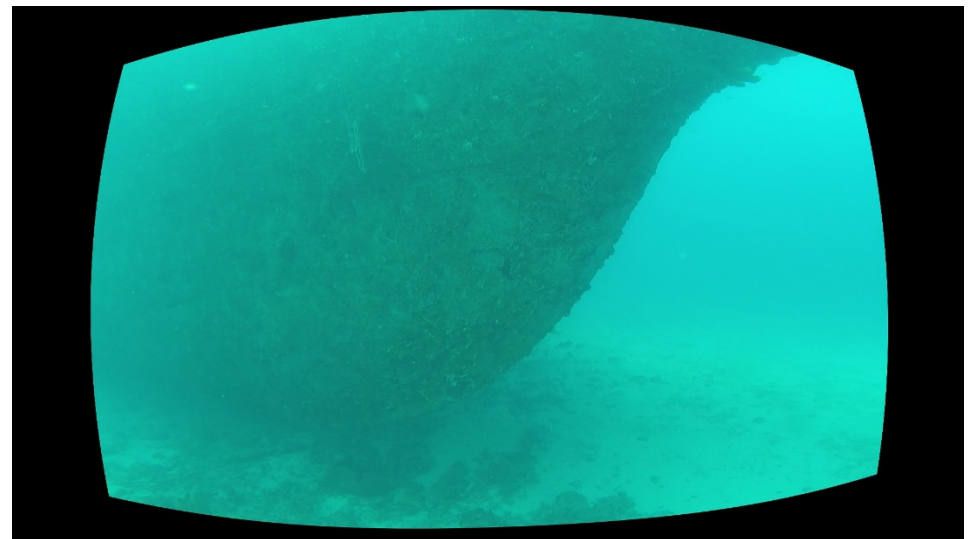
From Parrot ARDrone 2.0 front camera

Rectified Image Sample

Unrectified



Rectified



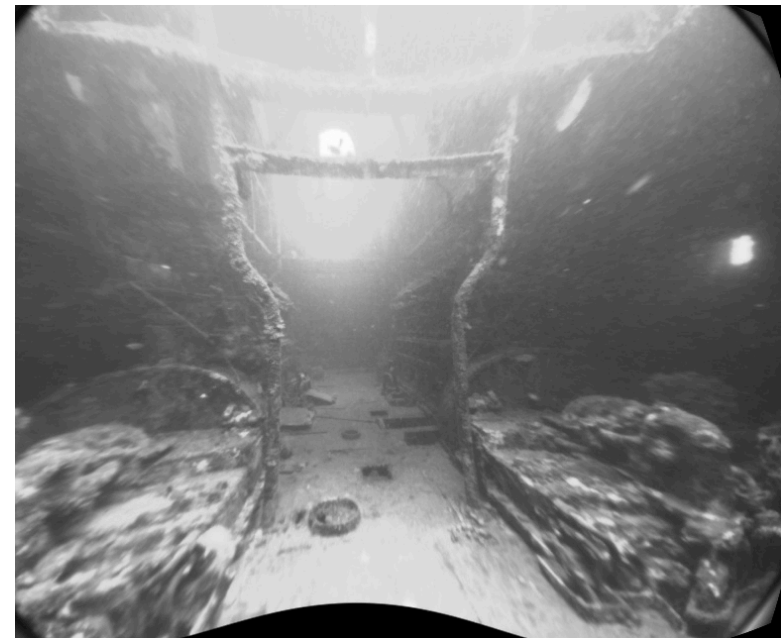
From GoPro HERO3+ at Barbados 2015 Field Trials

ReRectified Image Sample

Rectified

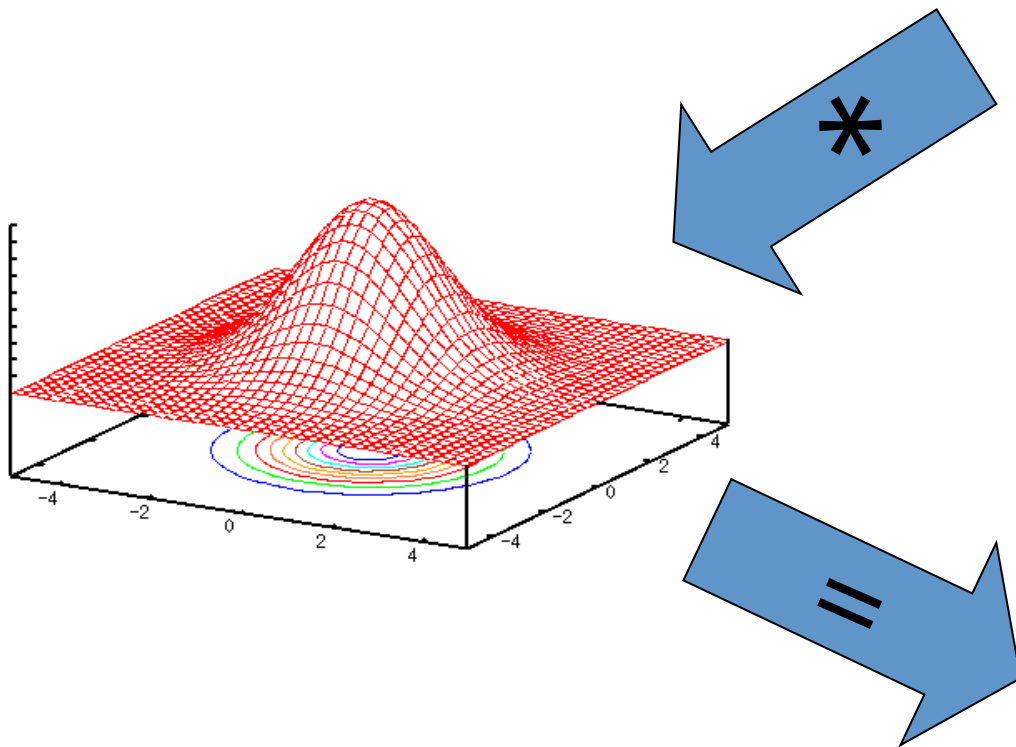


ReRectified



From Aqua front camera at Barbados 2013 Field Trials

Gaussian Blur



Gaussian Blur and Noise

$\sigma = 4.0$ pix



$\sigma = 8.0$ pix



$\sigma = 12.0$ pix



$\sigma = 4.0$ pix



$\sigma = 8.0$ pix



$\sigma = 12.0$ pix



Gaussian Blur, Noise, Sobel

$\sigma = 0.0$ pix



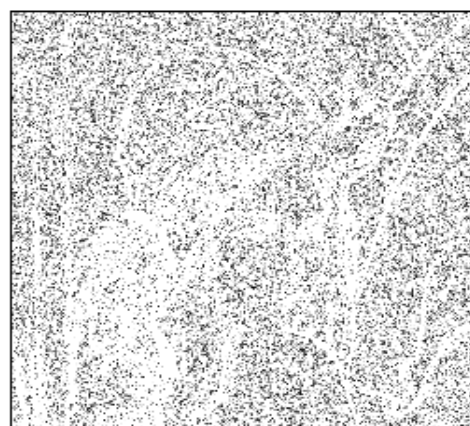
$\sigma = 4.0$ pix



$\sigma = 8.0$ pix



$\sigma = 0.0$ pix



$\sigma = 4.0$ pix



$\sigma = 8.0$ pix



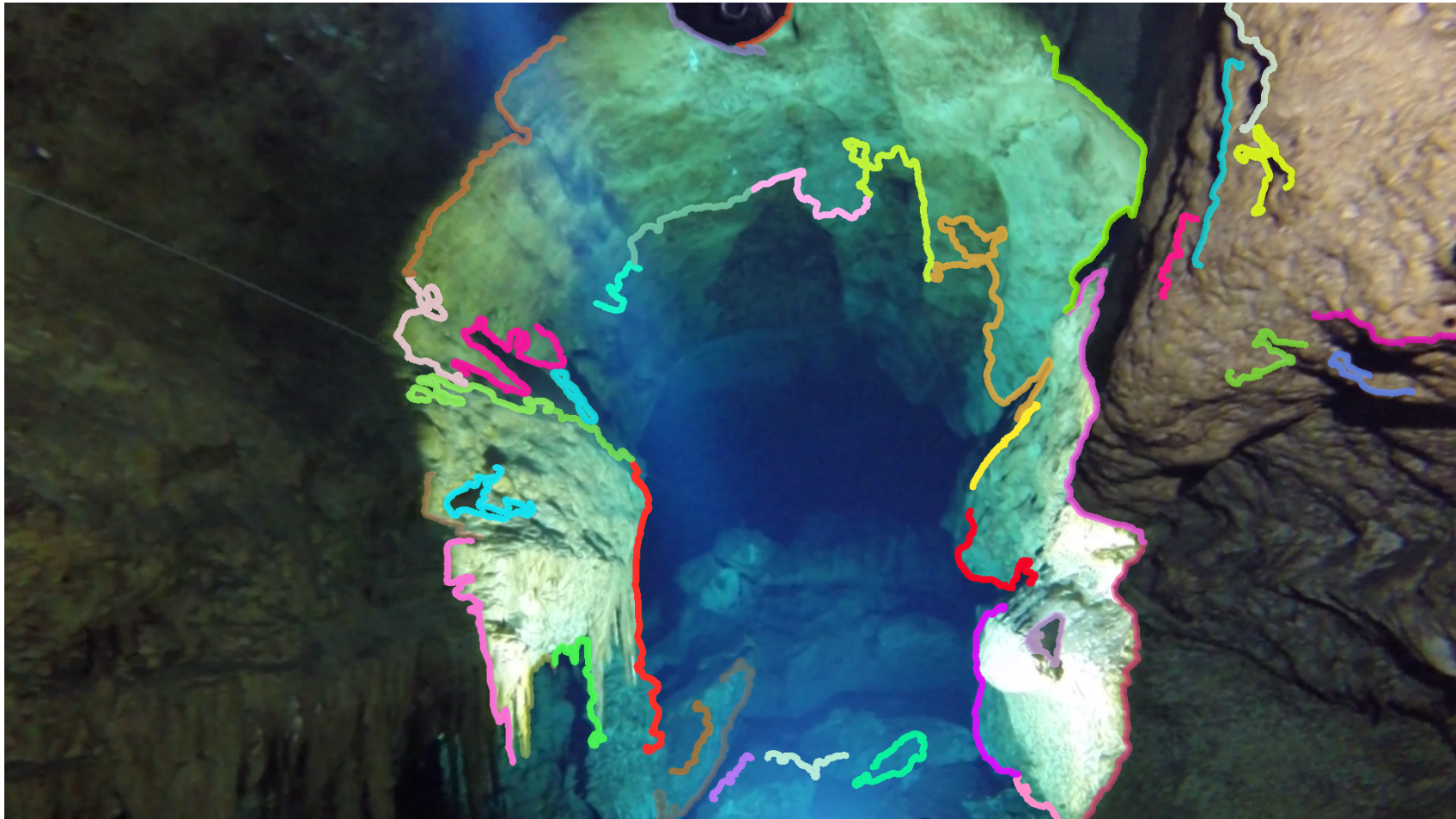
Image Downsampling



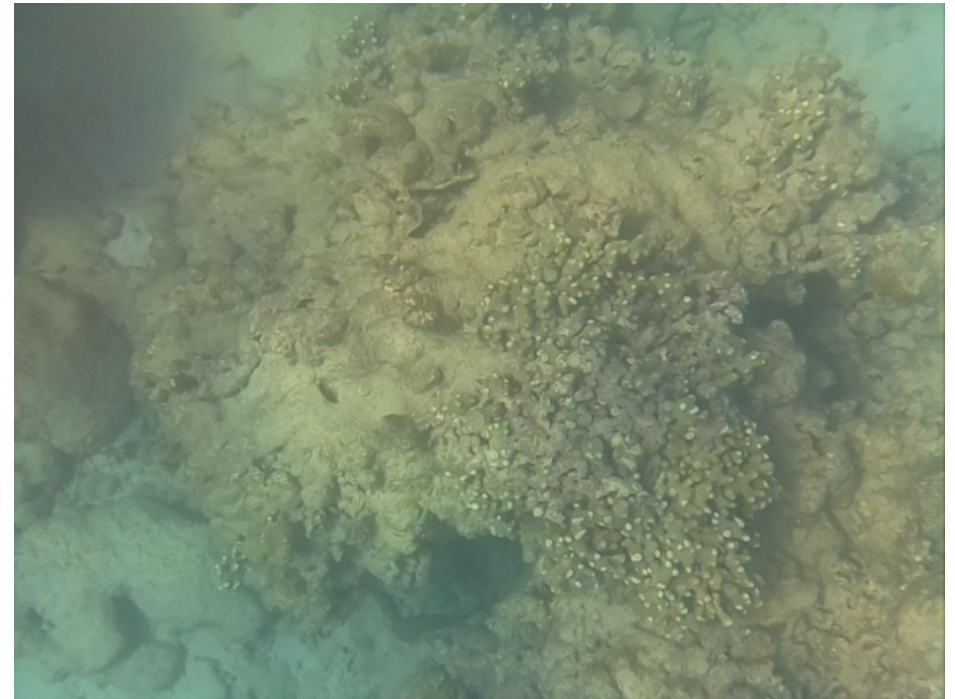
Thresholded image



Edge detection



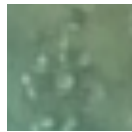
Correspondence Problem



From Raspberry PI camera at Barbados 2016 Field Trials

Correspondence

From I_1



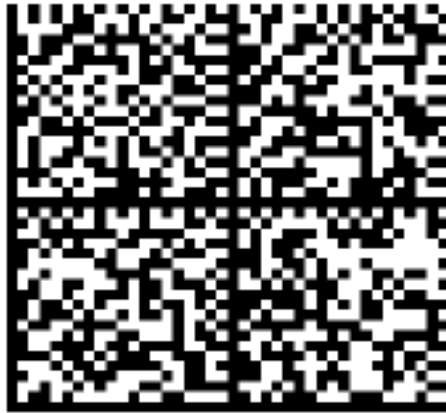
From I_2



Fiduciary Markers/Fiducial



(a) MaxiCode



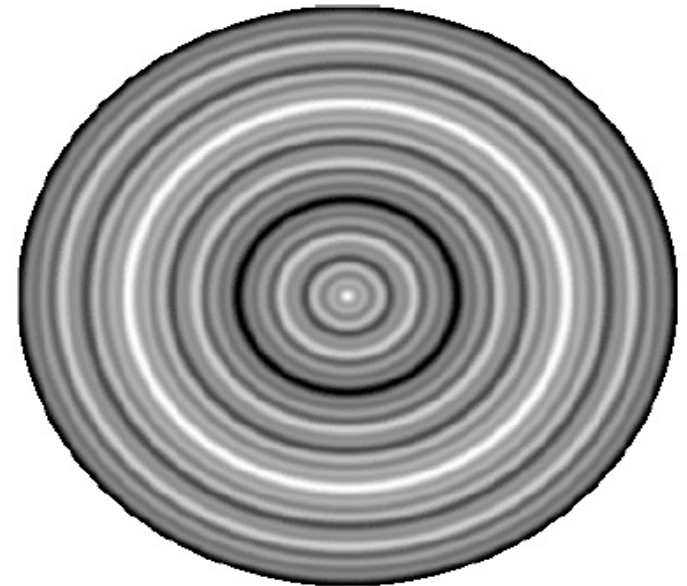
(b) DataMatrixSymbol



(c) ARToolkit



(d) ARTag



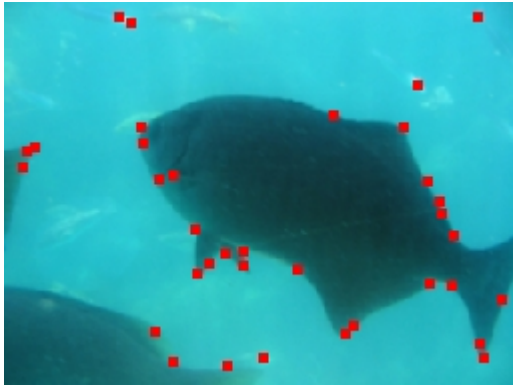
Fourier Tag

Good Feature

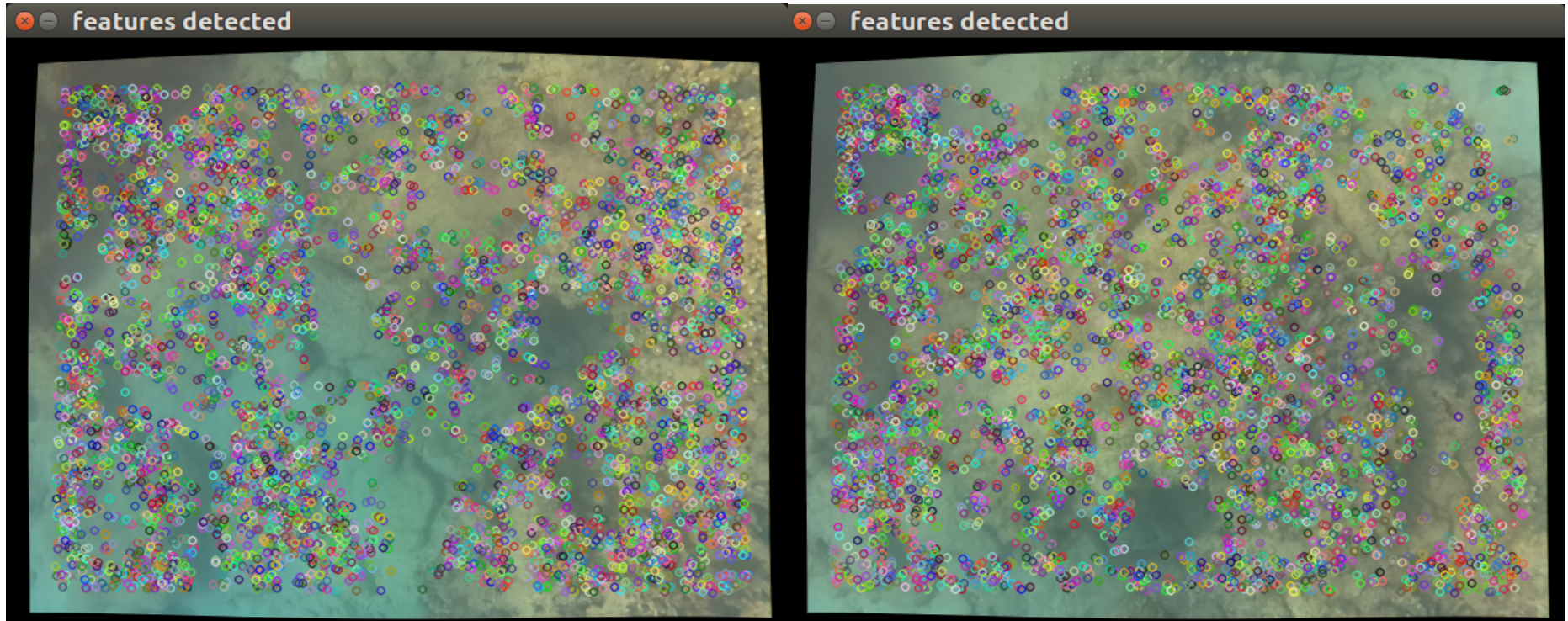
- Invariant to transformations
- Unique
- Efficient to compute
- Good precision and high recall
- Several Alternatives:
 - Harris Corners (OpenCV)
 - SURF (OpenCV)
 - SIFT
 - ORB
 - etc



