Lecture 2
Introduction to image feature detection and 3D reconstruction

concepts and ideas

Samuele Carli Martin Hellmich

5 febbraio 2013

Contents

Contents

Purpose

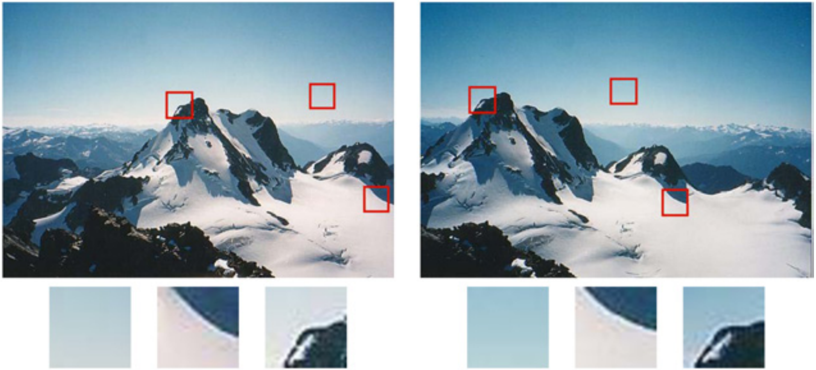
- Detect features in input image
  - Identify keypoint features (mountain peaks, building corners, doorways...)
  - Edges (profile of mountains against the sky...)
  - Features usable for object classification
  - Extract relevant characteristics of image
- Create keypoint and region-based descriptors usable for matching
- Create abstract sketchy representation of subject
- Support higher level algorithms for recognition and tracking

Point features

- Set of corresponding locations in different images
- Easiest to find
- Not too easy to match, but easy to track in subsequent frames
- Alignment of images
- Image mosaics
- Video stabilization
- Stereo matching
- Suitable for matching with occlusion
- Suitable for large scale and orientation changes

Points and patches: example

Not all patches are born equal...



Feature detectors

What are good features to track?

- Textureless patches: nearly impossible to track
- Large contrast changes easy to localize
  - Straight line segments can only be aligned perpendicularly to the line itself
- Best are patches with gradients in at least two directions

Feature detection: matching criteria

Simplest possible: weighted summed square difference

$$E_{WSSD}(u) = \sum_i w(x_i)[I_1(x_i + u) - I_0(x_i)]^2$$

I_n image, $u = \binom{x}{y}$ displacement vector, $w(x)$ spatially varying weighting function, i pixels in patch

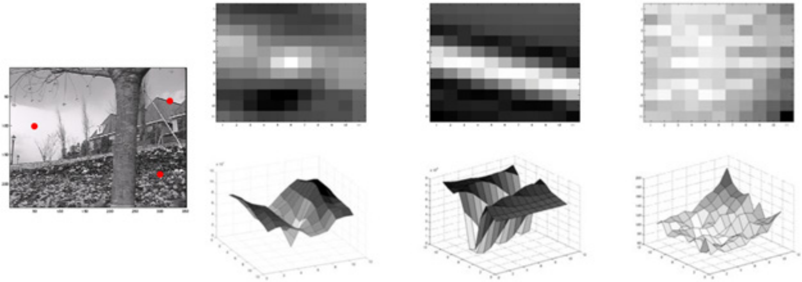
Is a patch suitable for matching?

Auto correlation surface: self stability of E_{WSSD} metric in respect to small position variation Δu

$$E_{AC}(\Delta u) = \sum_i w(x_i)[I_0(x_i + \Delta u) - I_0(x_i)]^2$$

Auto correlation surface: example

Stable, unique minimum indicates good localization



Outline of feature detection algorithm

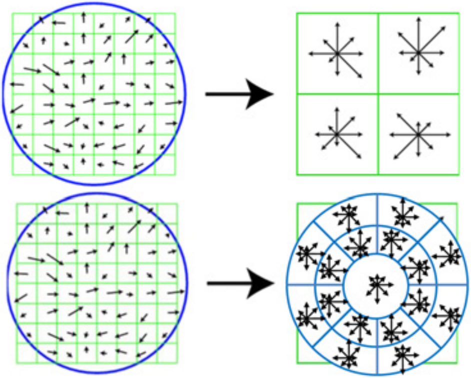
- Compute efficiently auto correlation surface for whole image
  - Different approaches and surface definitions possible
  - Best if image-plane rotation invariant
  - Usually is a matrix (defined by the image) product per pixel
- Compute a scalar interest measure per pixel
- Find local maximum above threshold: those are feature points!
  - Some care needed in choosing local maximum to get as-even-as-possible feature points distribution

Which is a good feature detector?

