Computer Vision

Riccardo Caprile

May 2022

1 Images

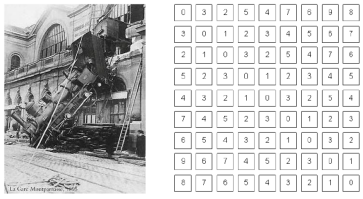
Computer Vision : Understanding the content of an image , usually by estimating a 3D model of the depicted scene

1.1 Introduction

The sense of vision plays an important role in the life of primates : it allows them to infer 3D spatio-temporal properties of the environment that are necessary to perform crucial tasks for survival. The inference is performed by using 2D spatio-temporal signals like images.

But vision is deceivingly easy because it is immediate for us and we perceive the visual world as external to ourselves but it is a reconstruction within our brain. Vision is computationally demanding.

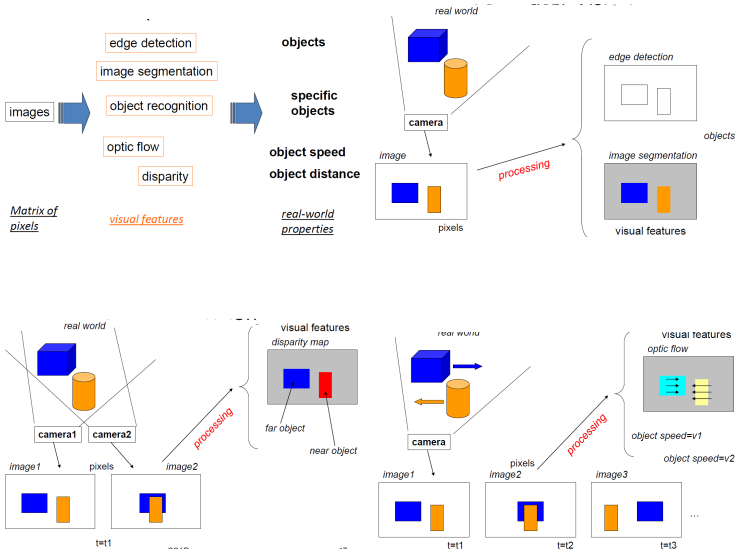
The goal of computer vision is to bridge the gap between pixels and "meaning"



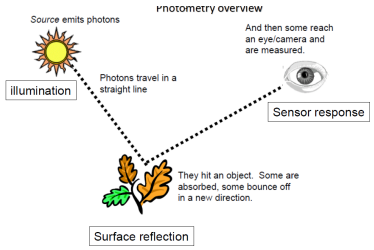
What we see What a computer sees

To interact with the real-world, we have to tackle the problem of inferring 3D information of a scene from a set of 2D images. In general this problem falls into the category of so called inverse problems , which are pone to be ill-conditioned and difficult to solve in their full generality unless additional assumptions are imposed. before we address how to reconstruct 3D geometry from 2D images , we first need to understand how 2D images are generated and processed.

1.2 Computer Vision



1.3 Image Formation



Visual signal s(x,y) : s : $R^2 -> R$.
s(x,y) is a two dimensional function : x and y are spatial coordinates.
The amplitude of s is called **intensity or gray level at the point (x,y)**
s(x,y) = il(x,y) r(x,y).

- s(x,y) : intensity at the point (x,y)
- il(x,y) : illumination at the point (x,y) (the amount of source illumination incident on the objects)
- r(x,y) : reflectance at the point (x,y) (the amount of illumination/transmitted by the objects), where 0 ¡ il(x,y) ¡ inf and 0 ¡ r(x,y) ¡ 1

Illumination

Lumen is a unit of light flow. Lumen per square meter is the metric unit of measure for illuminance of a surface.

Digital image formation is the first step in any digital image processing application. The **digital image formation system** (camera) consists basically of the optical system , the sensor and the digitizer.

1.4 Optical system

The optical system can be modeled as a linear shift invariant having a two-dimensional impulse response h(x,y). The input-output relation of the optical system is described by a 2D convolution.

The two dimensional impulse response h(x,y) is also called PSF (point spread function) and its Fourier transform is called transfer function.

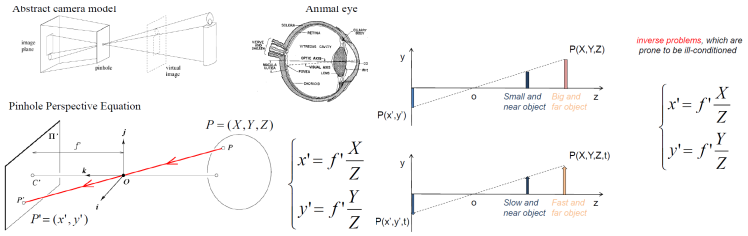
Linear motion blur : it is due to the relative motion , during exposure , between the camera and the object being photographed.

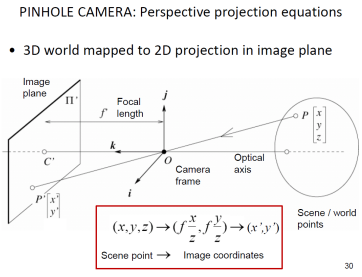
1.5 Camera model

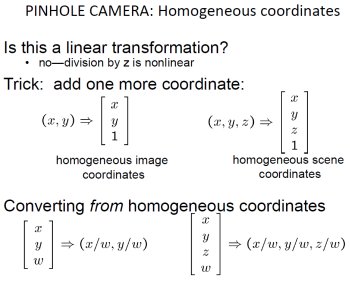
Images are two-dimensional patterns of brightness values. They are formed by the projection of 3D objects on the camera image plane.

Basic abstraction is the **pinhole camera**.

1.5.1 Pinhole Camera







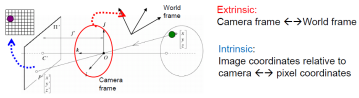
1.5.2 Perspective Projection Matrix

Projection is a matrix multiplication using homogeneous coordinates :

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f' & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z/f' \end{bmatrix} \Rightarrow (f'\frac{x}{z}, f'\frac{y}{z})$$

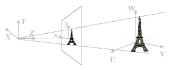
divide by the third coordinate to convert back to non-homogeneous coordinates

1.5.3 Perspective projection & Calibration



Extrinsic:
Camera frame ↔World frame

Intrinsic:
Image coordinates relative to camera ↔ pixel coordinates



Projection model
Intrinsic (lens) parameters
Extrinsic (pose) parameters
Camera calibration

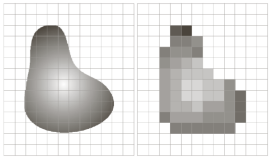
1.6 Cameras with lenses

A lens focuses parallel rays onto a single focal point. Gather more light , while keeping focus; make pinhole perspective projection practical.