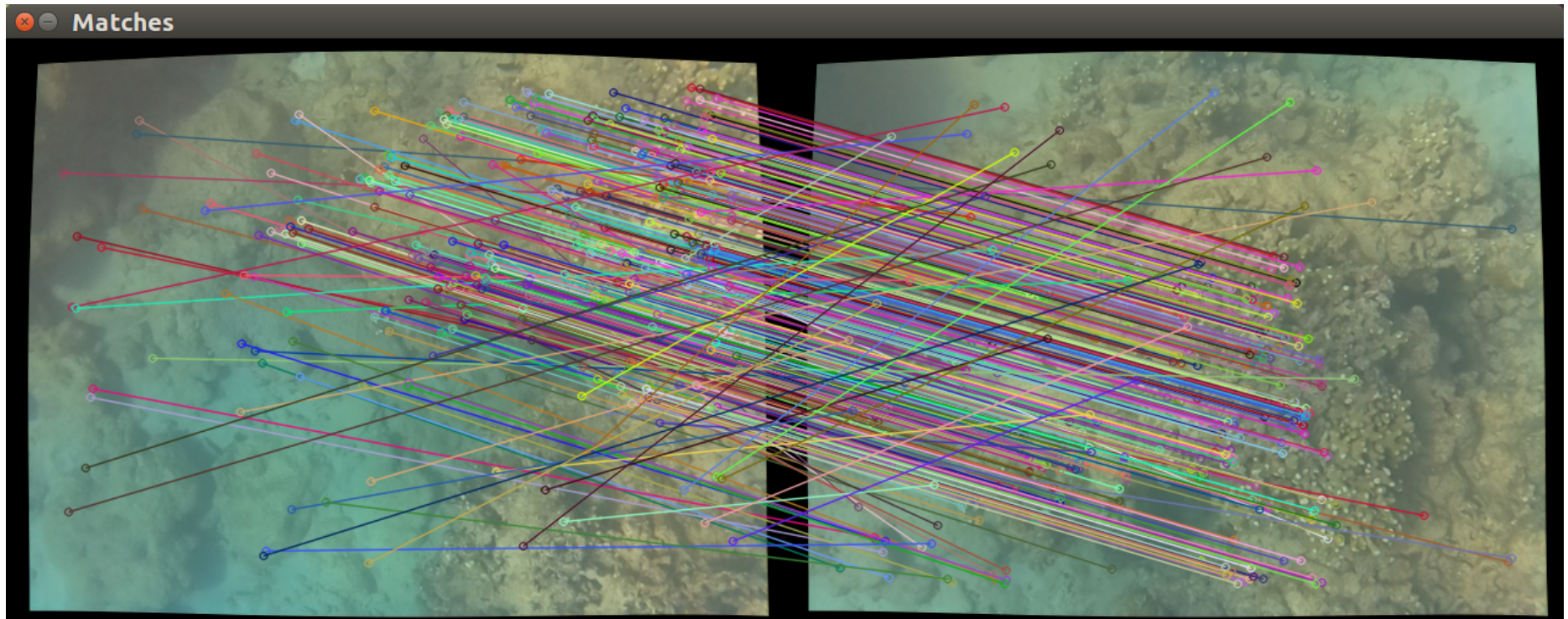
Harris Corners



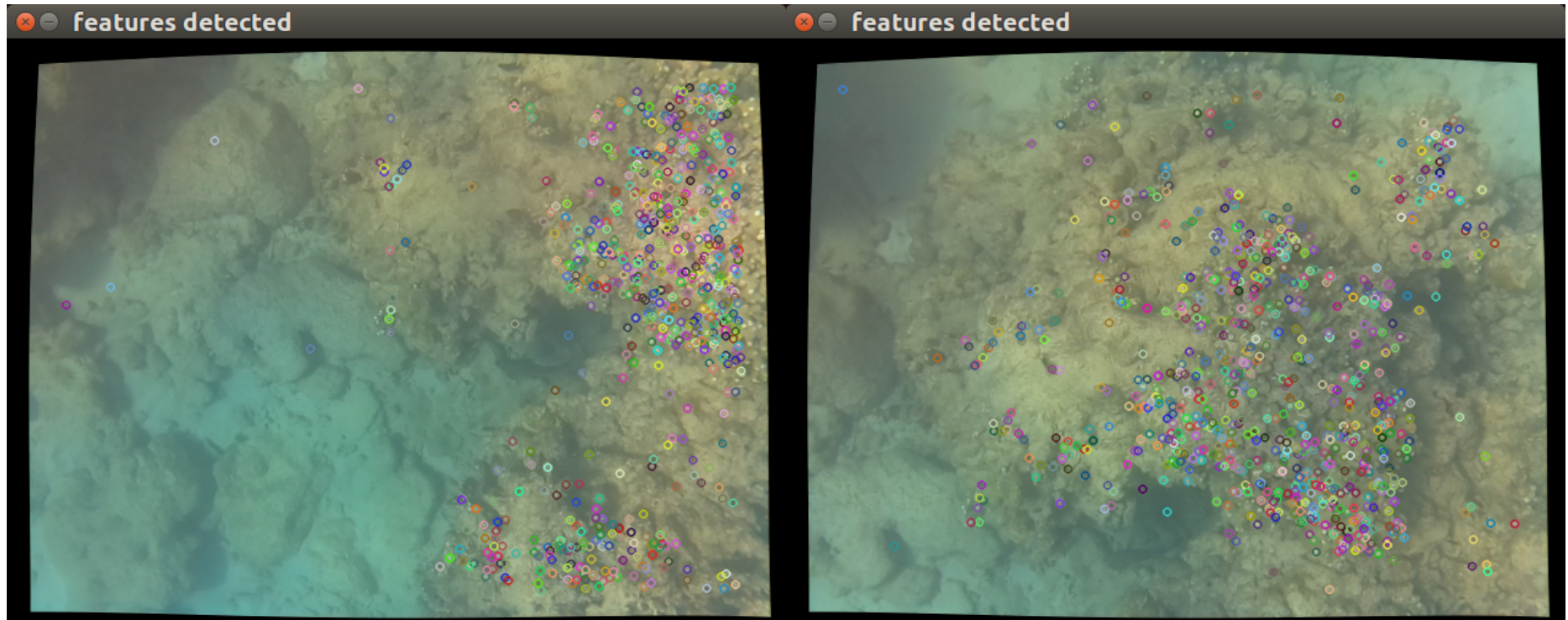
Harris Corners



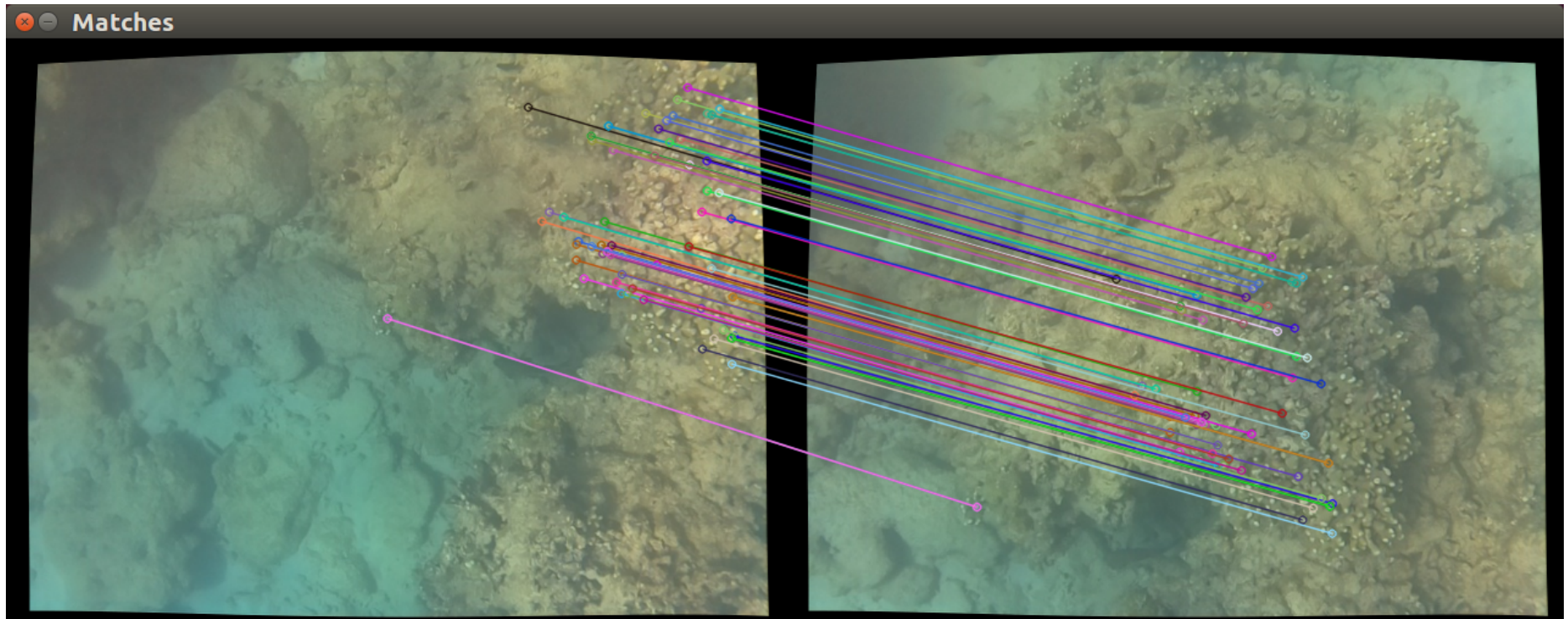
Harris Corners



SIFT



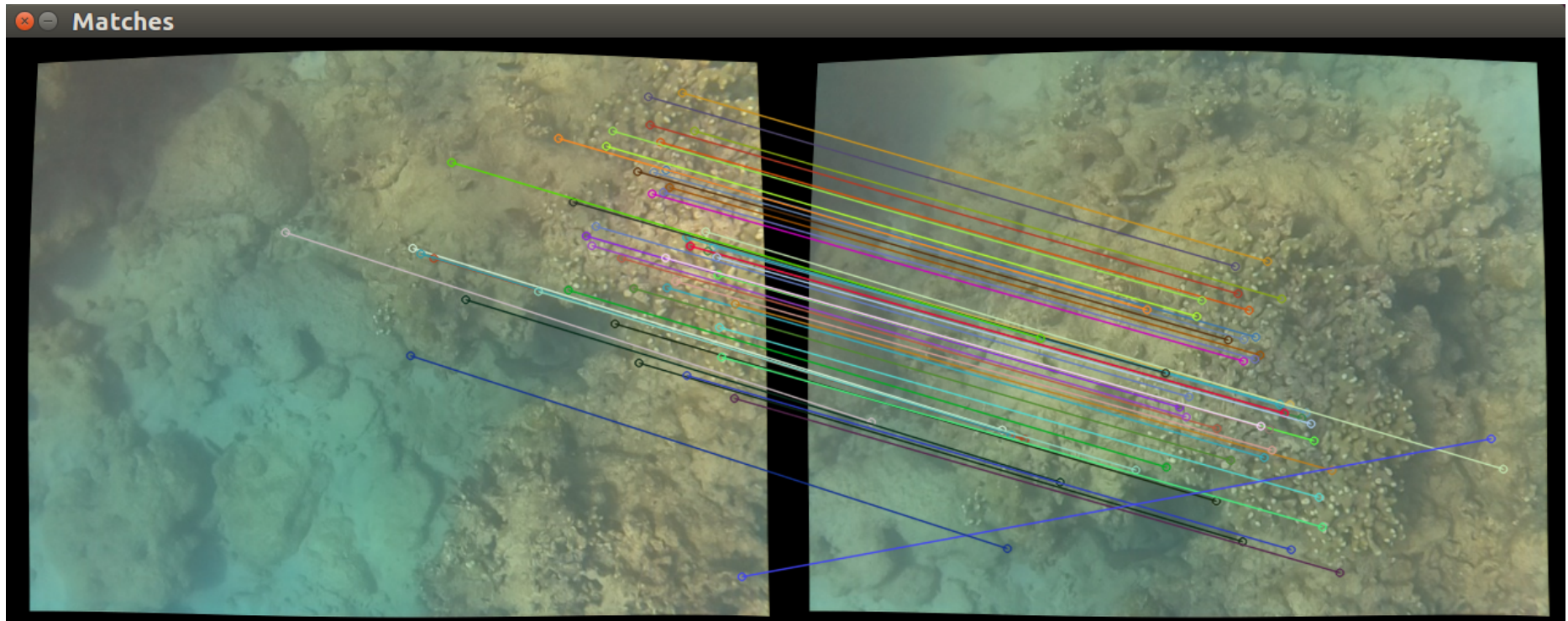
SIFT



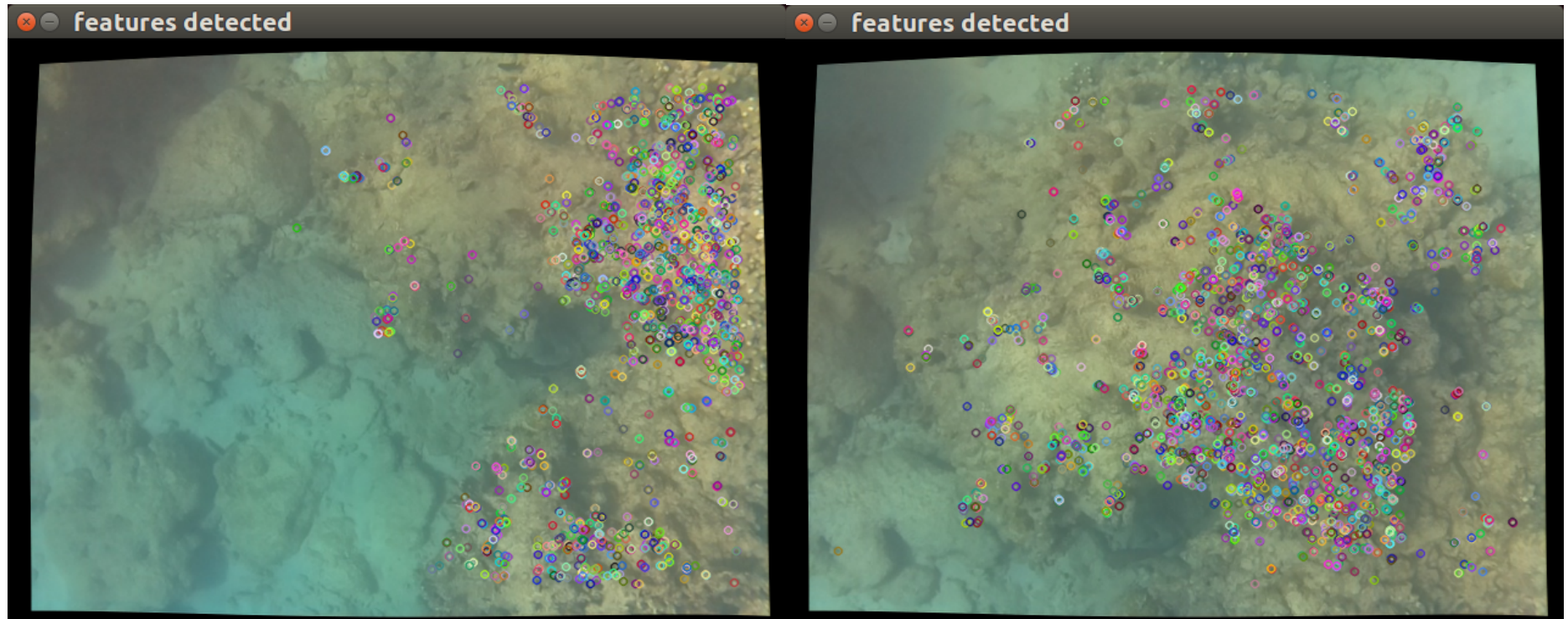
SURF



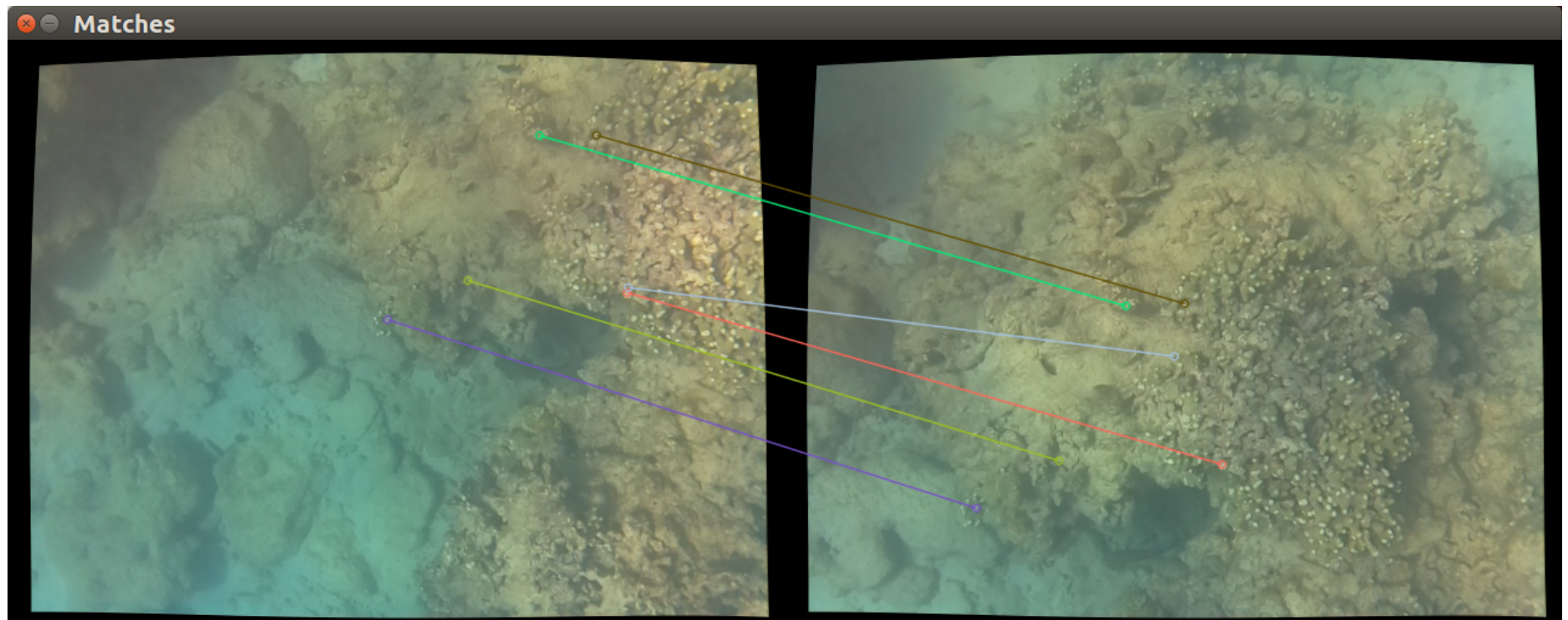