- **Repeatability**: property of the detector of finding the same keypoints within ϵ pixels in a transformed image (brightness, contrast, rotation, scale, viewpoint change, noise...)
- **Scale invariance**: better to search for keypoints that are stable in both location and scale upon image transformation
- **Rotational invariance**: associate to each keypoint a dominant orientation to avoid mismatching

Feature descriptors

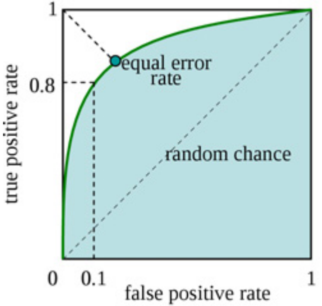
- (matching) features differ by affine transformations and colors even after a compensation is applied → need something better than an image patch
- Many kinds of descriptors exist, gradient-based examples:

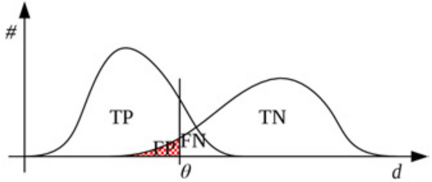


Preliminary feature matching

Create a correspondence between features of two images

- Matching strategy is context dependent, example:
  - image stitching (many well matching features)
  - object recognition in cluttered scene (few matching features)
- Matching threshold needs contextual adjustments





Efficient matching

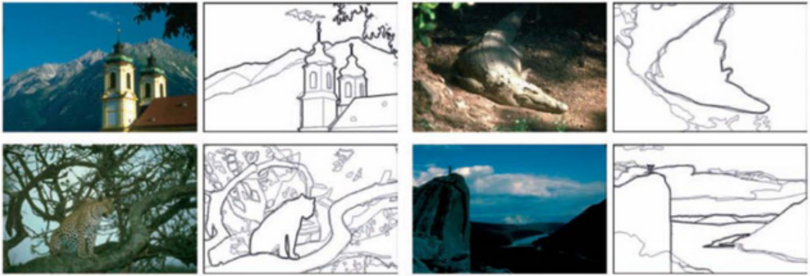
Create a correspondence between matching features of two or more images

- **Indexing structure**: multi-dimensional search tree or hash table
  - Per-image (object search) or global (panorama)
- **Match verification**: geometric alignment
  - global transform usually estimated on random subset of matching features
  - discard matches which do not conform to global geometric transformation
  - more matching features are added later on and transformation gets better estimation

Contents

Edges are important

- Important semantic content: object boundaries, shadows, shapes...
- Does the edges traced under match your expectations?
- Which ones would you trace?



Edge detection

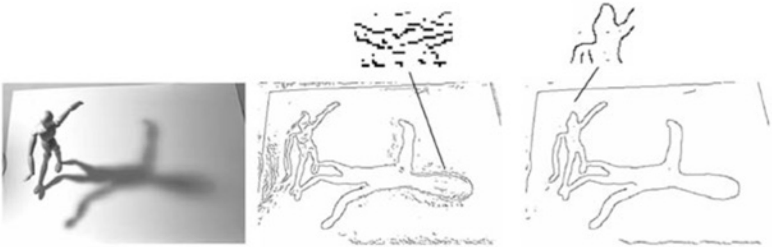
- We define edges as **regions of rapid intensity variation**

$$J(p) = \Delta I(p) = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)(p)$$

- Unfortunately gradient amplifies high frequencies (noise), necessary to smooth image before
  - Smoothing must be circularly symmetric (Gaussian is the only separable circularly symmetric filter)