- **Depth of field** : a smaller aperture increases the range in which the object approximately in focus
- **Field of view (FOV)** : Angular measure of portion of 3D space seen by the camera. As f gets smaller , image becomes more wide angle (more world points project onto the finite image plane. As f gets larger , image becomes more telescopic.

1.7 Visual sensor and digitizer

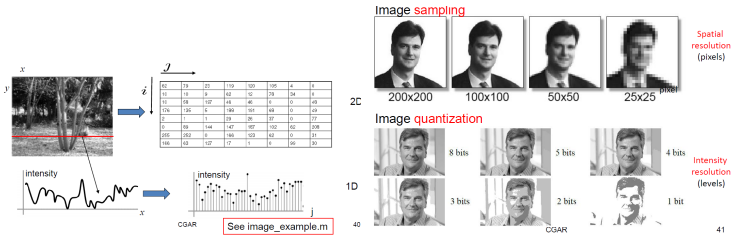
We can think of an image as a function , $g : R^2 -> R$ g(x,y) gives the intensity at position (x,y). Realistically , g: [a,b]x[c,d] $->$ [0,1]



mage sampling and quantization (*errors are introduced*)

1.8 Digital images representation

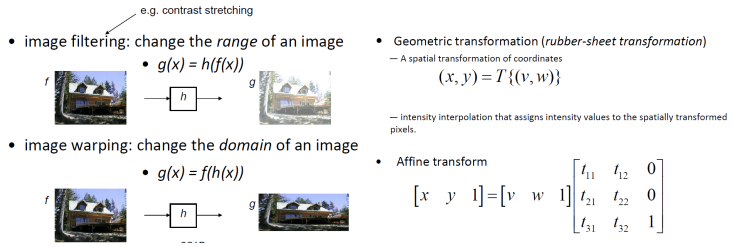
In Computer Vision we operate on digital(discrete) images : Sample the 2D space on a regular grid and quantize each sample. Image represented as a matrix of integer values(intensity)

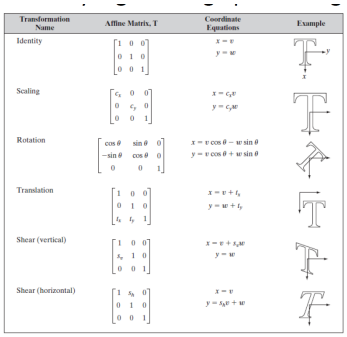


1.8.1 Elementary digital image processing operations

We consider the images a(i,j) , b(i,j) and c(i,j) with i = 1,...,N and j = 1,...,M.

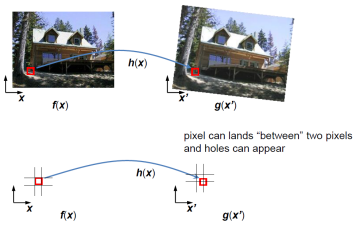
- Image addition/subtraction : c(i,j) = a(i,j) +- b(i,j)
- Thresholding : b(i,j) = a1 if a(i,j) ¡ T , a2 if a(i,j) ¿= T
- Contrast stretching : b(i,j) = ((a(i,j) - amin)) (omax-omin) + omin
- Point nonlinear transformations : b(i,j) = h(a(i,j))



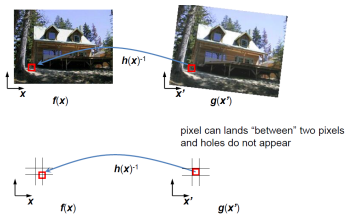


Forward Mapping

(x,y) = T(v,w). It's possible that two or more pixels can be transformed to the same location in the output image.



Inverse Mapping



1.9 Image Interpolation

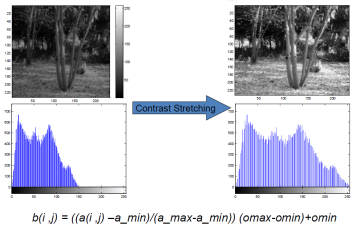
Interpolation : Process of using known data to estimate unknown values.

Interpolation sometimes called resampling , an imaging method to increase the number of pixels in a digital image

1.10 Image Processing

Standard image processing operators map pixel values from one image to another one :

- **Point Operators** : where each output pixel's value only depends on the corresponding input pixel value
- **Neighborhood operators** : where each new pixel's value depends on a small number of neighboring input pixel values.
- **Global Operators** : histogram and Fourier Transform

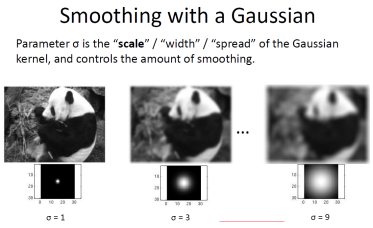


1.10.1 Image Filtering (Convolution)

In order to form a new image , **whose pixels are a weighted sum of original pixel values**, by using the same set of weights at each point :

O(i,j) = I * H = sumk suml I(k,l) H(i-l,j-l).

Where I is the input image , O is the output image and H is the convolution kernel.



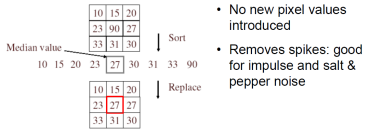
1.11 Noise Models

Digital image are corrupted by noise during image formation process. We often assume the noise is additive : l(x,y) = s(x,y) + ni, where s(x,y) is the deterministic signal and ni is a random variable.

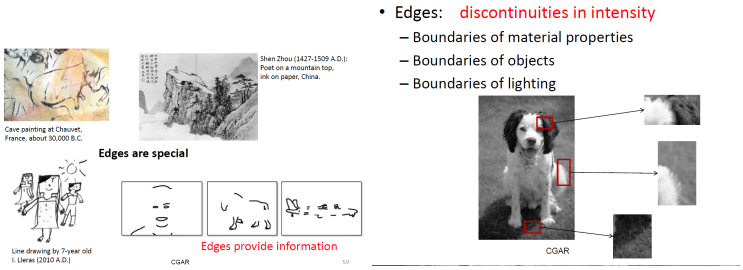
Common types of noise : Gaussian noise (variations in intensity drawn from a Gaussian normal) , Salt and pepper noise (there are random occurences of black and white pixels)

1.12 Image Filtering(Median Filter)

It is a non-linear filter : 1) rank-order neighborhood intensities and 2) take middle value.



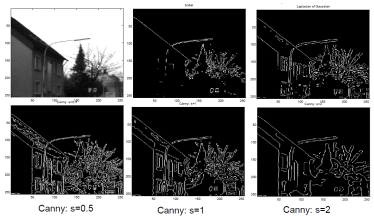
1.13 Image feature : edges



Edge Detection

Goal : identify visual changes (discontinuities) in an image.