SURF



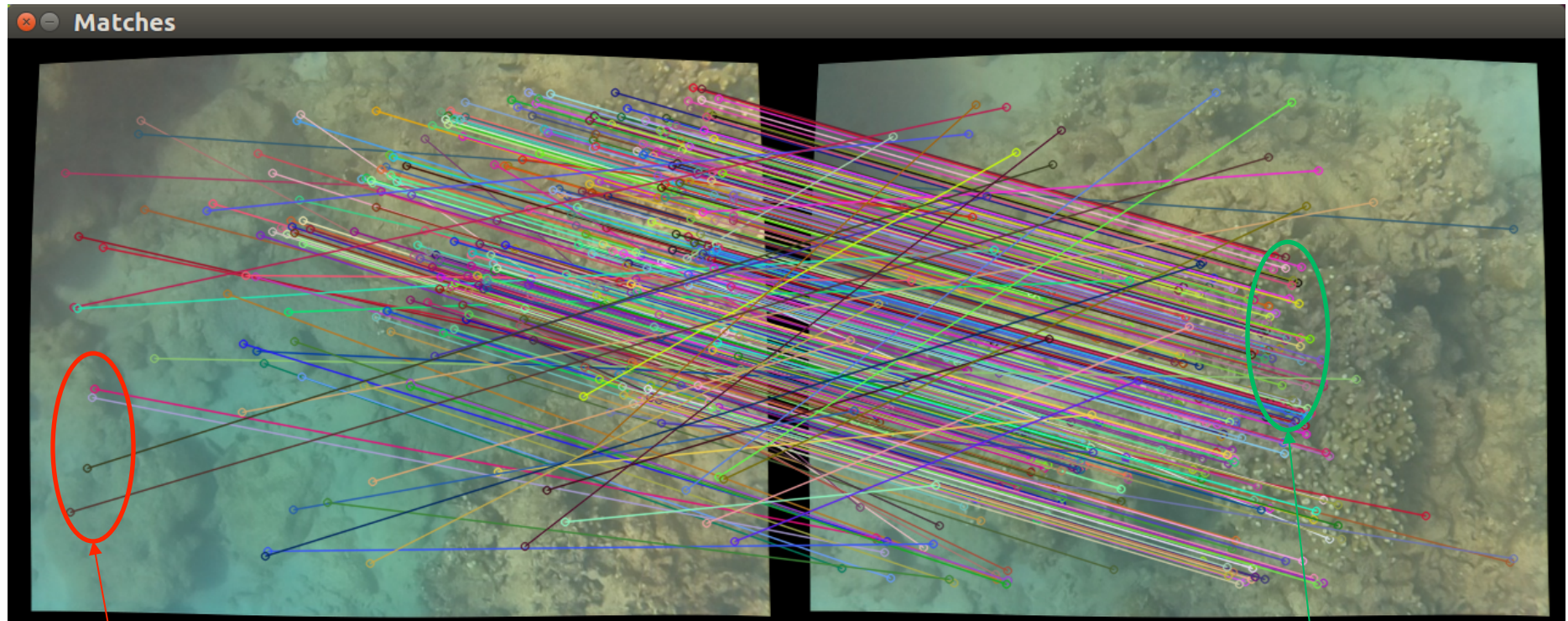
ORB



ORB



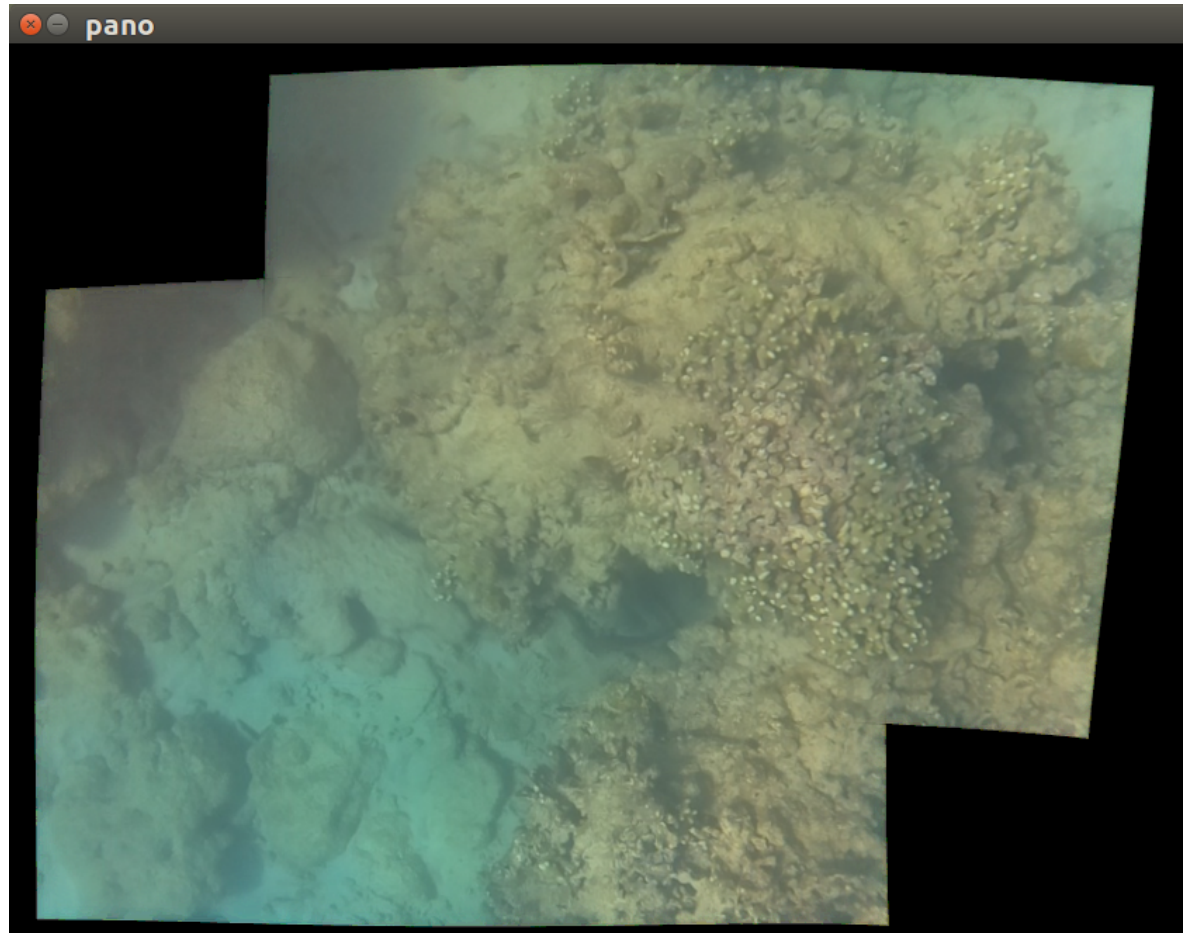
Outliers



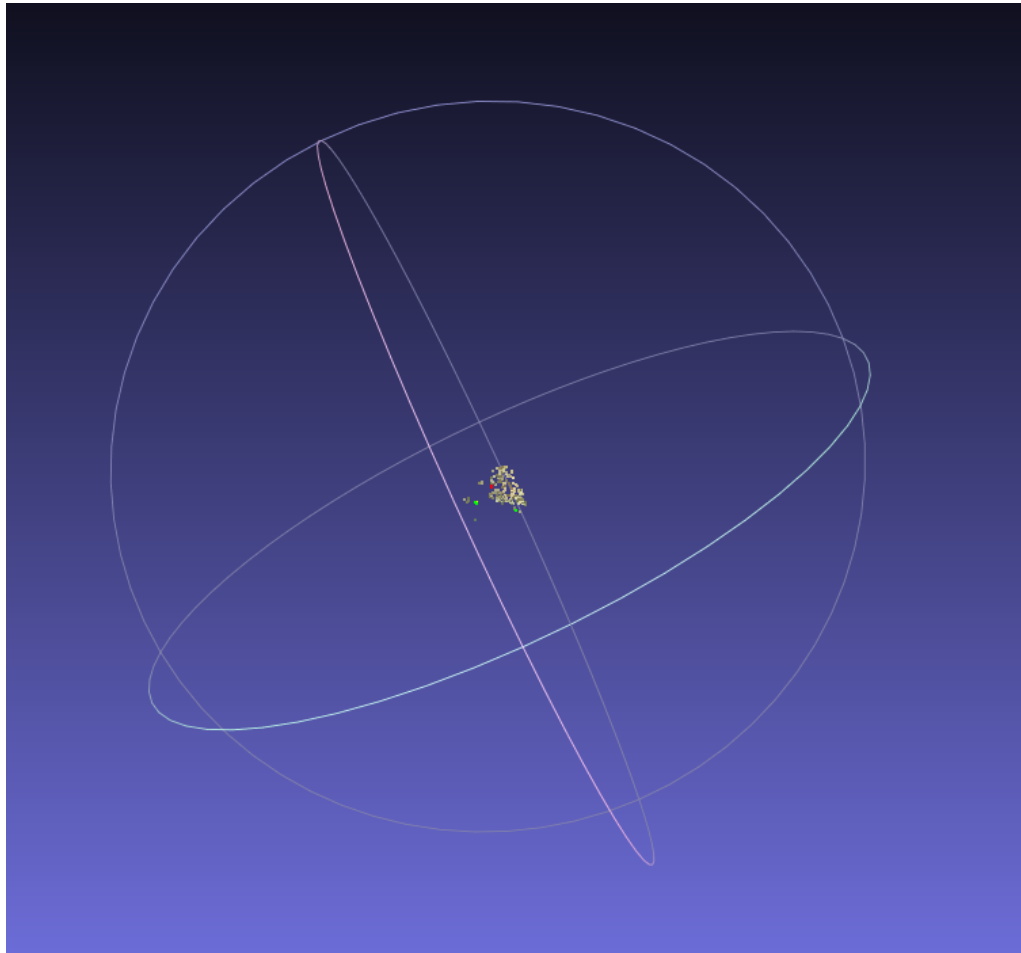
Outliers

Inliers

Mosaic



3D Sparse reconstruction

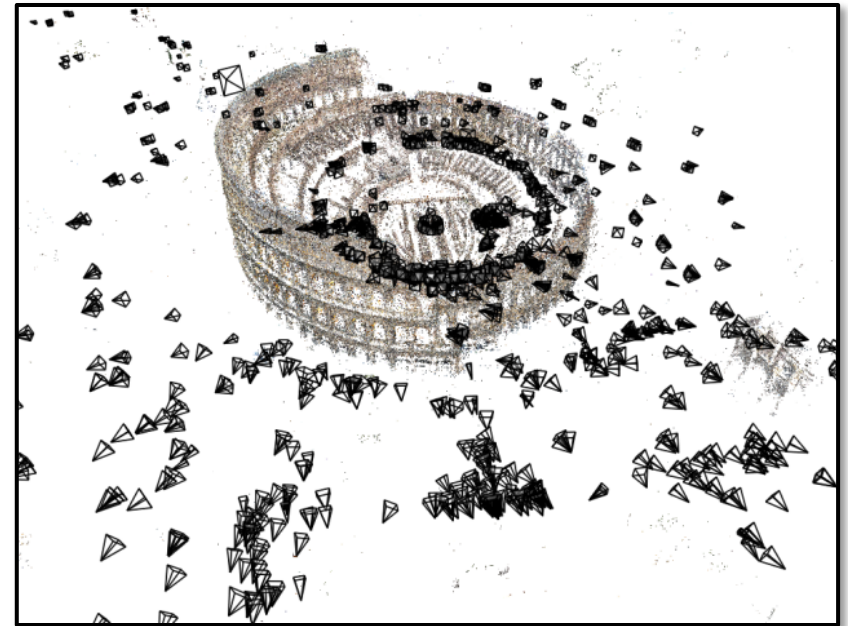
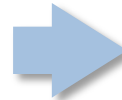