Scale and blur estimation

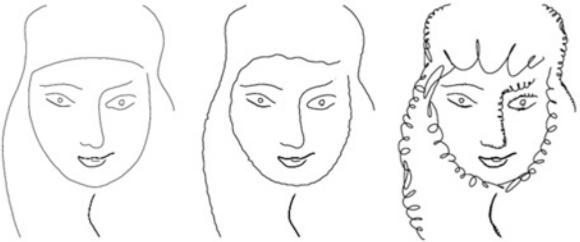
- Estimation of smoothing scale is necessary to have relevant results



- Colors can give useful cues (iso-luminant different colors)
- Cues from brightness, different color channels and texture can be combined to improve global performance

Edge linking

- Isolated edges very useful for stereo matching...
- ...but curves can be much more useful!
- Many techniques available to link edges and form curves
- Useful to vectorize and/or scale images
- Parameterization could be used to change character of a curve to help recognition (or to have fun)



Lines

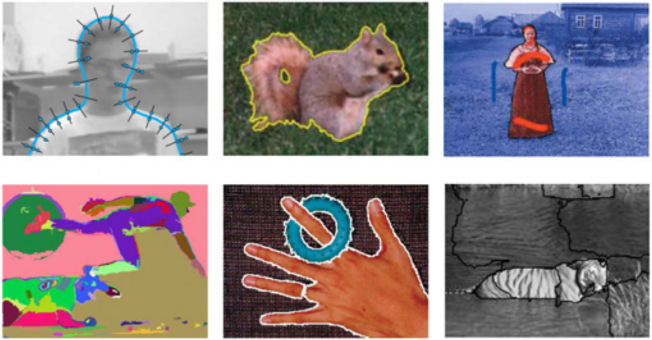
- Human world is full of straight lines: they can be a good hint in many useful applications
- Curves can be approximated with segments, many techniques available
- Segments can be grouped together into extended lines (Algorithms available to take care of holes and missing pieces)
- Vanishing points can be used
  - to detect parallel lines in 3D
  - as good hints to refine line measurements
  - to estimate camera intrinsics and estrinsics parameters
- In robotics context can carry more useful information than curves

Contents

Segmentation

Task of **finding groups of pixels that "go together"**

- similar to cluster analysis in statistics, hundreds of different algorithms available
- even in CV one of oldest and most widely studied problem



Active contours

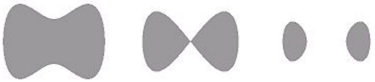
- Family of boundary detectors which move iteratively towards a final solution
- Example: **Snakes**
  - Iteratively minimize energy defined as sum of:
    - **External energy**: minimal when snake at object boundary position (follow image edges) [E_{int}]
    - **Internal energy**: minimal when snake has sensible shape: prefer smooth shapes (high energy to high curvature and elongated contours [E_{ext}]
    - **Constraint energy**: minimal when snake follows eventual user hints [E_{con}]

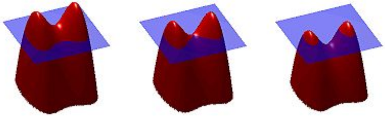
Active contours: Snake example

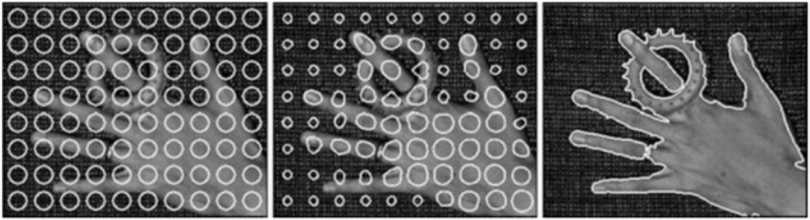
$$E_{snake} = \int_S (E_{int}(s) + E_{ext}(s) + E_{con}(s))ds$$

Level Sets

- Contour represented as an evolving signed function
- Surface function and zero level evolved
- Often used for medical imaging