Why : Intuitively , most semantic and shape information from the image can be encoded in the edges. More compact than pixels.

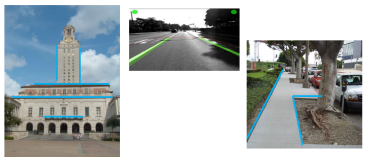


1.14 Boundaries

There are several techniques for representing the shape of boundaries (edges). These representations are the abstraction of edges in a symbolic form. We consider the grouping of edge points into a straight line : Hough transform.

1.15 Line Fitting

Why fit lines? Many objects are characterized by straight lines.



Choose a parametric model to represent a set of features.

Membership criterion is not local - can't tell whether a point belongs to a given model just by looking at that point.

Three main questions : 1) What model represents this set of features best? 2) Which of several model instances gets which feature? 3) How many model instances are there?

Extra edge points(clutter) , multiple models : which points go with which line , if any?

Only some parts of each line detected , and some parts are missing : How to find a line that bridges missing evidence?

Noise in measured edge points , orientations : How to detect true underlying parameters?

It's not feasible to check all combinations of features by fitting a model to each possible subset. **Voting** is a general technique , where we let the features vote for all models that are compatible with it (Cycle through features , cast votes for model parameters and look for model parameters that receive a lot of votes. Noise & clutter features will cast votes too , but typically their votes should be incosistent with the majority of good features.

1.16 Hough Transform

Hough Transform is a voting technique that can be used to solve all of those difficulties.

Main Idea :

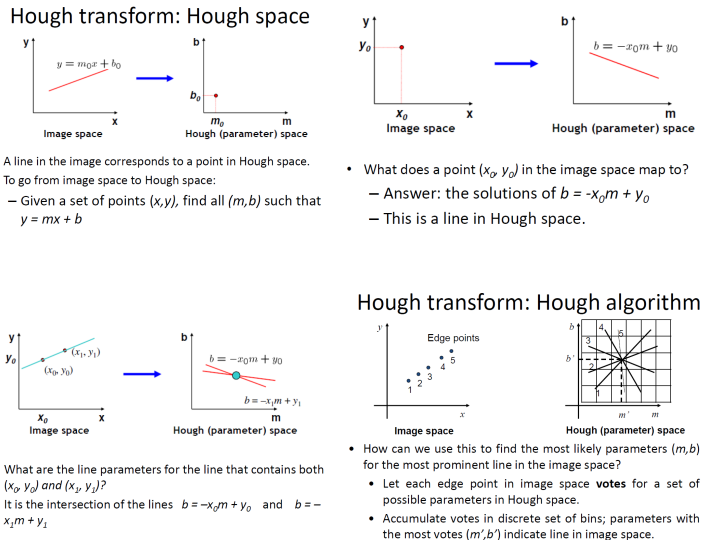
1. Record all possible lines on which each edge point lies. 2. Look for lines that get many votes.

We will use the Hough transform to fit the equation of a straight line to the edge points.

The equation of a line y = mx+b can be rewritten as b = (-x)m+y , this is a the equation of a line in the m-b parametric space.

Therefore, a point (x,y) in the x-y space is mapped to a straight line in the m-b space.

The lines intersect at a single point(m',b') in the m-b space , if the (x,y) points belong to the same straight line in the x-y space.

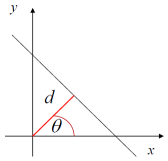


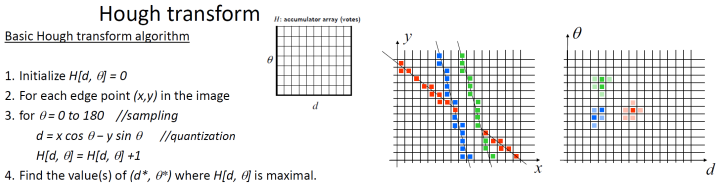
1.17 Polar representations for lines

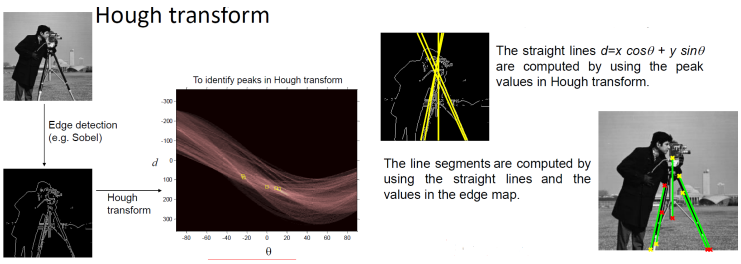
The parametric model presented has some difficulties in the representation of vertical straight lines , because the parameter m tends to infinity.

Thus , we chose the polar representation of a straight line.

The polar form of the equation of a straight line : d = x cos θ + y sinθ







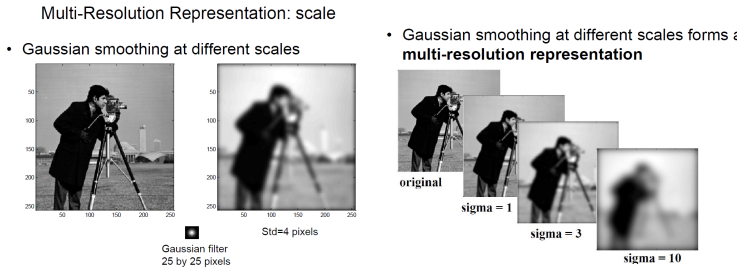
2 Segmentation and Disparity

2.1 Multi-Resolution Representation

We may wish to change the resolution of an image before proceeding further : - We may need to interpolate a small image to make its resolution match the desired one. - We may want to reduce the size of an image to speed up the execution of an algorithm. - Sometimes , we do not even know what the appopriate resolution for the image should be for example the task of finding a face in an image. Since we do not know the scale at which the face will appear, we need to generate a whole pyramid of differently sized images and scan each one for possible faces.