3D Sparse reconstruction

Source: <https://grail.cs.washington.edu/rome/>

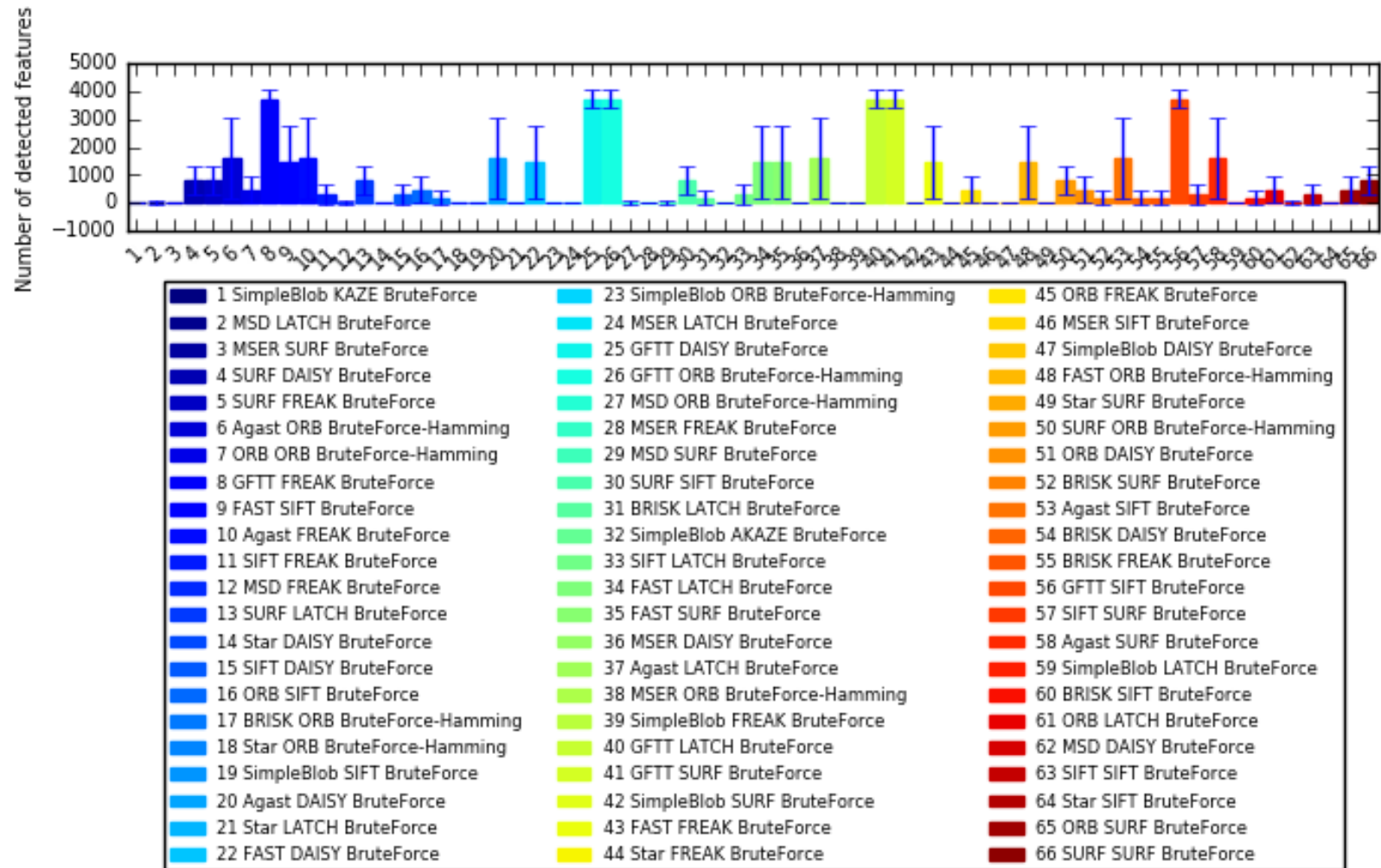


**Internet Photos
("Colosseum")**

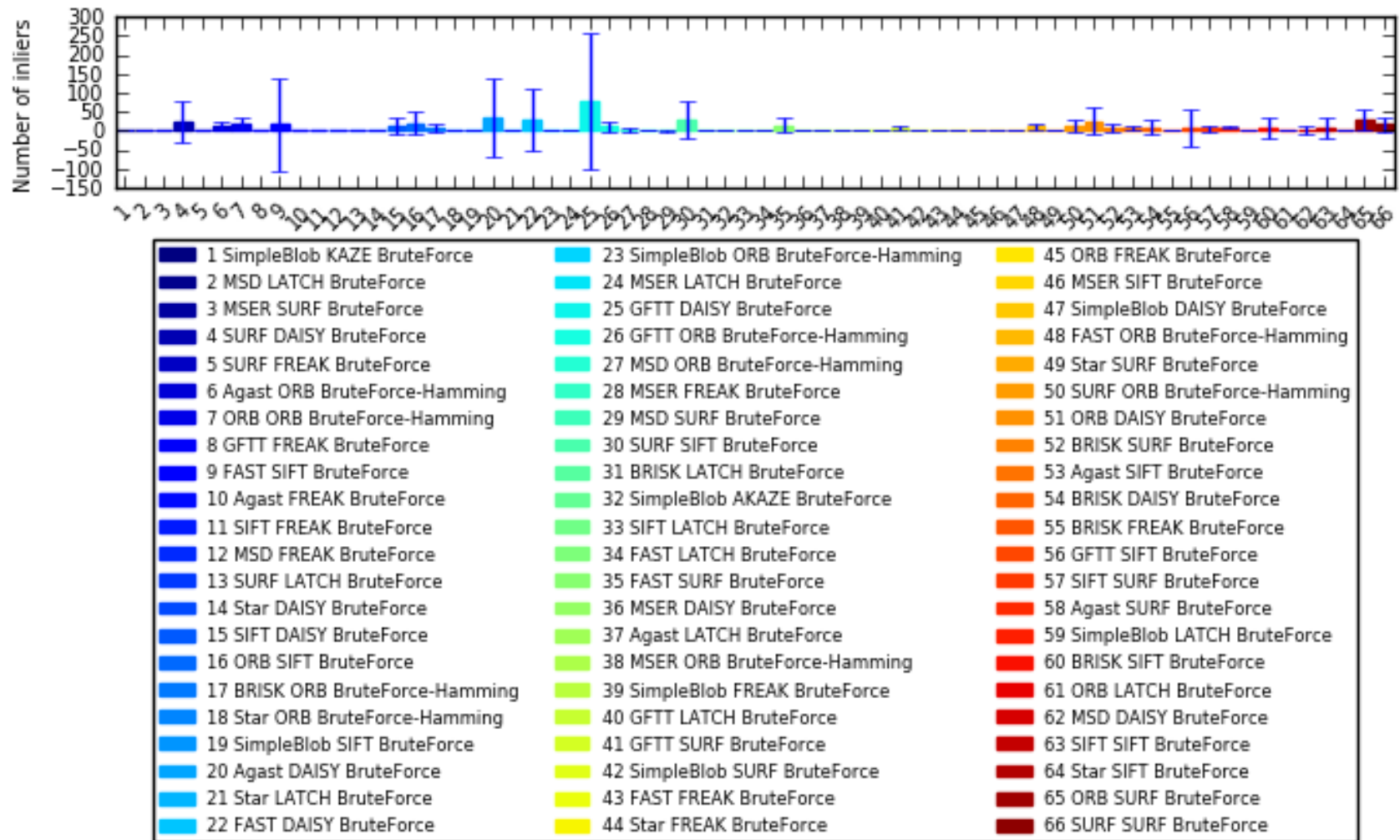


**Reconstructed 3D cameras and
points**

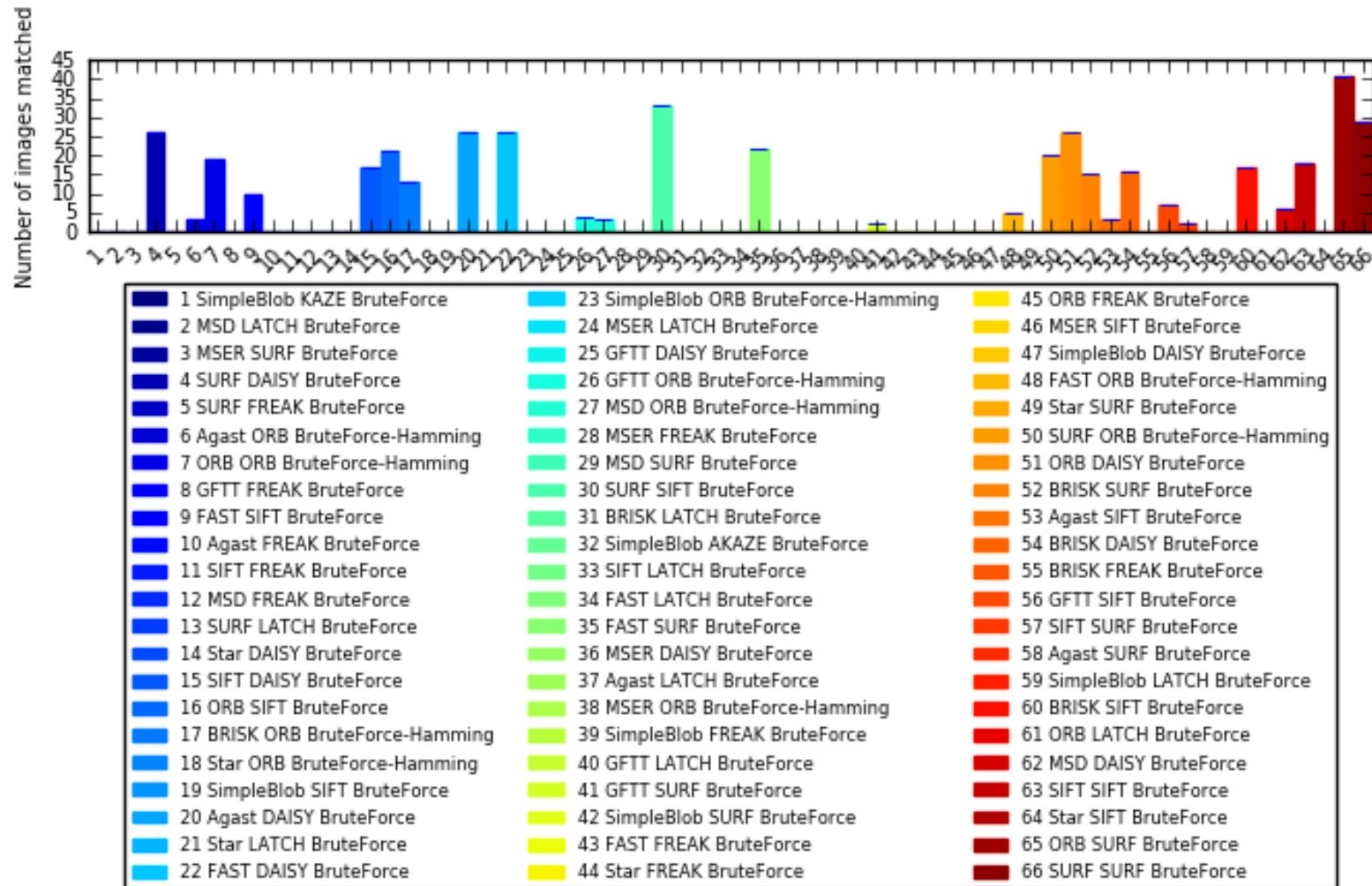
Feature quality



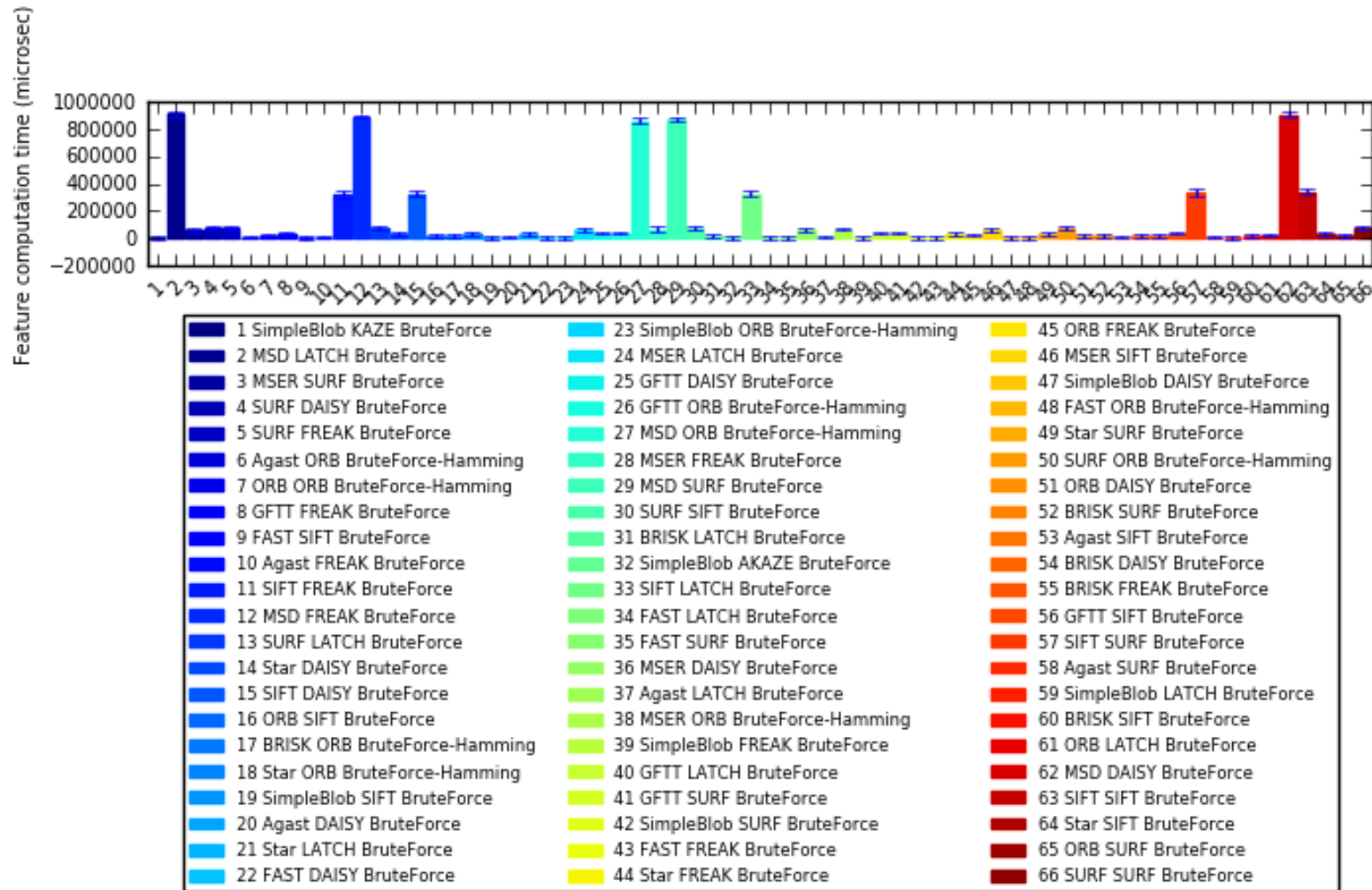
Feature quality



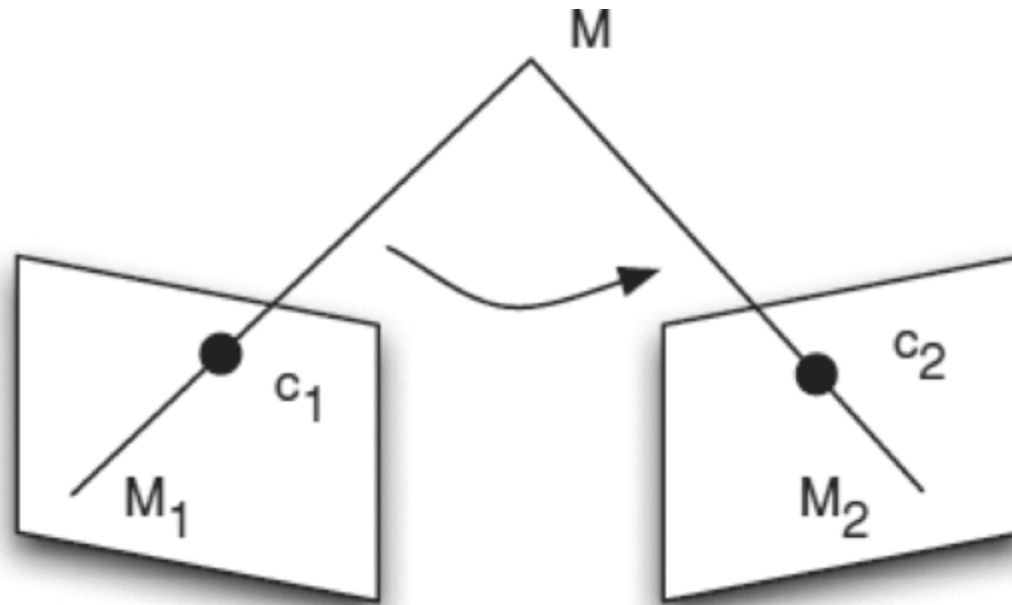
Feature quality



Feature quality

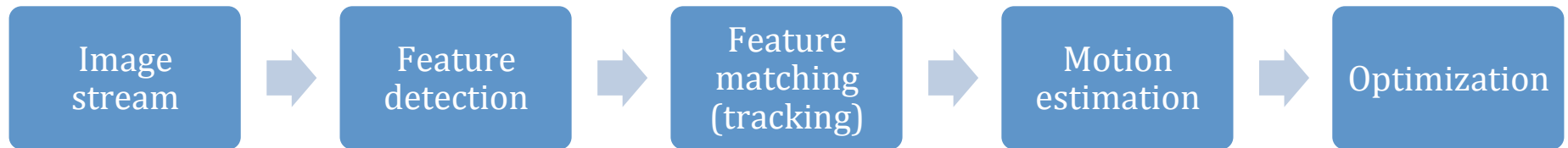


Egomotion



$$C_1 M_1 (T \times R C_2 M_2) = 0$$

Visual Odometry/Structure from Motion

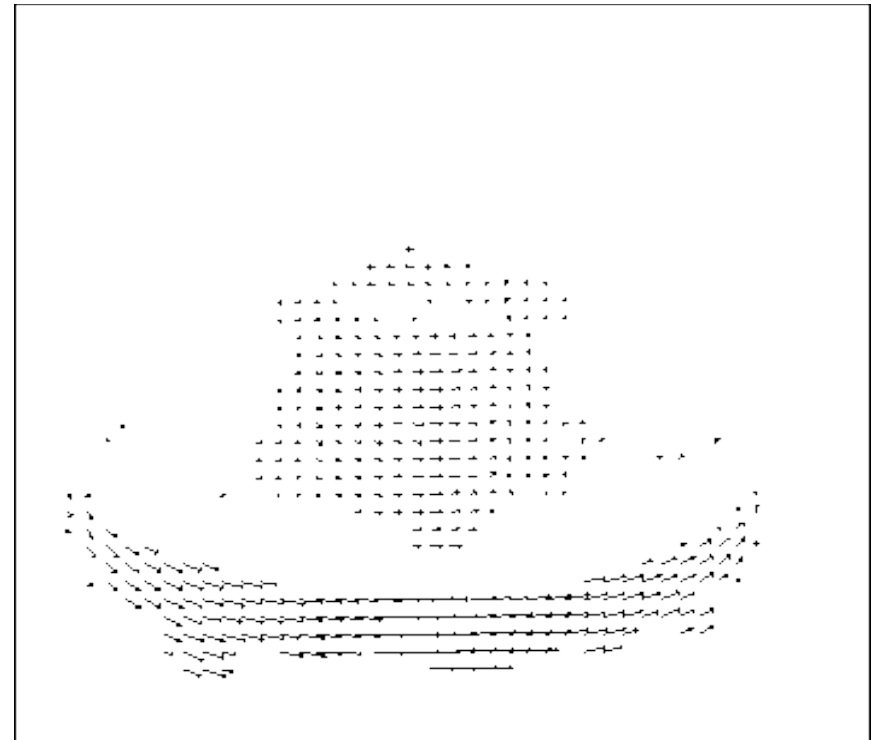


Optical Flow

- Definition:
 - *the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer (an eye or a camera) and the scene.*