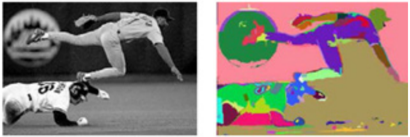
Split and merge

- Simplest way of segmenting (greyscale) image: set threshold and compute connected components
- Usually not enough due to lighting and intra-object statistical variations
- Many techniques try to overcome limitation by:
  - recursive split of image in pieces based on region statistic
  - merging pixels and regions in a hierarchical fashion
  - merging and splitting of regions combined

Split and merge: examples

Watershed segmentation: evolve local minimum until other bins are met

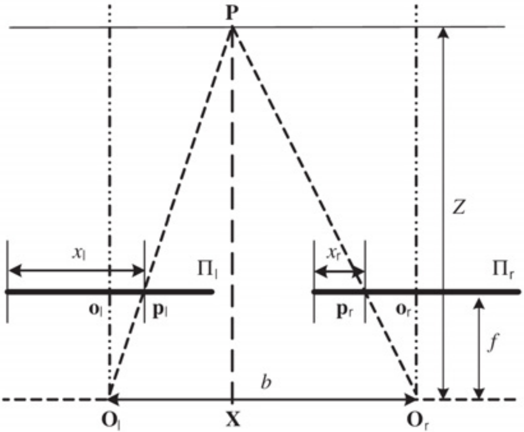
Graph-based merging segmentation



Contents

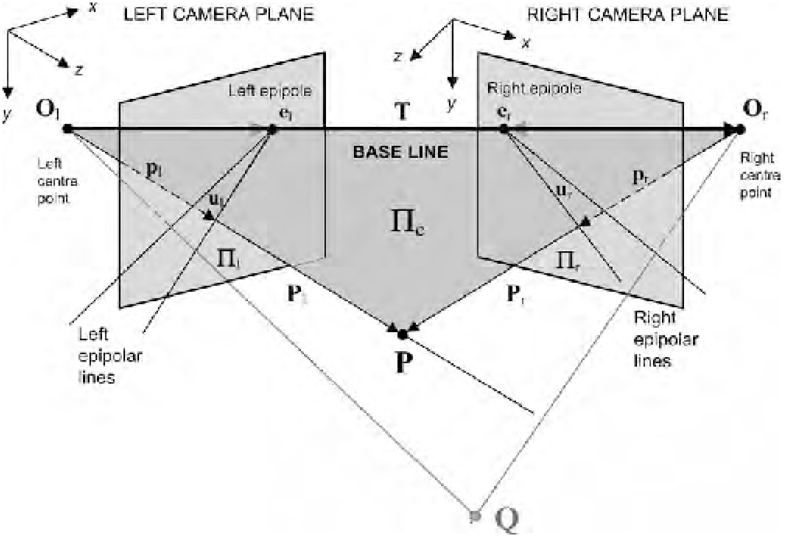
The simplest setup

Triangulation



$$D_x(p_l, p_r) = p_{l1} - p_{r1} = x_l - x_r = \frac{bf}{Z}$$
$$Z = \frac{bf}{x_l - x_r}$$

Epipolar geometry



The essential and fundamental matrices

The Essential matrix describes the relationship between the cameras in homogeneous camera coordinates:

$$p_l^T \mathbf{E} p_r = 0$$

The Fundamental matrix describes the same in pixel coordinates and is related to the fundamental one:

$$\mathbf{F} = \mathbf{K_l^T E K_r}$$

... but we must find (approximate) them!

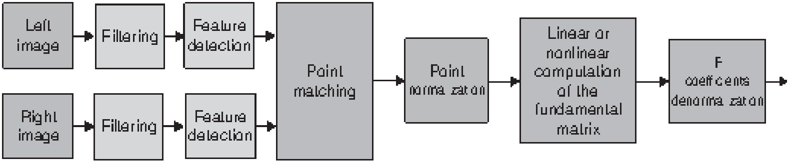
The matrix F

Canonical case:

$$\mathbf{F}_C = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & c \\ 0 & -c & 0 \end{bmatrix}$$