2.1.1 Concept : Scale Space

Basic idea : different scales are appropriate for describing different objects in the image and we may not know the correct size ahead of time

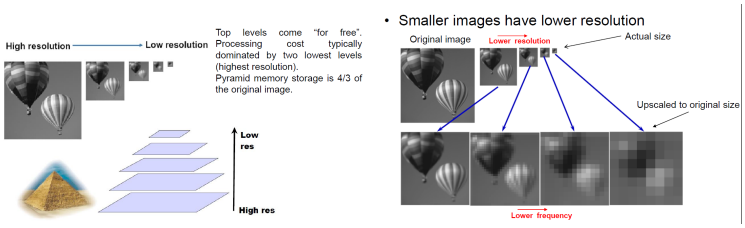


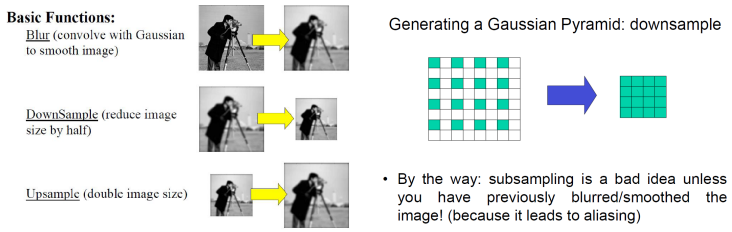
2.2 Pyramid Representation

Because a large amount of smoothing limits the frequency of features in the image , we do not need to keep all the pixels around!.

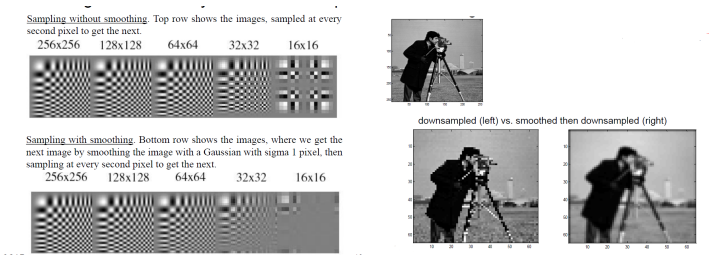
Strategy : progressively reduce the number of pixels, as we smooth more and more. Leads to a "pyramid" representation if we subsample at each level.

2.2.1 Gaussian Pyramid



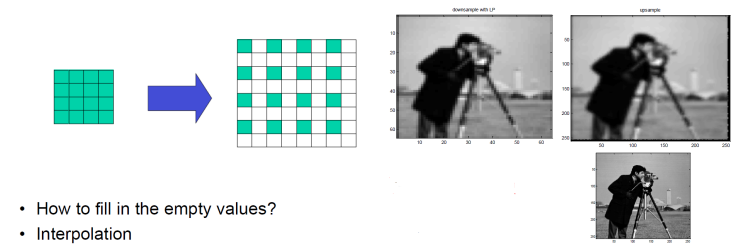


Can't shrink an image by taking every second pixel. If we do it , characteristics errors appear. Typically , small phenomena look bigger and fast can look slower. The message is that high frequencies lead to trouble with sampling. Solution : suppress high frequencies before sampling.



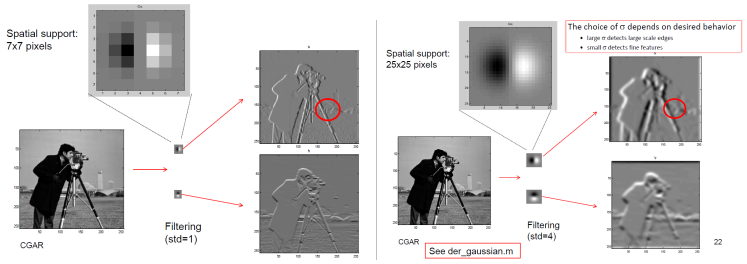
Upsample

Multi-resolution image analysis : Look for an object over various spatial scales by first finding a smaller instance of that object at a coarser level of the pyramid and then looking for the full resolution object only in the vicinity of coarse-level detections. **Coarse to fine image processing** : form blur



estimate motion analysis on very low-resolution image , upsample and repeat. Often a successful strategy for avoiding local minima in complicated estimation tasks.

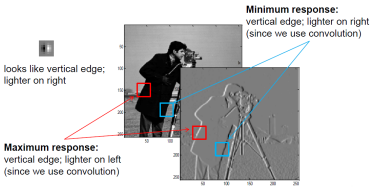
2.3 Derivative of Gaussian Filter : Scale



2.4 Derivative of Gaussian Filter : Matching

Convolution with a filter can be viewed as comparing a little picture of what you want to find against all local regions in the image.

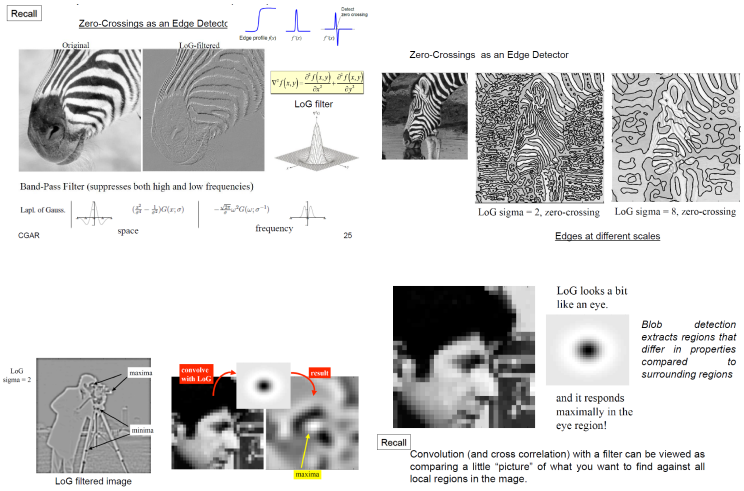
For this reason , it is sometimes called "matched filtering". In fact , you can prove that the best linear operator for finding an image patch is essentially the patch itself. This is the template matching.



2.5 Laplacian of Gaussian (LoG) Filter

2.5.1 Other uses of LoG : Blob Detection

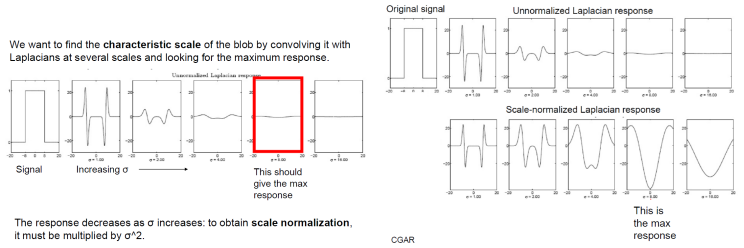
How can an edge finder also be used to find blobs in an image??



2.6 From edges to blobs

Blob = superposition of nearby edges

2.6.1 Scale selection

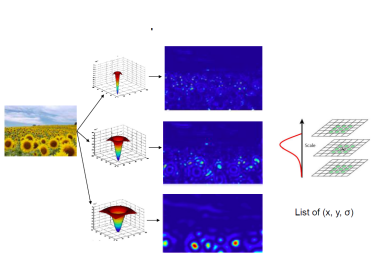


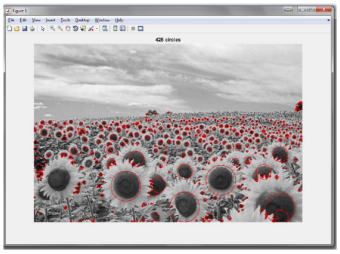
2.6.2 Characteristics Scale

We define the characteristics scale as the scale that produces peak of Laplacian response.