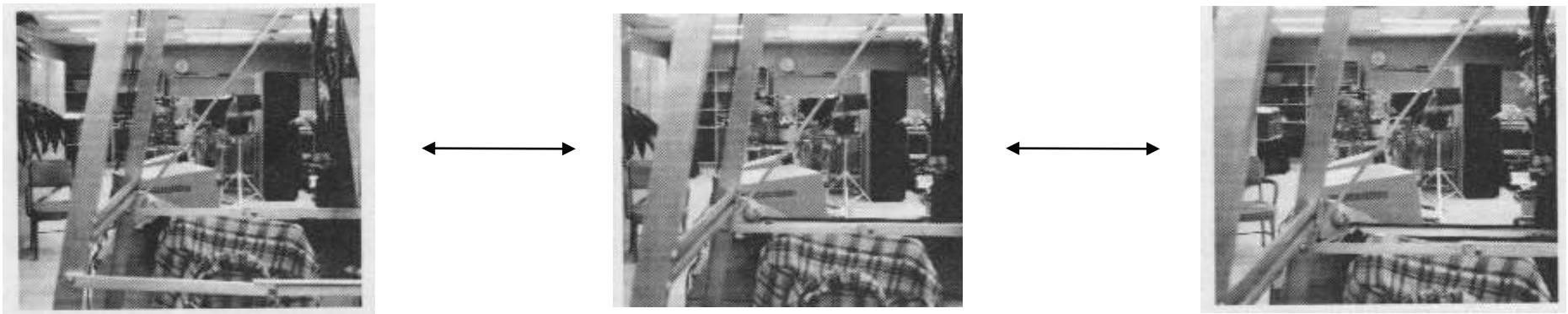
Optical Flow Field



Optical flow

Information about *image motion* rather than the *scene*.
*This is a classic **reconstruction** problem.*

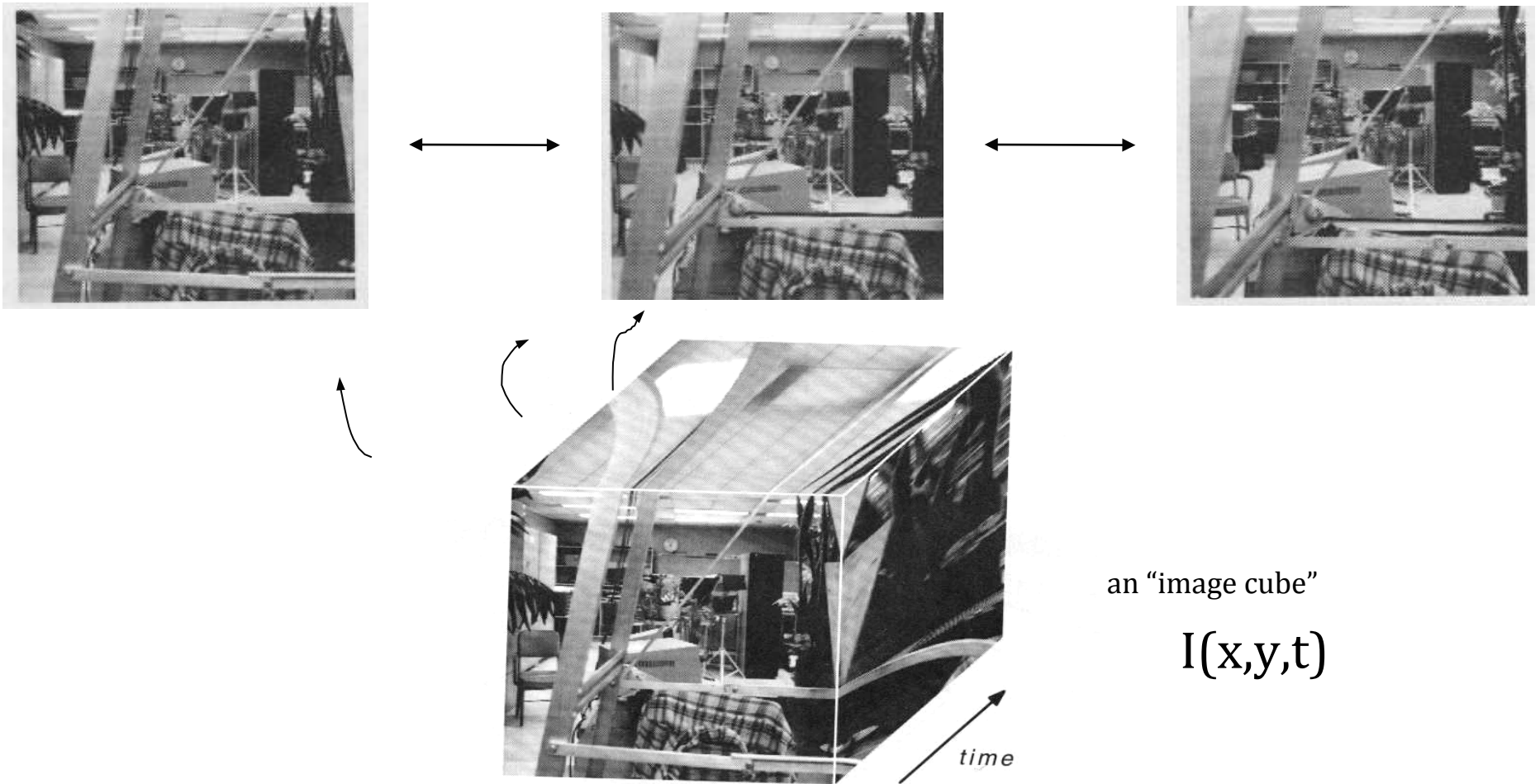
This next step might be to use the image motion to infer scene motion, robot motion or 3D layout.



time sequence of images

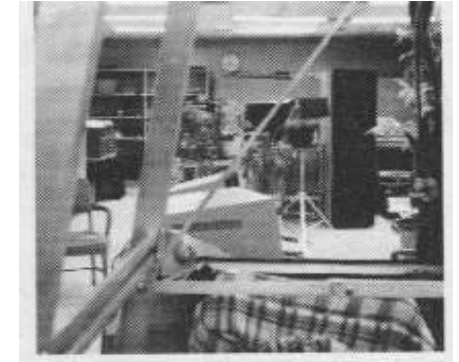
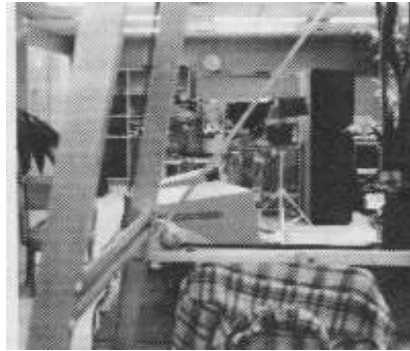
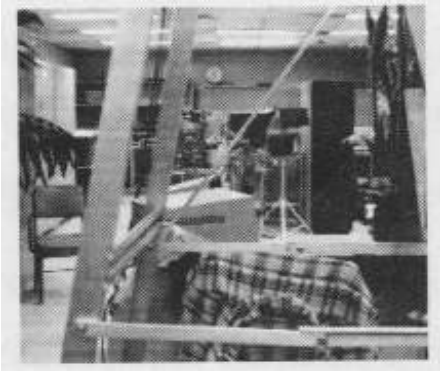
Optical flow

Information about *scene motion* rather than the *scene*.



Optical flow

Information about *scene motion* rather than the *scene*.



optical flow

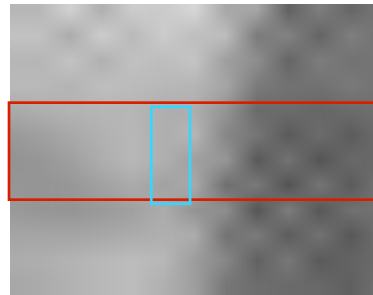
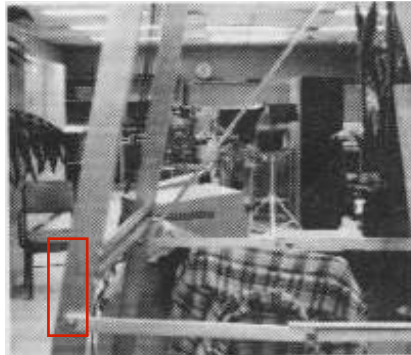
How ?



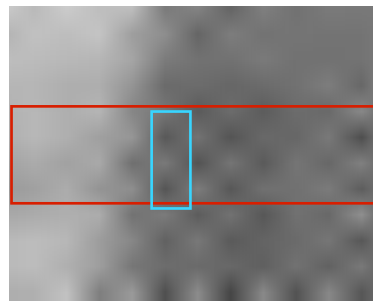
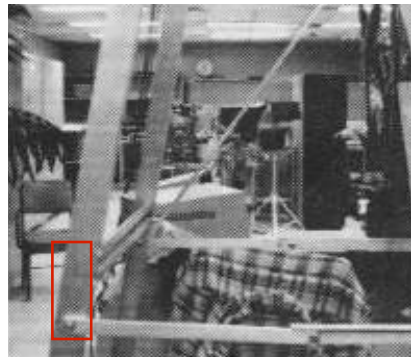
Optical Flow

- By measuring the direction that intensities are moving...

$I(x,y,t)$



99	90	90	70	40
95	90	70	40	40
90	90	70	40	40
90	90	70	40	40
90	70	50	40	30

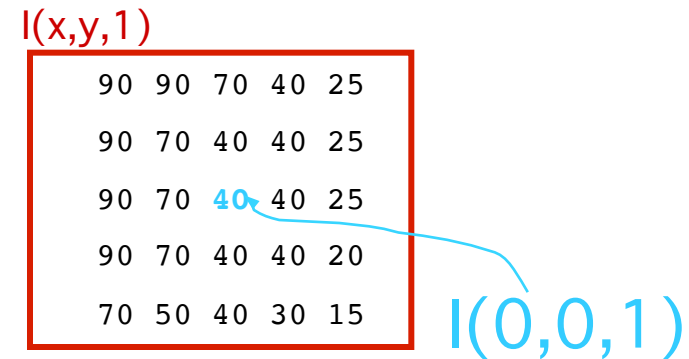
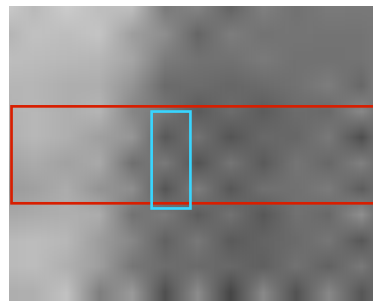
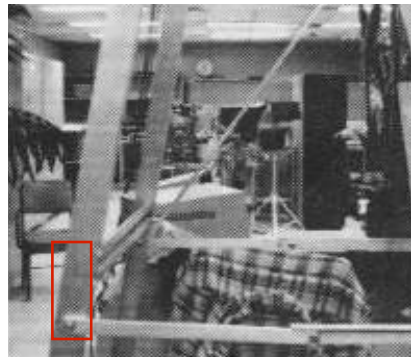
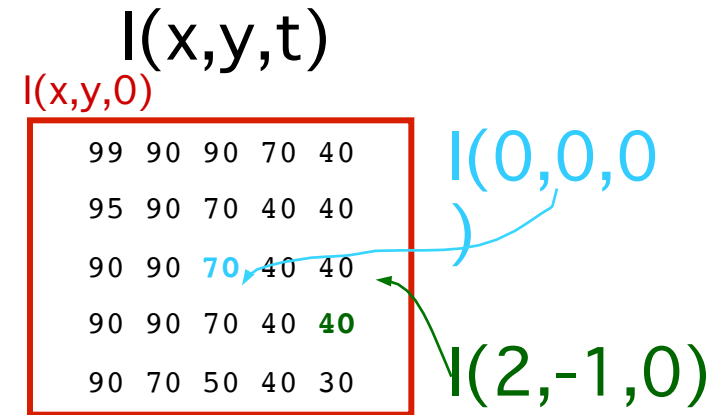
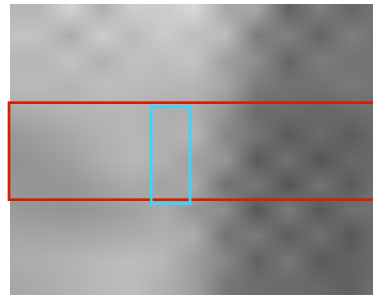
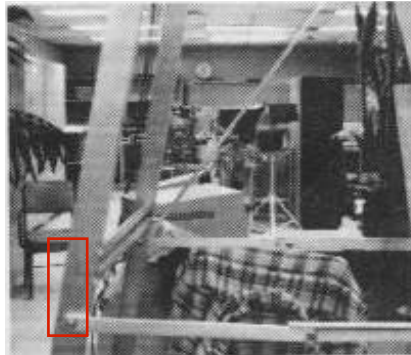


90	90	70	40	25
90	70	40	40	25
90	70	40	40	25
90	70	40	40	20
70	50	40	30	15

- We can estimate things...

Optical Flow

By measuring the direction that intensities are moving...



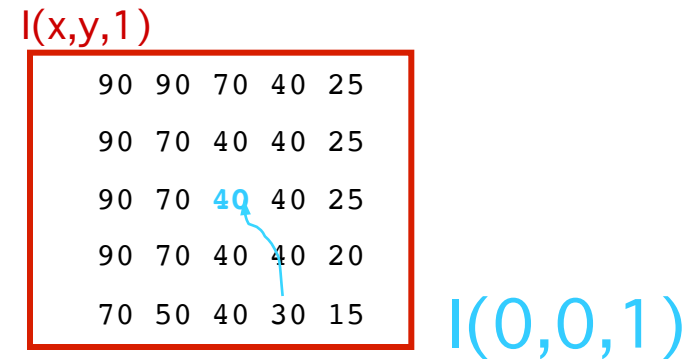
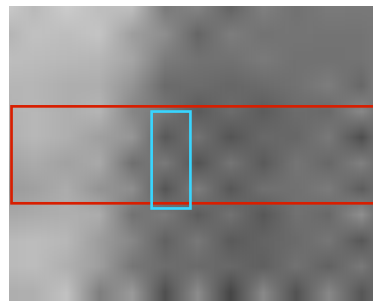
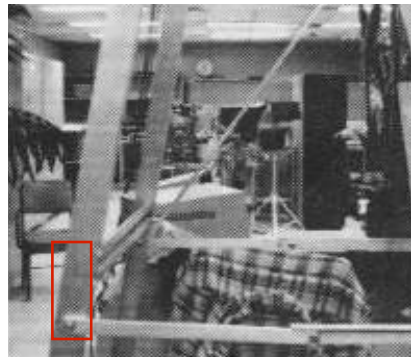
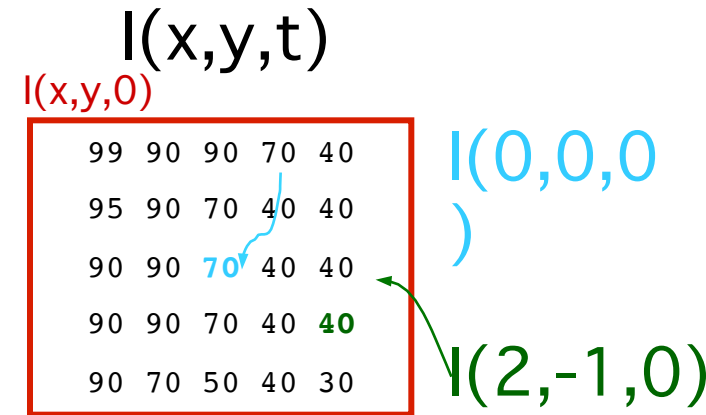
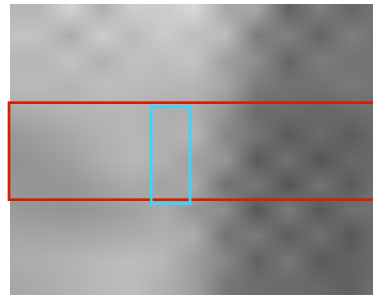
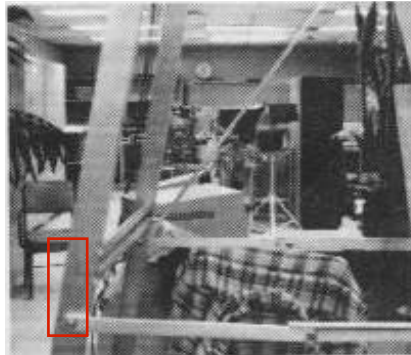
We can estimate things ...

$$\frac{dI}{dx} = I_x \text{ at } (0,0,0)$$



Optical Flow

By measuring the direction that intensities are moving...



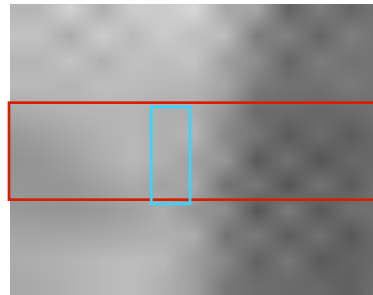
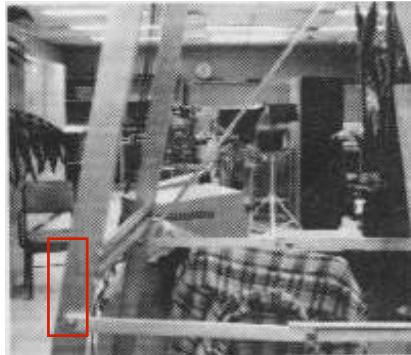
We can estimate things like

$$\frac{dI}{dx} = I_x \text{ at } (0,0,0) = \frac{\Delta I}{\Delta x} = \frac{I(1,0,0) - I(0,0,0)}{1 - 0} = -30$$



Optical Flow

By measuring the direction that intensities are moving...