Otherwise: unknown

How to find F



8 point matching algorithm

8 points matching

Find 8 points, solve $\sum_{i=0}^{7} q_i f_i = 0 \rightarrow easy$

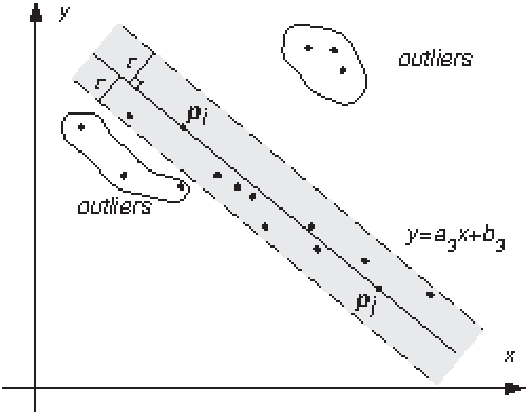
But:

- numerical instabilities
- wrong matches

Solution:

- normalize
- more points, then least squares
- remove outliers

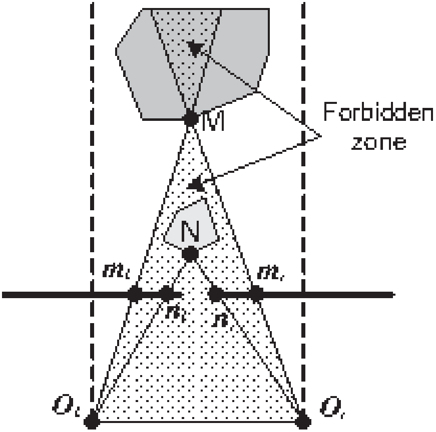
The RANSAC Algorithm



Constraints to simplify our life

- Epipolar contraint
- Uniqueness constraint
- Similar regional brightness

Ordering constraint



Feature compatibility constraint

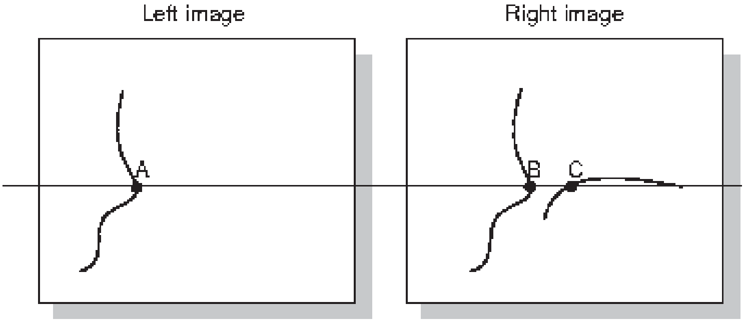
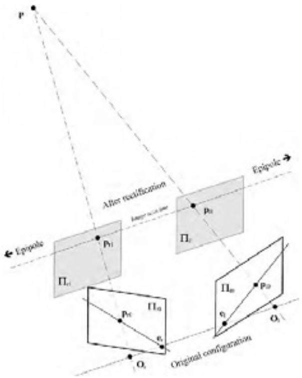