2.7 Scale-space blob detector

1. Convolve image with scale-normalized
2. Find maxima of squared Laplacian response in scale-space
3. Non-maxima suppression in scale-space





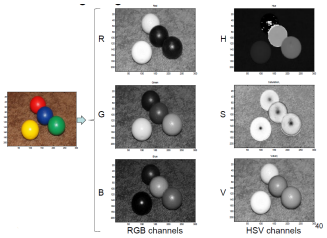
2.8 Objects' description

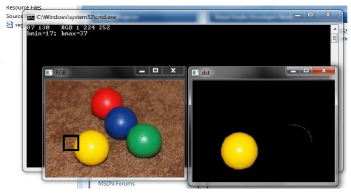
The **shape of an object** can be described either in terms of its boundary or in terms of the region it occupies(blob).

Region-based representation requires image segmentation in several homogeneous region. Image regions are expected to have homogeneous characteristics

2.9 Image segmentation based on color

To **segment an image** based on color , it is natural to think first of the HSV/HSI space , because color is conveniently represented in the hue channel. The objective is to classify each pixel in a given image as having a color in the specified range or not. Coding these two sets of pixel in an image with black and white produces a binary segmented image.





We choose a rectangular region that contains samples of color we wish to segment out of the color image. We compute the mean h_m and the standard deviation h_std of the hue values of pixels contained within the rectangle. Then , we code each point as 1 , if it is inside the range of hue values, otherwise as 0.

2.10 Geometrical properties of regions (blob analysis)

Once the scene is segmented into regions (blobs), we can determine the geometrical properties of these regions. We assume that the blob is represented by a m x n binary sub-image B(i,j).

Area or size of the region is a total number of pixels occupied by the region

:

$$\sum_{i=1}^{m}$$

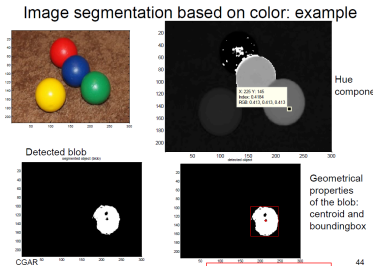
$$\sum_{j=1}^{n} B(i,j)$$

The **centroid** is like the center of an arbitrary shaped region , and it can be used to represent the location of a region :

$$\bar{i} = \frac{1}{A} \sum_{i=1}^{m} \sum_{j=1}^{n} iB(i,j)$$

$$\bar{j} = \frac{1}{A} \sum_{i=1}^{m} \sum_{j=1}^{n} jB(i,j)$$

The **perimeter** of a region is the sum of its border pixels. A pixel which has at least one pixel in its neighborhood from the background is called a border pixel.



2.11 Feature Points

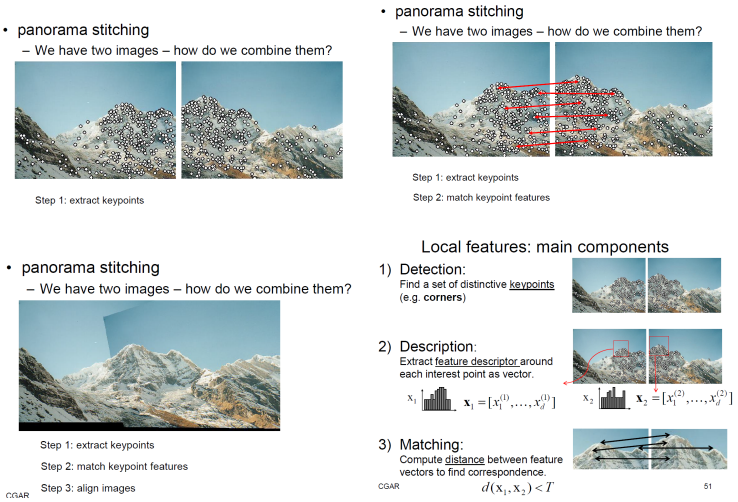
The first kind of **feature** that you may notice are specific locations in the images , such as mountain peaks , building corners , doorways , or interestingly shaped patches.

These kinds of localized feature are often called **keypoint features** or **interest points** and are often described by the appearance of patches of pixels surrounding the point location. Edges and lines provide information that is complementary to both keypoint and region-based descriptors.

2.12 Correspondence across views Example

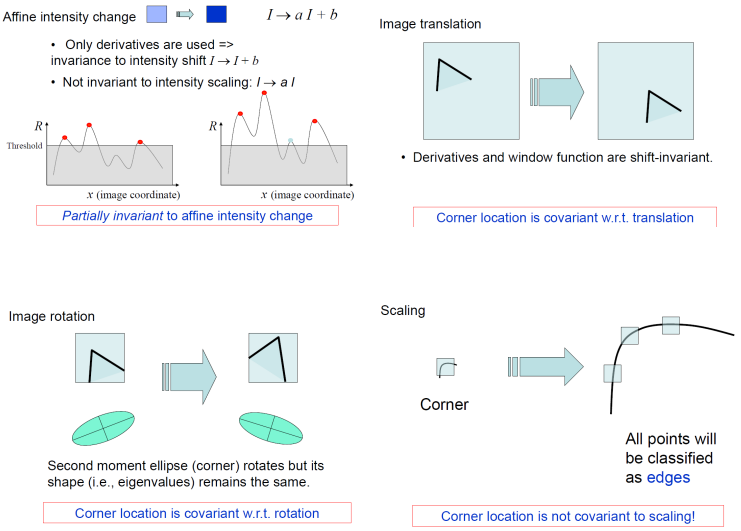
2.13 Invariance and Covariance

Are keypoint invariant to photometric transformations and covariant to geometric transformations?



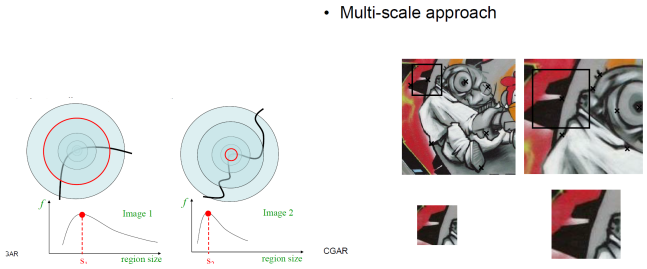
Invariance : image is transformed , and corner locations do not change. **Covariance**: if we have two transformed versions of the same image , features should be detected in corresponding locations.