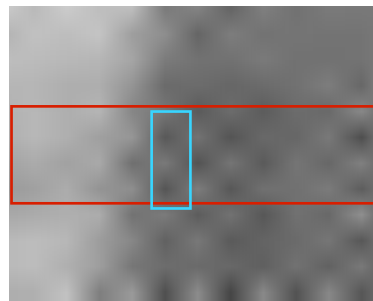
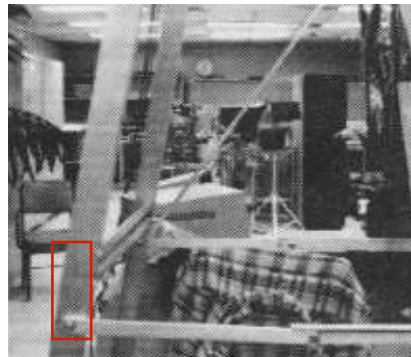


$I(x,y,t)$
 $I(x,y,0)$

99	90	90	70	40
95	90	70	40	40
90	90	70	40	40
90	90	70	40	40
90	70	50	40	30

$I(0,0,0)$

$I(2,-1,0)$



$I(x,y,1)$

90	90	70	40	25
90	70	40	40	25
90	70	40	40	25
90	70	40	40	20
70	50	40	30	15

$I(0,0,1)$

We can estimate things like

$$\frac{dI}{dx} = I_x$$

$$\frac{dI}{dy} = I_y$$

$$\frac{dI}{dt} = I_t$$

SO...



Measuring Optical Flow

Let $I(x,y,t)$ be the sequence of images.

Simplest assumption (constant brightness constraint):

$$I(x,y,t) = I(x + dx, y + dy, t + dt)$$

(x,y,t)

99	90	90	70	40
95	90	70	40	40
90	90	70	40	40
90	90	70	40	40
90	70	50	40	30



Measuring Optical Flow

Let $I(x,y,t)$ be the sequence of images.

Simplest assumption (constant brightness constraint):

$$I(x,y,t) = I(x + dx, y + dy, t + dt)$$

(x,y,t)

99	90	90	70	40
95	90	70	40	40
90	90	70	40	40
90	90	70	40	40
90	70	50	40	30

Reminder: $f(x + dx) = f(x) + f'(x) dx + f''(x) dx^2 / 2 + \dots$

Measuring Optical Flow

Let $I(x,y,t)$ be the sequence of images.

Simplest assumption (constant brightness constraint):

$$I(x,y,t) = I(x + dx, y + dy, t + dt)$$

(x,y,t)

99	90	90	70	40
95	90	70	40	40
90	90	70	40	40
90	90	70	40	40
90	70	50	40	30

Reminder: $f(x + dx) = f(x) + f'(x) dx + f''(x) dx^2 / 2 + \dots$

$$I(x,y,t) = I(x,y,t) + I_x dx + I_y dy + I_t dt + \text{2nd deriv.} + \text{higher}$$



Measuring Optical Flow

Let $I(x,y,t)$ be the sequence of images.

Simplest assumption (constant brightness constraint):

$$I(x,y,t) = I(x + dx, y + dy, t + dt)$$

(x,y,t)

99	90	90	70	40
95	90	70	40	40
90	90	70	40	40
90	90	70	40	40
90	70	50	40	30

Reminder: $f(x + dx) = f(x) + f'(x) dx + f''(x) dx^2 / 2 + \dots$

$$I(x,y,t) = I(x,y,t) + I_x dx + I_y dy + I_t dt + \text{2nd deriv. + higher}$$

$$0 = I_x dx + I_y dy + I_t dt$$

ignore these terms



Measuring Optical Flow

Let $I(x,y,t)$ be the sequence of images.

Simplest assumption (constant brightness constraint):

$$I(x,y,t) = I(x + dx, y + dy, t + dt)$$

(x,y,t)

99	90	90	70	40
95	90	70	40	40
90	90	70	40	40
90	90	70	40	40
90	70	50	40	30

Reminder: $f(x + dx) = f(x) + f'(x) dx + f''(x) dx^2 / 2 + \dots$

$$I(x,y,t) = I(x,y,t) + I_x dx + I_y dy + I_t dt + \text{2nd deriv. + higher}$$

$$0 = I_x dx + I_y dy + I_t dt$$

ignore these terms

$$-I_t = I_x \frac{dx}{dt} + I_y \frac{dy}{dt}$$

intensity-flow equation

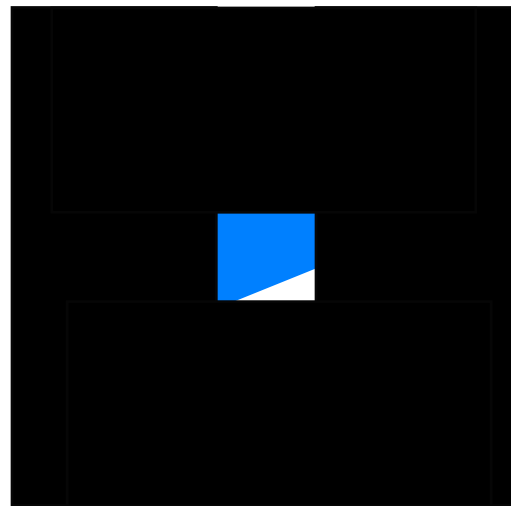
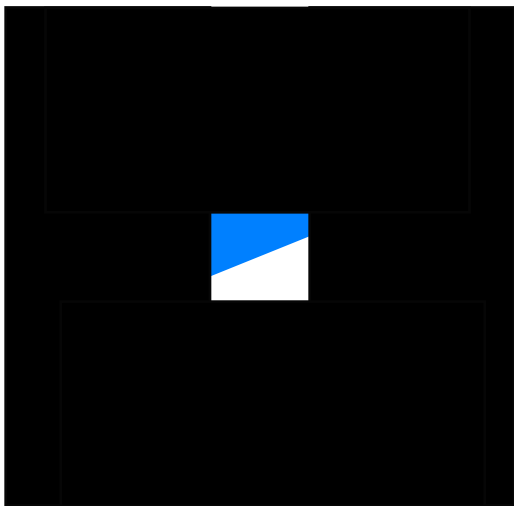
good and bad...



The “aperture” problem

$$-I_t = I_x \frac{dx}{dt} + I_y \frac{dy}{dt}$$

- The intensity-flow equation provides only one constraint on *two* variables (x-motion and y-motion)
- It is only possible to find optical flow in one direction...



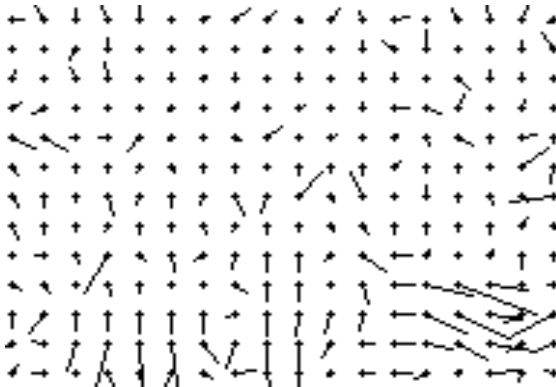
The “aperture” problem

- It is only possible to find optical flow in one direction...
*at any **single** point in the image !*

img1



raw
optical
flow



img2



smoothed
for ten
iterations



Smoothing can be done by incorporating neighboring points' information.

Observations & Warnings

- Assume the scene itself is static.
- Find matching chunks in the images.
- An instance of *correspondence*.

BUT

- World really isn't static.
- Lightning might change even in a static scene.



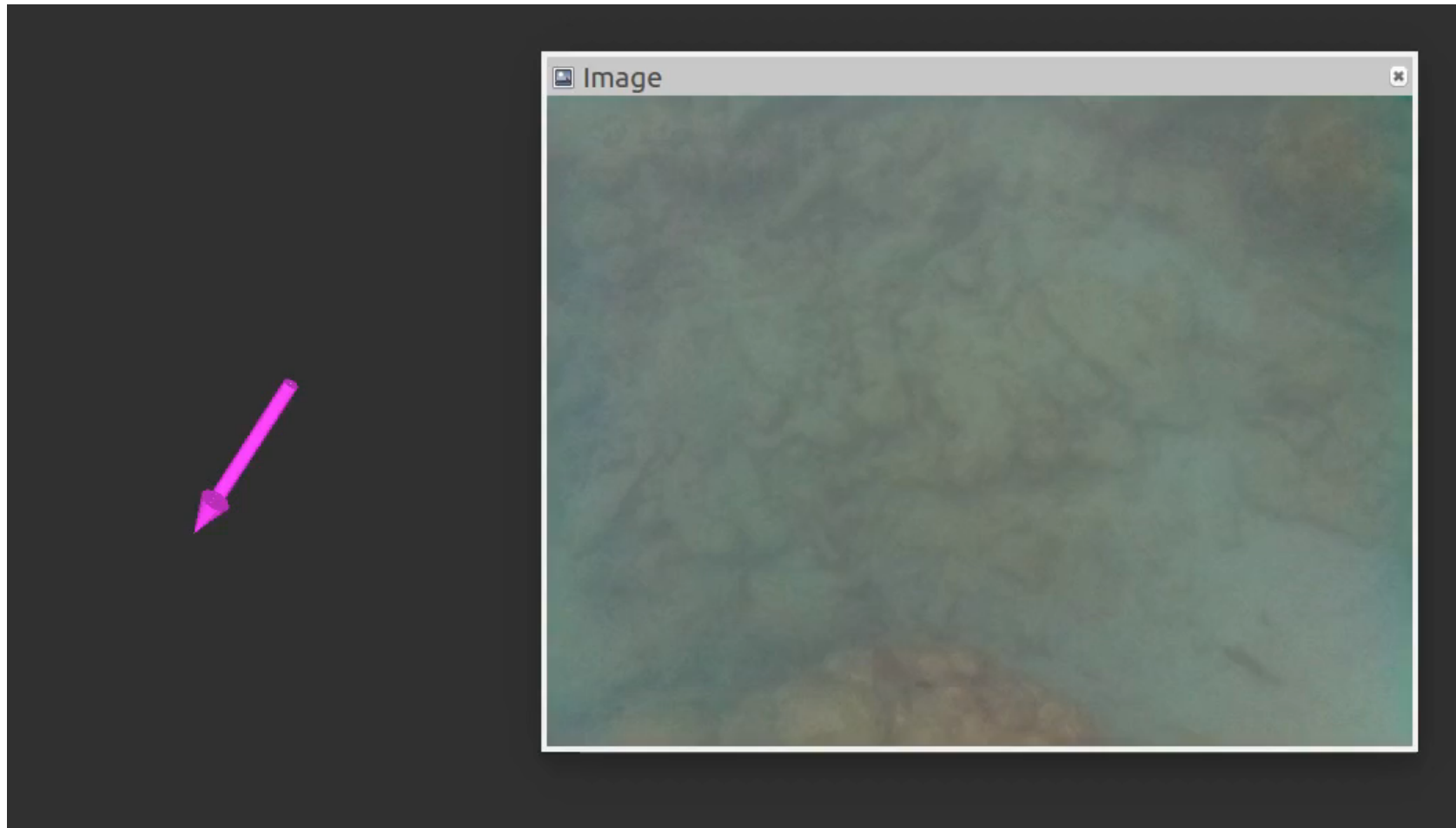
Features vs Optical Flow

- Feature-based methods
 - Detect features (corners, textured areas), extract descriptors, and track them
 - Sparse motion fields, but possibly robust tracking
 - Suitable especially when image motion is large (10s of pixels)
- Direct methods (optical flow)
 - Directly recover image motion from spatio-temporal image brightness variations
 - Global motion parameters directly recovered without an intermediate feature motion calculation
 - Dense motion fields, but more sensitive to appearance variations
 - Suitable for video and when image motion is small (< 10 pixels)



Camera and IMU

From drifter with Raspberry PI Camera and Pololu MinIMU-9 v3 at Barbados 2016 Field Trials

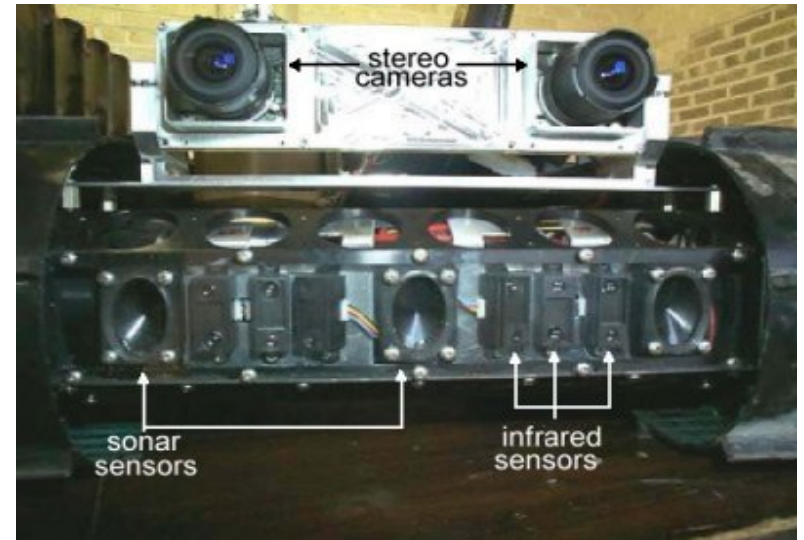


A Vision “solution”

- If interpreting a single image is difficult... What about more ?!