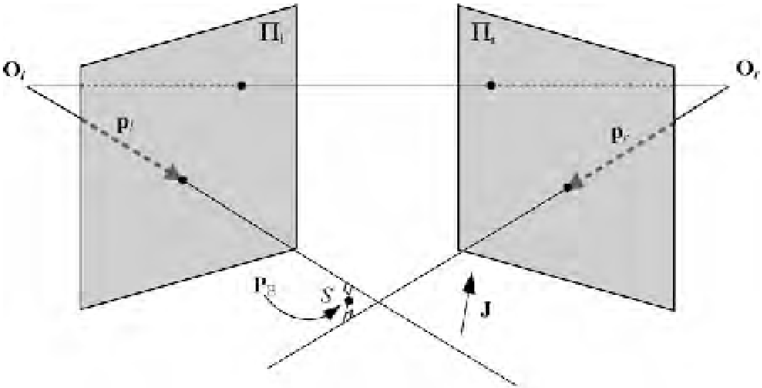
Image rectification

Wanted: canonical case

- transformation **Q**
- based on **T**
- rotation **R** of right camera

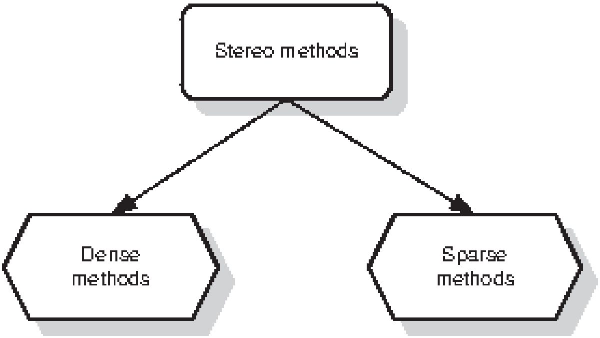


Approximated triangulation

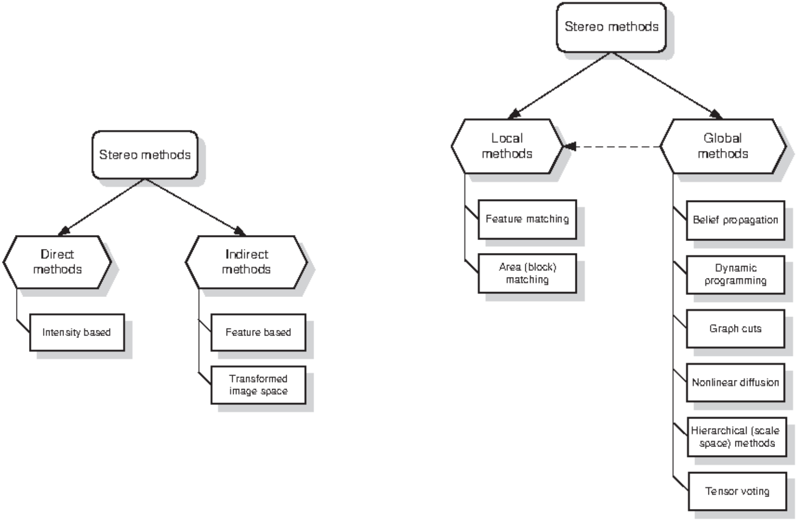


Many options

Now we can get the rest of the points

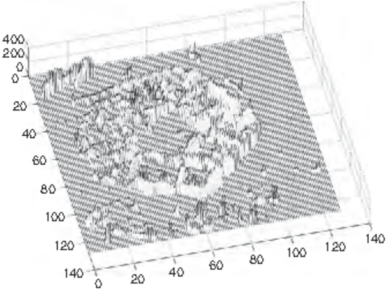


More choices

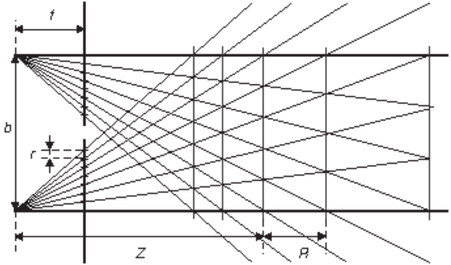


Results


1
2
3
4
5
6
7
8
9
10
11
12
13
15
16


400
200
0
0
20
40
60
80
100
120
140
0
20
40
60
80
100
120
140

Depth resolution



Z[m]/b[m]	0.1	0.5	1.0	5	10
0.05	0.000226	0.0057	0.023	0.635	2.91
0.3	0.000038	0.00094	0.0038	0.096	0.39

Stereo calibration

Camera calibration

- Based on model
- Self-calibration

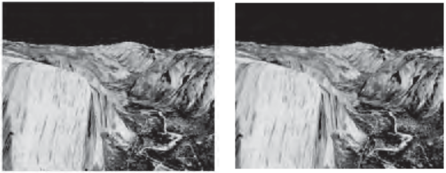
Stereo calibration

- Camera extrinsic parameters:
- $\mathbf{R_x}$ rotation, $\mathbf{t_x}$ translation
- $\mathbf{R} = \mathbf{R_l}\mathbf{R_r^T}$