2.13.1 Harris corner detector : properties

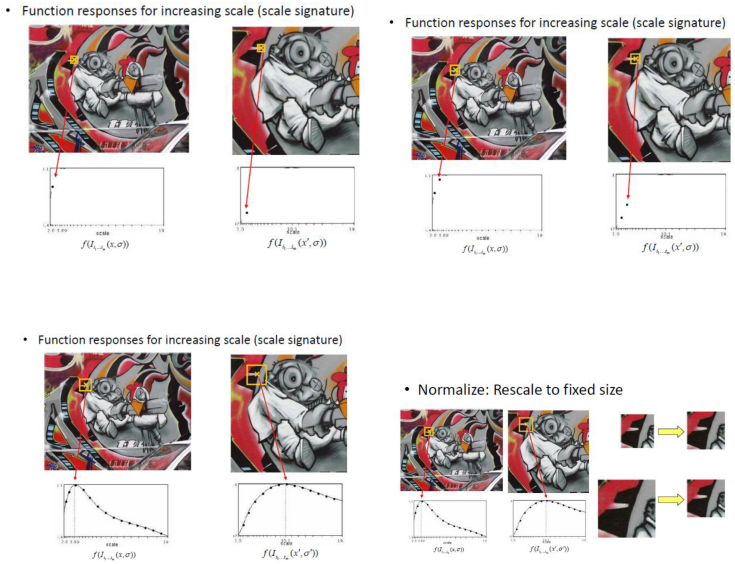


2.13.2 Scale invariant interest points

How can we detect scale invariant interest points? Intuition : Find scale that gives local maxima of some function f in both position and scale.



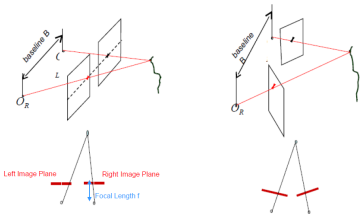
Design a function f on the region , which is scale invariant. This scale invariant region size is found in each image independently



2.14 Stereopsis

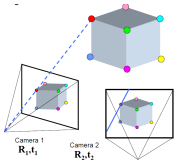
It is hard to recover 3D information from 2D images without extra knowledge. Much of geometric vision therefore is based on information from 2 or more camera locations.

Some definitions of a stereo system : Parallel optical axes and converging optical axes.

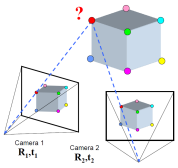


Stereopsis Problems

Stereo Correspondence : Given a point in one of the images , where could its corresponding points be in the other images?



Structure : Given the projections of the same 3D point in two or more images , compute the 3D coordinates of that point.



2.14.1 Stereo System

Stereo : structure from shift between two views. We'll need to consider : info on camera pose and image point correspondences.

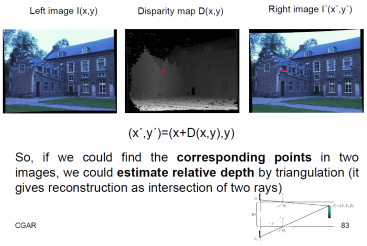
Let's look at a simple stereo system first. Assume : parallel optical axes and known camera parameters.

For stereo systems , which differ only by an offset in x, the projection of y is the same in both images. We call this a rectified system.

Stereo image rectification : we can always turn a verged system to a non verged system

The goal is to generate a depth map of a world scene by triangulating 3D positions of points in space using a passive 2-camera system

2.14.2 Depth from disparity

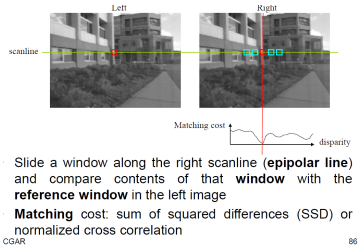


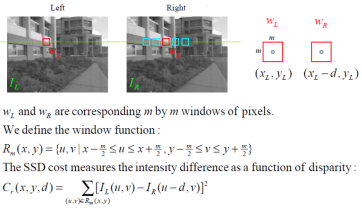
There are two problems for stereo. Correspondence problem (Given two images , how can you find corresponding points that match?) and Reconstruction problem (Given matching points between images , how can you reconstruct the 3D scene?)

2.14.3 Correspondence Problem

Region-Based : Region-based matching only works where there is texture (compute a confidence measure for regions, apply continuity or match ordering constraints). Region matching can be sensitive to changes in surface orientation.

Feature-Based : Feature-based leads to sparse disparity maps (interpolation to fill in gaps and scale-space approaches to fill in gaps.) Feature-based can be sensitive to feature "drop-outs"





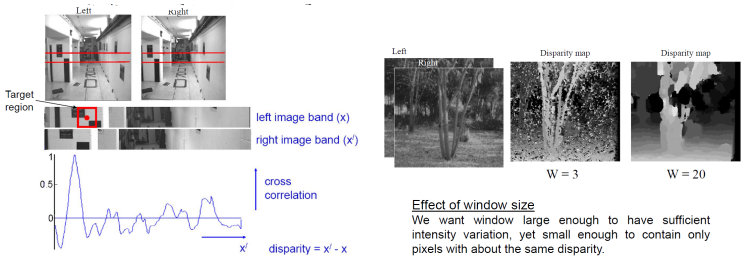
Even where the cameras are identical models , there can be differences in gain and sensitivity. Note that we can change the SSD by making the image brighter or dimmer. For these reasons and more , it is a good to normalize the pixels in each window. As a result , it is common to subtract the mean of both images and normalize by variance.

Algorithm

Deciding the Range : The first step in correspondence search is to compute the range of disparities to search. Assume a non-verged system. Therefore , we have :

d = f b / z_max, we calculate d_min = f b / z_max and d_max = f b / z_min

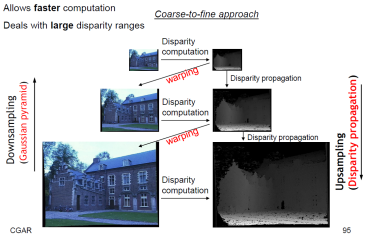
Thus, for each point u_l in the left image , we will search points $u_l + d_min + d_max$ in the right.