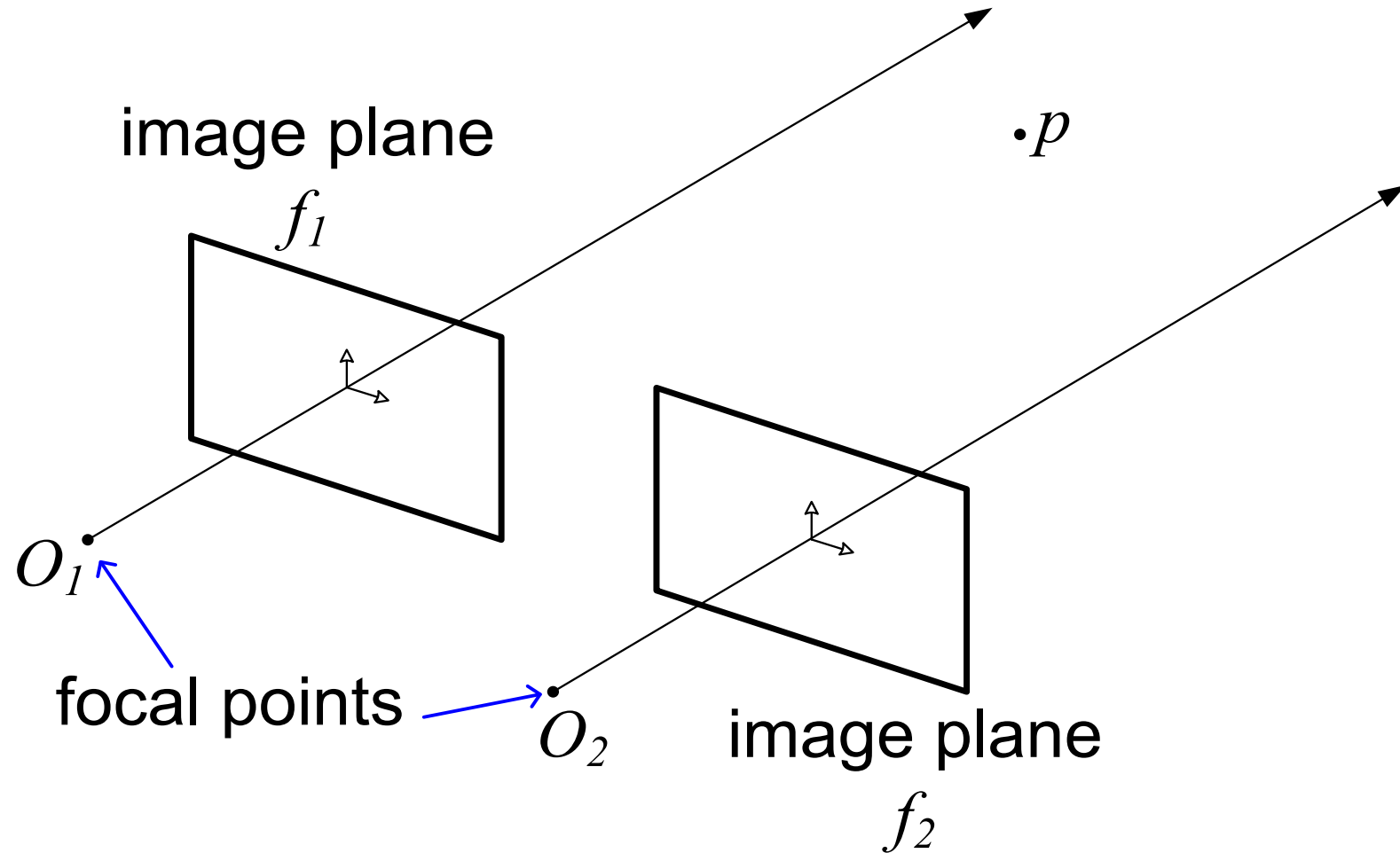


multiple cameras

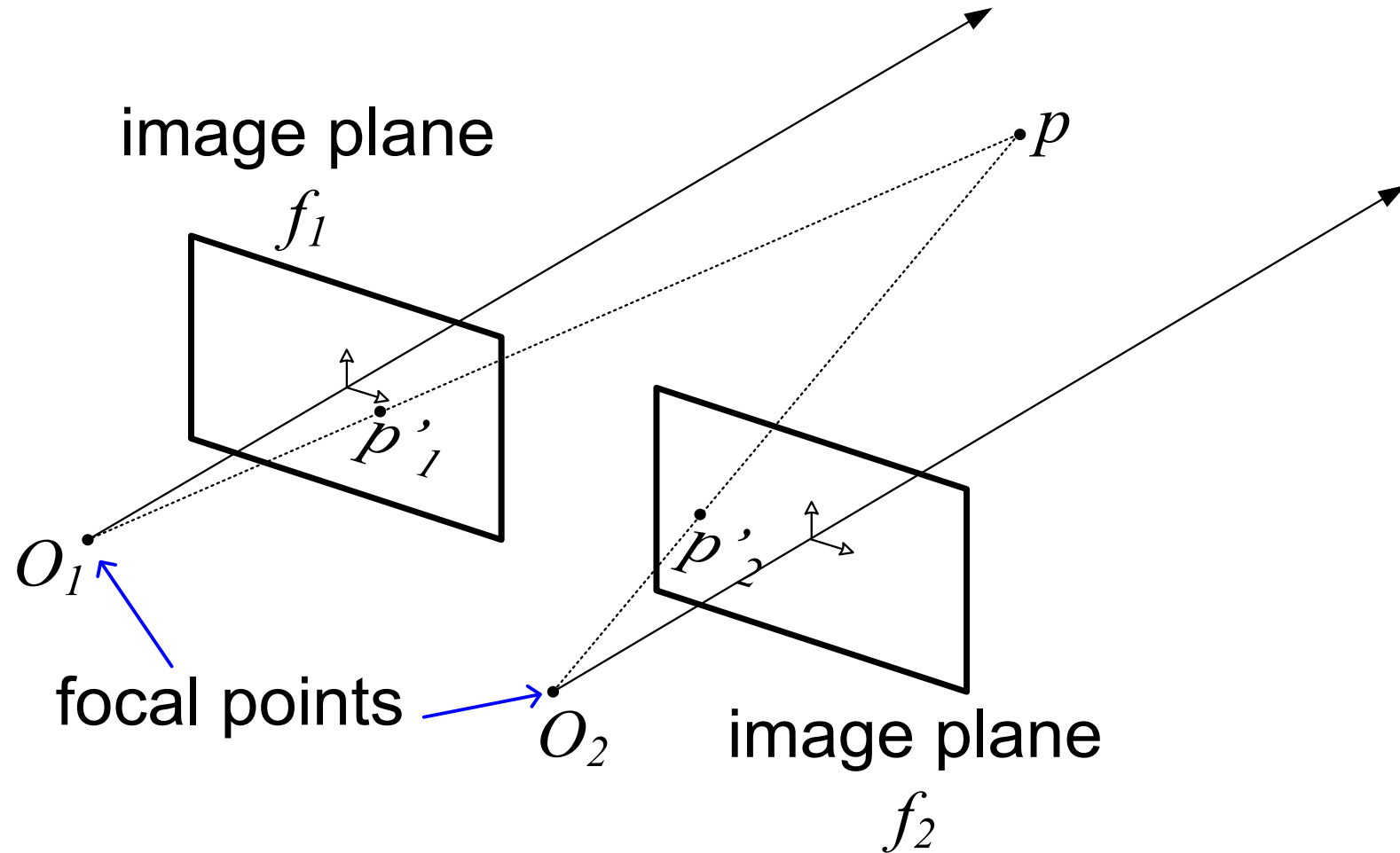


multiple times

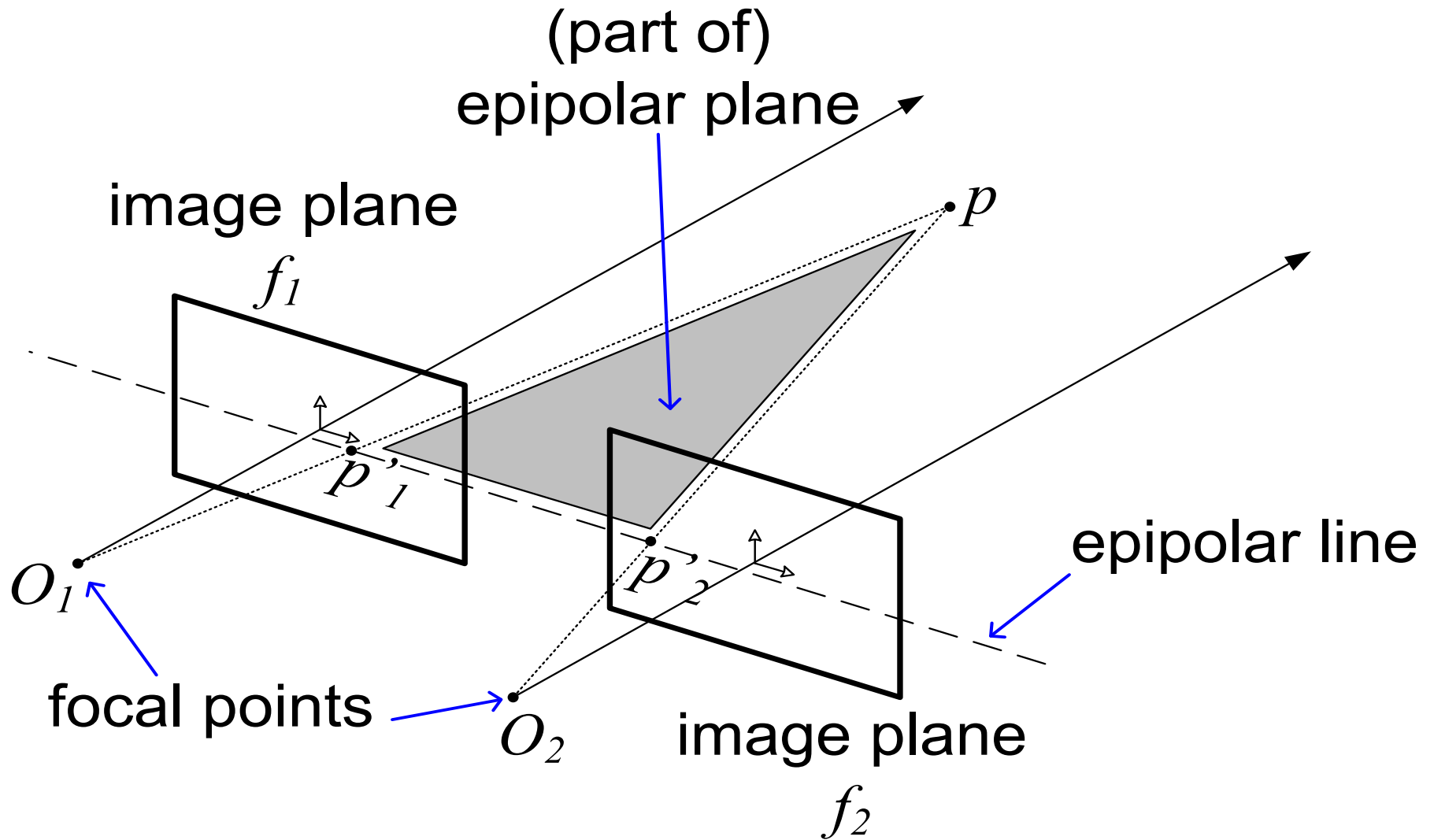
Stereo Vision: Pinhole Camera



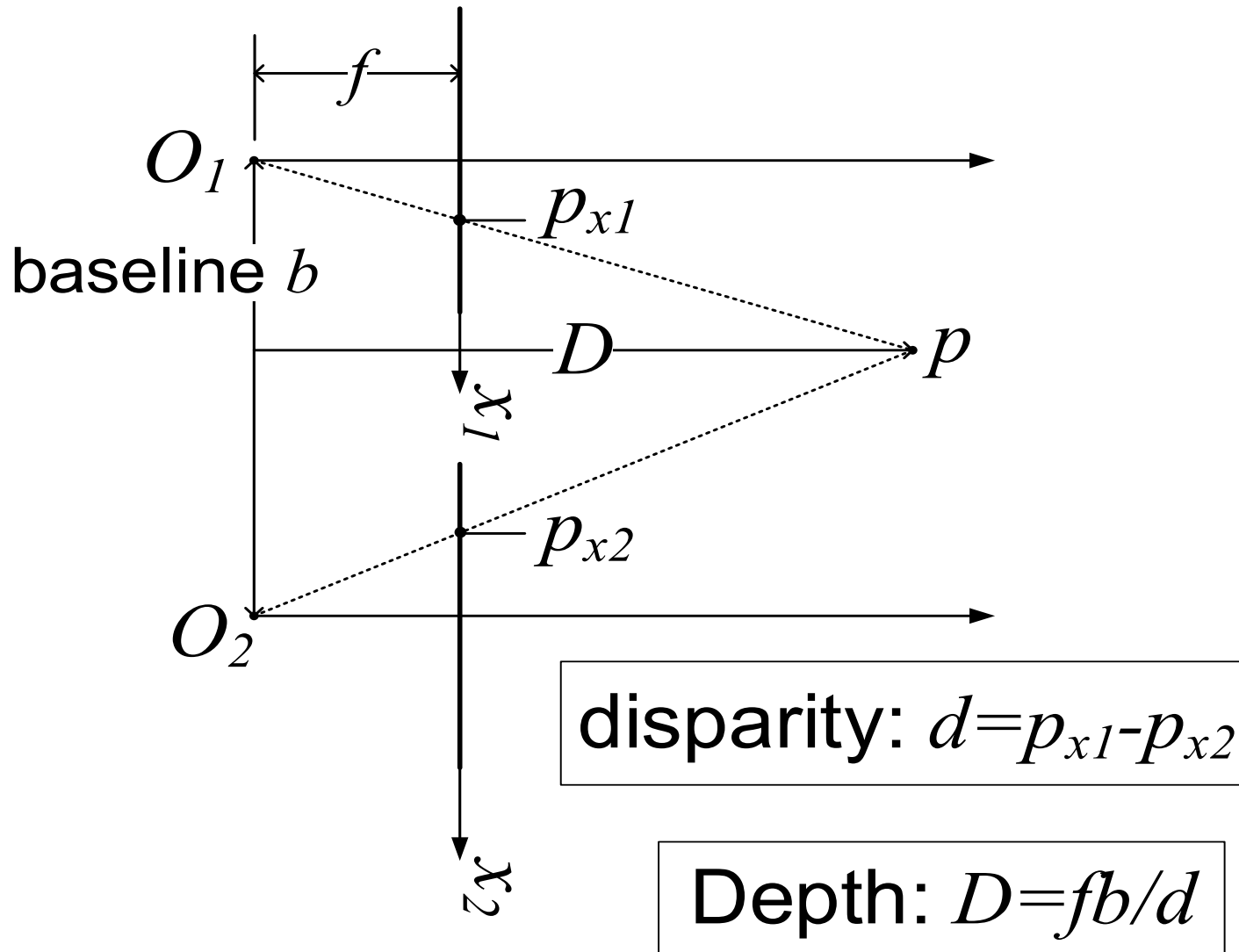
Stereo Vision: Pinhole Camera



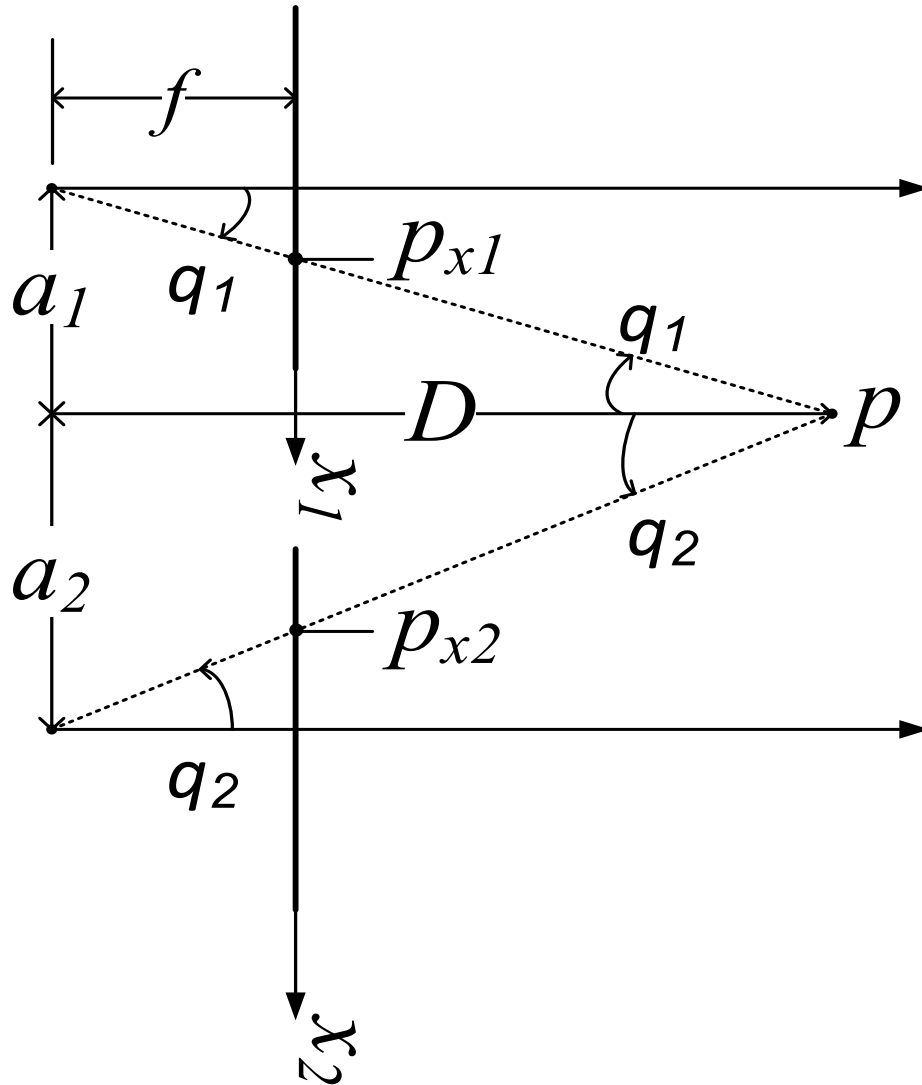
Stereo Vision: Pinhole Camera



Stereo Vision: Pinhole



Stereo Vision: Pinhole



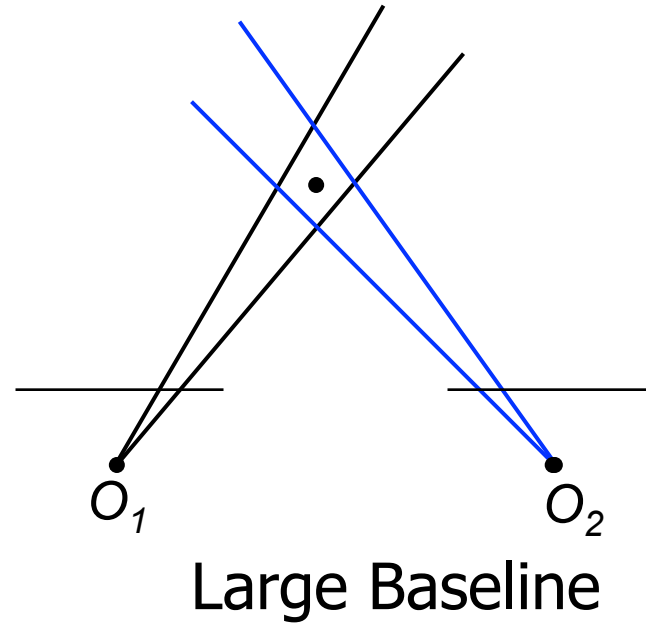
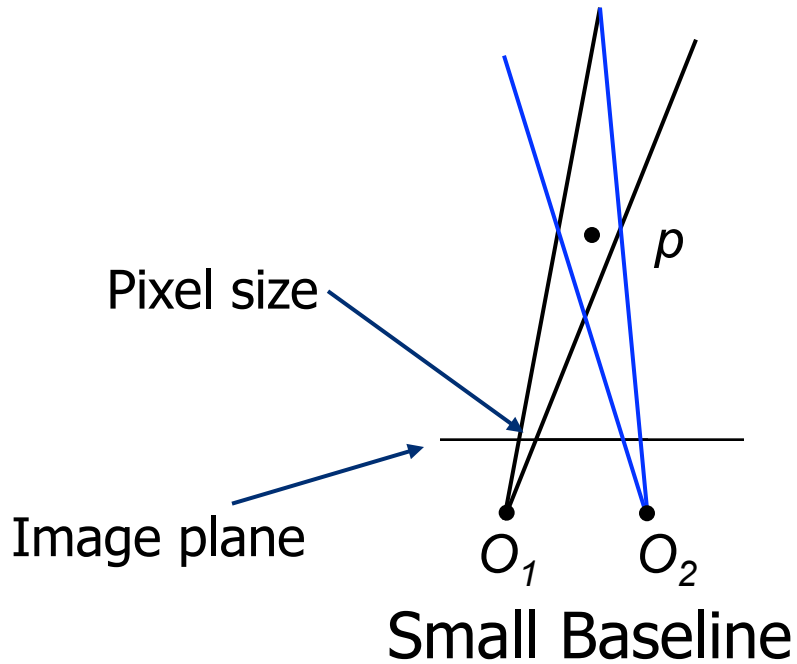
$$\frac{p_{x1}}{f} = \frac{a_1}{D}$$

$$\frac{p_{x2}}{f} = \frac{a_2}{D}$$

$$a_1 + a_2 = b$$



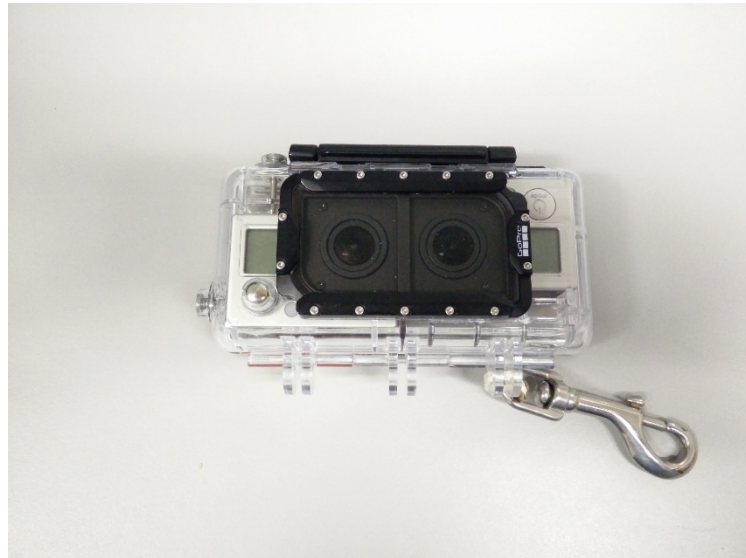
Baseline



- What's the optimal baseline?
 - Too small: large depth error
 - Too large: difficult search problem