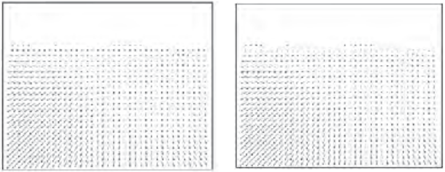
Stereo Vision with one camera

- Depth from motion
- Catadioptric systems
- Plenoptic cameras

Depth from motion



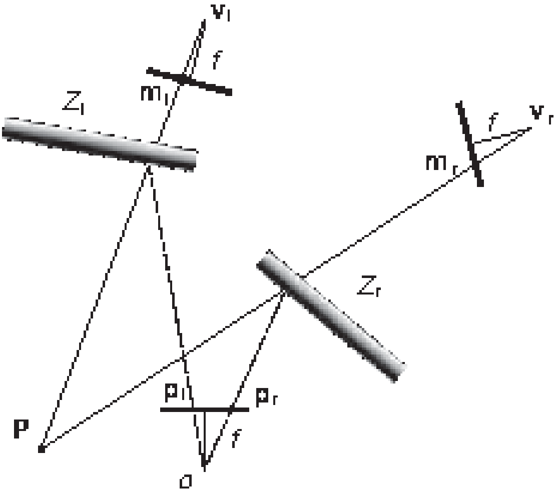
(a) (b)



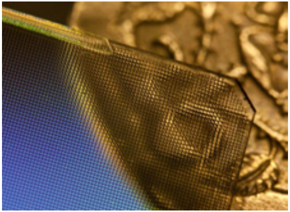
(c) (d)

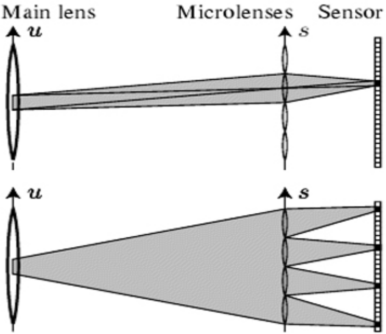
$$I_1(x, y) = I_2(x + \delta x, y + \delta y)$$

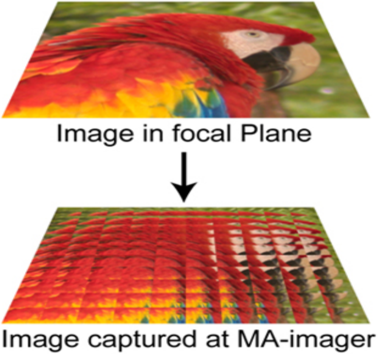
Catadioptric systems



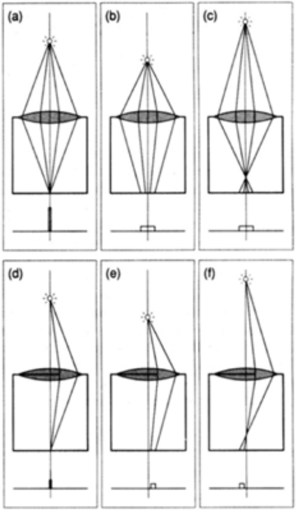
Plenoptic cameras

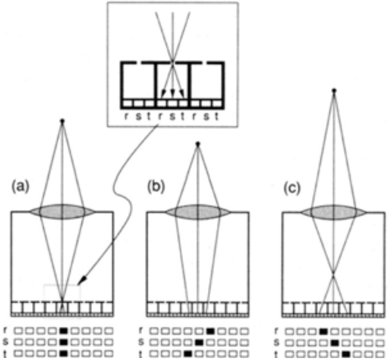






Single lens stereo





Recap

- No matter what you are going to to do...
- ... you surely want to extract features from images
- Make sure you represent features properly! (At least scale and rotation invariance)
- Even higher lever properties (lines, curves) can be extracted, but may require higher lever (classifiers) help
- There are methods to segment images in relevant regions, more or less performant (application dependent)
- In theory triangulation is not a big deal...
- ...but practice is a little more complicated
- 3D vision can be achieved even with just one camera, with different performances