Note we can turn this around and start at a point u_r and search from $u_r - d_max$ to $u_r - d_min$

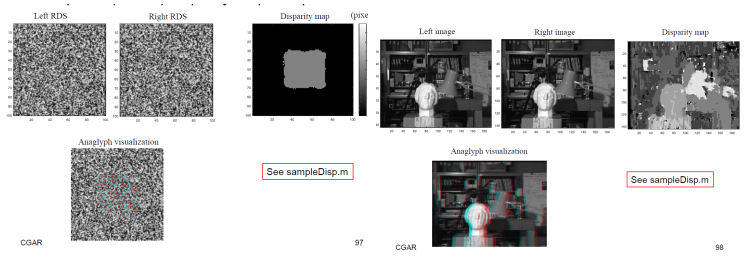
For each pixel (i,j) of the left image and offset o in disparity range , we compute d(i,j,o) and the disparity at (i,j) is the value o that minimizes d.

The result performing this search over every pixel is the disparity map.



2.15 Feature-based correspondence

Restrict search to sparse set of detected features (**keypoints**) Rather than pixel values use feature descriptor and an associated feature distance. Still narrow search by epipolar lines



2.16 Measurement error

Accuracy

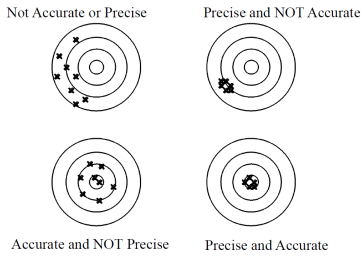
How close is measurement to true valued. Affected by systematic errors. Can be improved with better calibration.

Precision

How closely do multiple measurements agree. Varies per type of sensor . Varies pere degree of freedom. Can be improved with filtering.

Resolution

Minimum difference that can be discriminated betweeen two measurements. Cannot be reached in practice because of noise



2.17 How does a depth camera work?

Passive illumination (standard RGB Camera), natural light sources and visual features and Active illumination (RGB-D Camera) , often infrared spectrum.

2.18 Active stereo

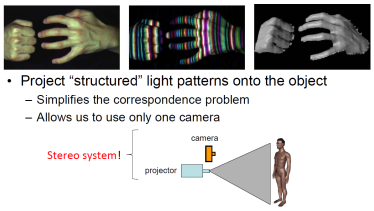
Structured light : project a known pattern into the scene and Projector with regular light of laser.

Laser ranging : Measure time of flight taken by laser pulse

2.18.1 Time of flight

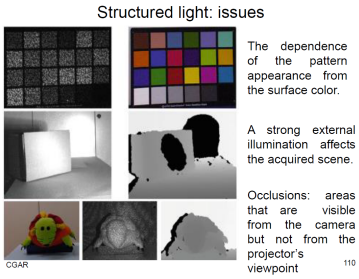
Depth cameras in HoloLens use time of flight. Emit light of a known wavelength, and timme how long it takes for it to come back.

2.18.2 Active stereo with structured light

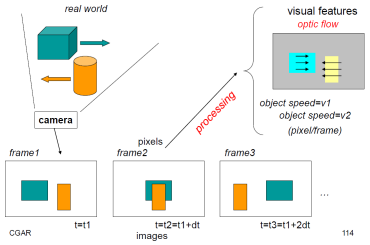


The depth map is constructed by analyzing a speckle pattern of infrared light. Structure light general principle : project a known pattern onto the scene and infer depth from the deformation of that pattern.

The objective of structured light systems is to simplify the correspondence problem through projecting effective patterns by the illuminator : to infer depth from the deformation of that pattern.



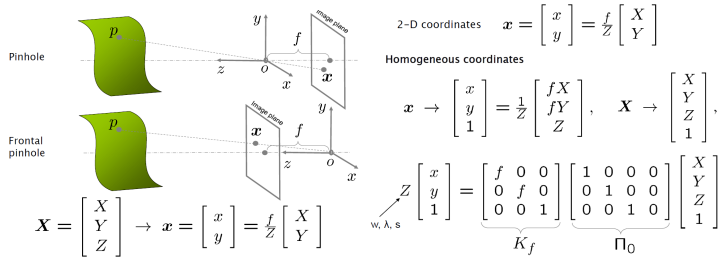
2.19 Motion Computation



Visual motion can be annoying (camera instabilities , jitter). Visual motion indicates dynamics in the scene and reveals spatial layout.

Optical flow : Given two images , find the location of a world point in a second close-by image with no camera info. It is used the KLT algorithm (1.Find corners in first image , 2. Extract intensity patch around each corner , 3. Use Lucas-Kanade algorithm to estimate constant displacement of pixels in patch)

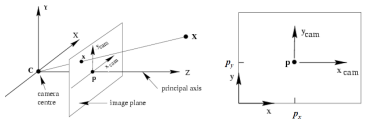
3 Camera Calibration and Epipolar Geometry



Camera calibration : Figuring out

1. Transformation from metric coordinates (camera) to pixels coordinates(image)
2. Transformation from world coordinate system to camera coordinate system.

3.1 Image coordinate system

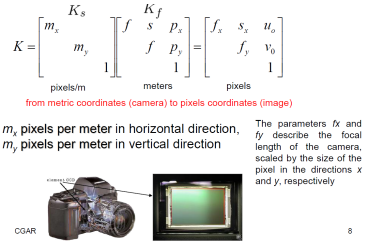


Principal point (p) : point where principal axis, the z-axis , intersects the image plane

Normalized coordinate system : origin of the image is at the principal point : x and y axes of the image plane are parallel to X and Y axes of the camera reference system.

Image coordinate system : origin is in the corner.

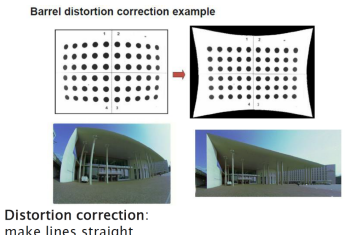
3.2 Pixel Coordinates



3.3 Camera Parameters : Lens distortion Model

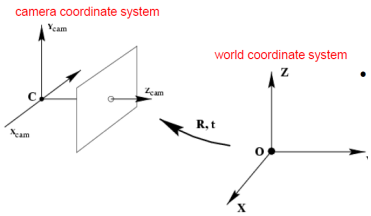
Non linear effects (short focal length and cheap cameras) : Radial and Tangential distortion

Radial distortion example

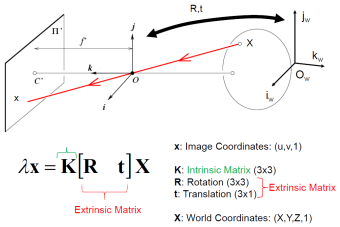


3.4 Camera Rotation and Translation

In general , the camera coordinate frame will be related to the world coordinate frame by a rotation R and a Translation t

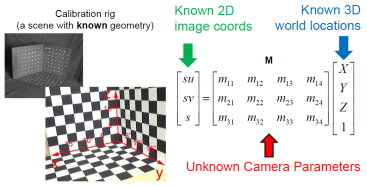


3.5 Camera Projection matrix



3.6 Calibrating the Camera

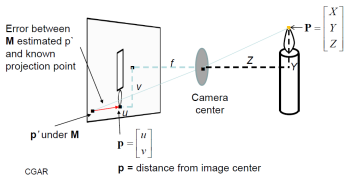
Given n points with known 3D coordinates X_i and known image projections x_i , to estimate the camera parameters (least squares solution).