Baseline

GoPro 3D HERO System



$b=3.2\text{ cm}$

source: <http://www.cvlibs.net/datasets/kitti>



$b=54\text{ cm}$

Matching Left and Right



3D reconstruction



Stereo: Disparity Map



Using real-time stereo vision for mobile robot navigation

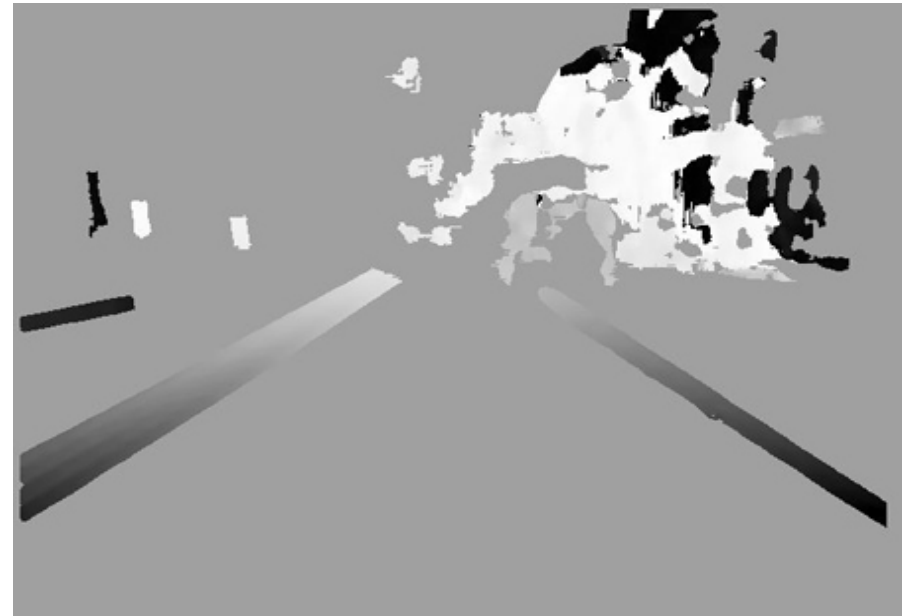
Don Murray

Jim Little

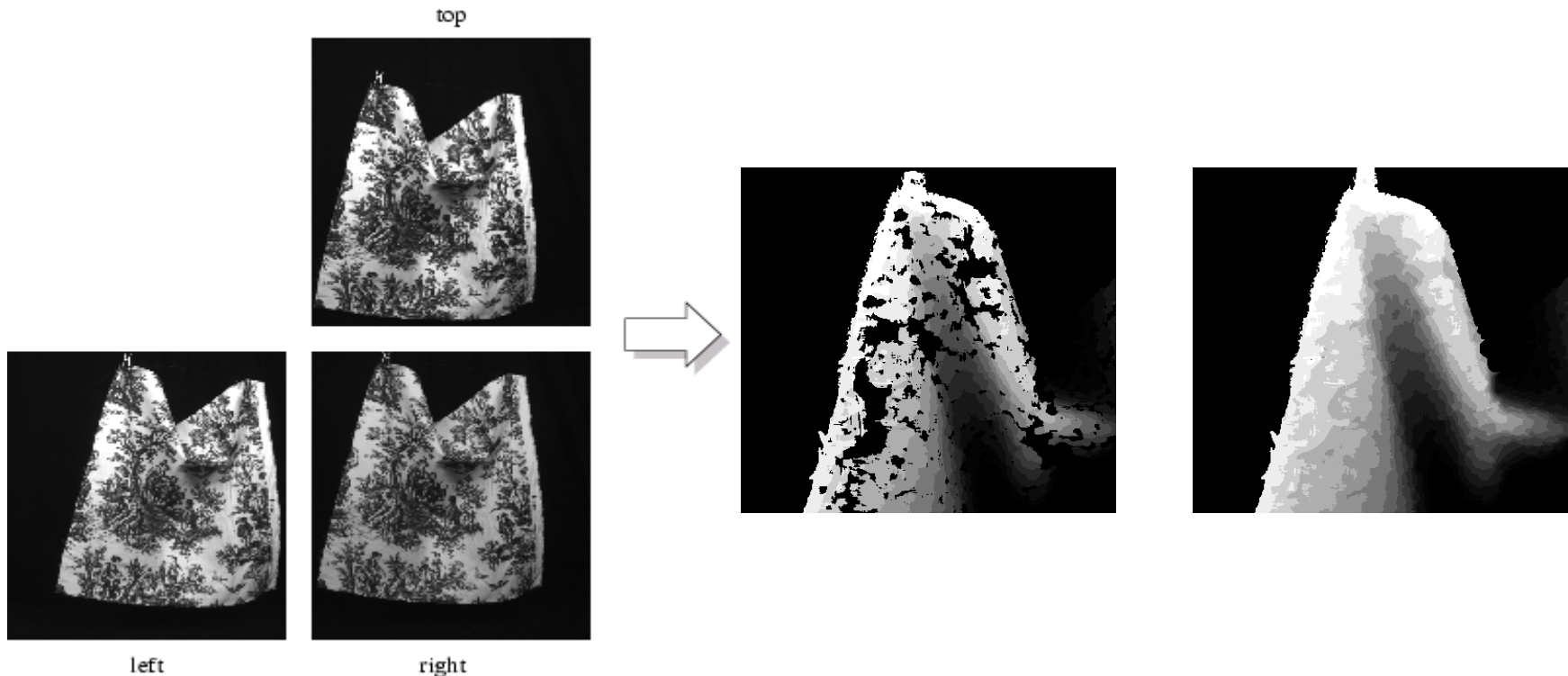
Computer Science Dept.
University of British Columbia
Vancouver, BC, Canada V6T 1Z4



Depth Map in a City



Another Example (Hole Filling)



Cloth Parameters and Motion Capture by David Pritchard
B.A.Sc., University of Waterloo, 2001



Stereo Vision

- Large number of algorithms out there:

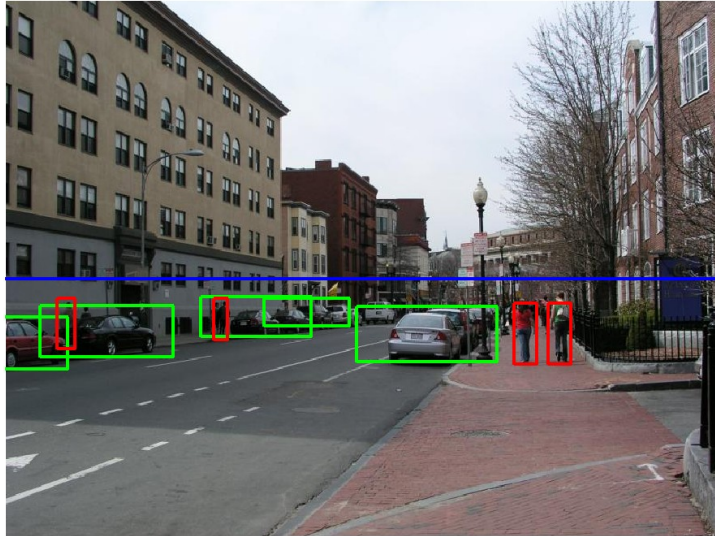
<http://vision.middlebury.edu/stereo/>

rank 43 different algorithms.



Object recognition

source: <http://www.cs.cornell.edu/courses/cs4670/2013fa/>

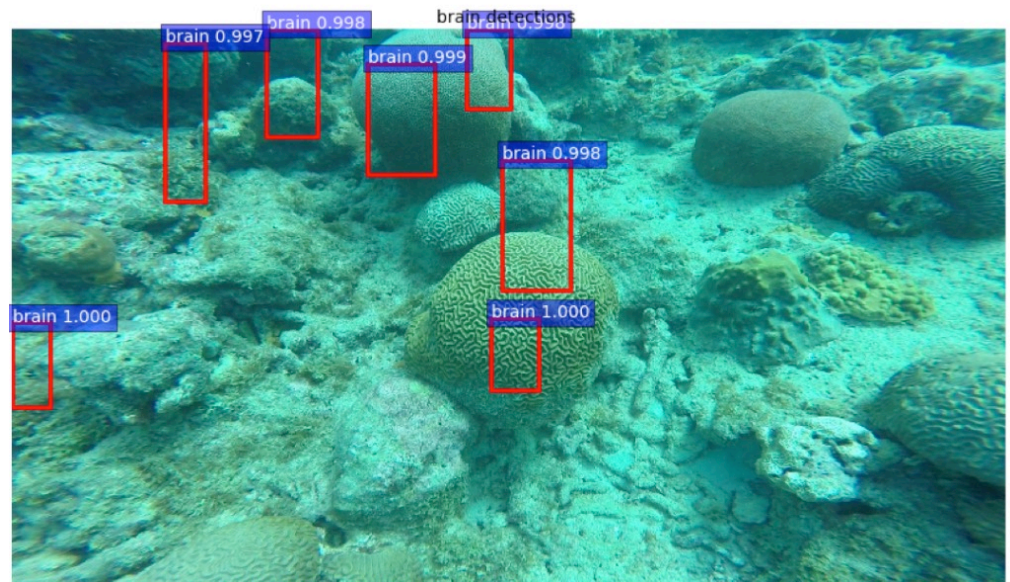


Pedestrian and car detection



Lane detection

From GoPro 3D Hero at Barbados 2015 Field Trial



Coral classification

Bag of words

Object



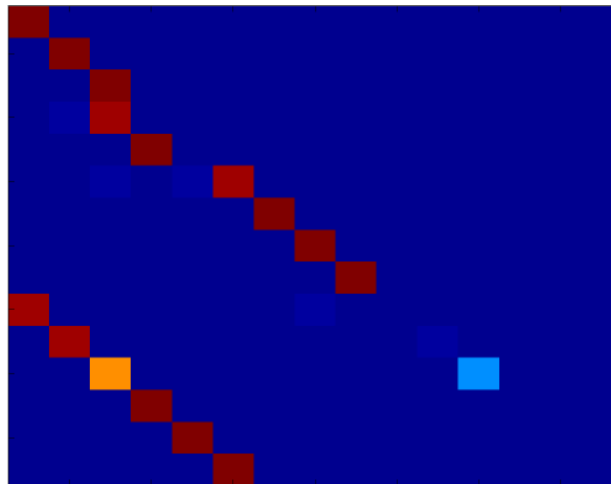
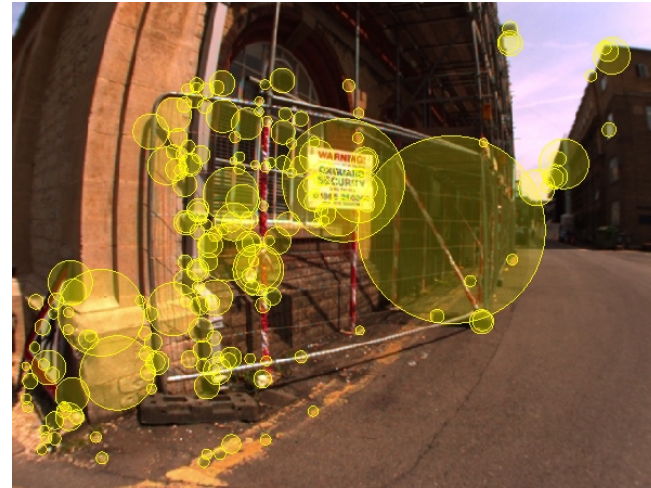
Bag of 'words'



source: <http://wikimedia.org>

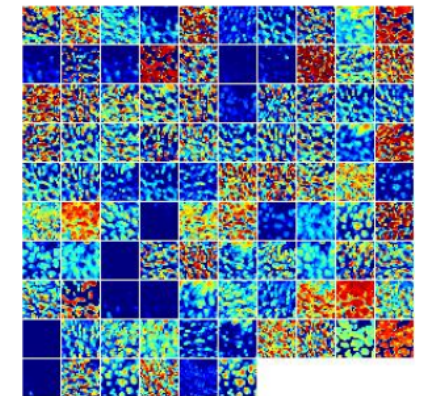
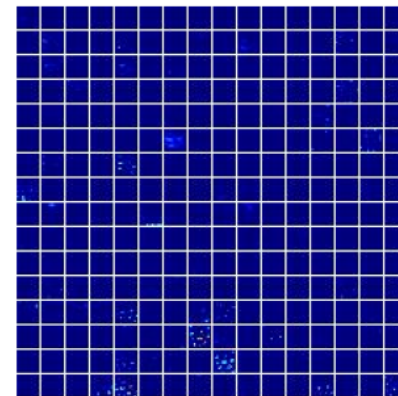
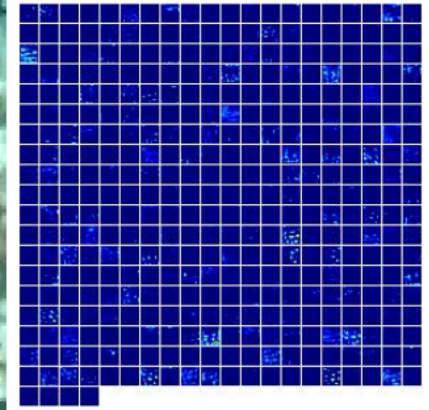
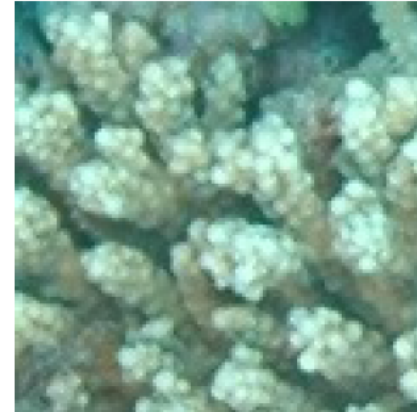
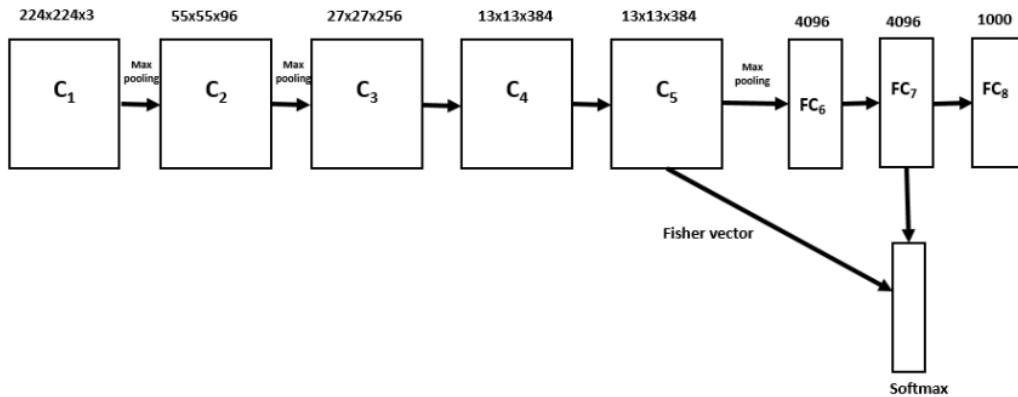


Appearance-based place recognition



source: <http://www.robots.ox.ac.uk/~mjc>

Deep learning based classification



Computer Vision Books

- Richard Szeliski, “Computer Vision: Algorithms and Applications”, Springer, 2010
- Richard Hartley and Andrew Zisserman, “Multiple View Geometry in Computer Vision”, Cambridge University Press, 2004
- David Forsyth and Jean Ponce, “Computer Vision: A Modern Approach”, Pearson, 2011



Nice Classes

- Noah Snavely – Introduction to Computer Vision
<http://www.cs.cornell.edu/courses/cs4670/2013fa/lectures/lectures.html>
- Steve Seitz and Rick Szeliski – Computer Vision
<http://courses.cs.washington.edu/courses/cse576/08sp/>