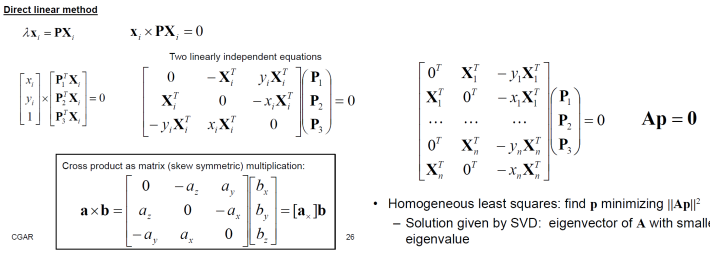
3.6.1 What is least squares doing?

Given 3D point evidence , find best M which minimizes the error between estimate (p') and known corresponding 2D points (p)

Best M occurs when p' = p , or when p' - p = 0. Form these equations from all point evidence. Solve for model via closed-form regression.



3.7 Calibrating the Camera : Tsai method

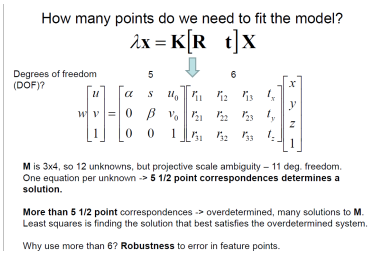


Homogeneous least squares : find p minimizing $||A_p||^2$

If rank(A) $<$ 11, there infinite solutions. Check if data is degenerate. If rank(A) is 11 , solution given by SVD : eigenvector of A with smallest eighenvalue.

In practice the smallest eigenvalue of A will not be exactly equal to zero but will have a small value due to noise.

Rule of thumb : always check the smallest eigenvalue and the ration between the largest and smalles values to estimate noise in the data.



Given n (>= 6) correspondences $x_i -> X_i$ (pixels to world coordinates). Compute P = $K[R|t]$ such that $x_i = PX_i$

But the algorithm for camera calibration has two parts : 1. Compute the matrix P from a set of point correspondences , 2. Decompose P into K , R and t.

3.7.1 Limitations of the linear approach

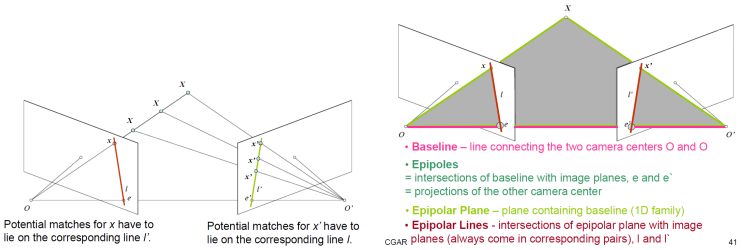
Using least square minimization to get q has little physical meaning. The method ignores constraints on the elements of P. The elements of P are not arbitrary. A more accurate approach is to use constrained non-linear optimization to find the calibration matrix.

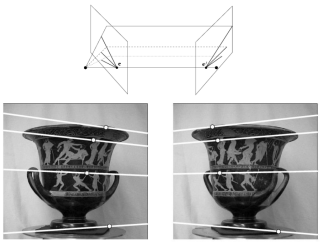
3.8 Epipolar Geometry

We consider the basic geometry that relates images of points to their 3D position. We started with the simplest case of two parallel calibrated cameras. Then, we introduce the basic building blocks of the geometry of two views , know as Epipolar Geometry.

3.8.1 Key idea : Epipolar Constraint

It has been long known in photogrammetry that the coordinates of the projection of a point and the two camera optical centers form a triangle , a fact that can be written as an algebraic constraint involving the camera poses and image coordinates.





3.8.2 What is useful for?

Find x' : if we know x , we can restrict x' to be along the line l'. Compute disparity for stereo.

Given candidate x and x' correspondences estimate relative position and orientation between the cameras.

Model fitting : see if candidate x , x' correspondences fit estimated projection models of cameras 1 and 2.

Given candidate x and x' correspondences , and having calibrated cameras , estimate the 3D position of corresponding image points.

3.9 Estimating the Fundamental Matrix : The eight-point algorithm

1. Least squares solution using SVD on equations from 8 pairs of correspondences
2. Enforce det(F) = 0 constraint using SVD on F , since F must be singular.

We can use the 8-point algorithm also to estimate the Essential Matrix

1. Solve a system of homogeneous linear equations. We want the eigenvector with smalles eigenvalue. We can find the eigenvectors and eigenvalues of $A^T A$ by finding the Singular Value Decomposition of A.
2. Resolve det(F) = 0 constraint by using SVD

The coordinates of corresponding points can have a wide range , thus leading to numerical instabilities. It is better first to normalize them , so they have average 0 and stddev 1 , and then denormalize F at the end.

Estimating the fundamental matrix is known as **Weak calibration**. If we know the calibration matrices of the two cameras , we can estimate the essential matrix The essential matrix gives us the relative rotation and translation between the cameras , or their **extrinsic parameters**

3.10 3D Reconstruction : Stereo reconstruction

Given point correspondences , how to compute 3D point positions using triangulation. Result depends on how calibrated the system is :

1. Intrinsic and extrinsic parameters known , can compute metric 3D geometry

2. Only intrinsic parameters known, can compute 3D geometry up an unknown scale factor

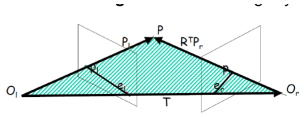
3. Neither intrinsic nor extrinsic known , recover structure up to an unknown projective transformation of the scene.

3.10.1 Fully Calibrated Stereo : Calibrated Triangulation

Known intrinsics : can compute viewing rays in camera coordinate system.

Know extrinsics : know how rays from both cameras are positioned in 3D space.

Reconstruction : triangulation of viewing rays.



ideally, P is the point of intersection of two 3D rays:
ray through O_l with direction P_l
ray through O_r with direction $R^T P_r$

Unfortunately , these rays typically don't intersect due to noise in point locations and calibration parameters.

Solution : Choose P as the Pseudo intersection point. This is point that minimizes the sum of squared distance to both rays.

3.11 3D reconstruction : intrinsics known

We introduce another algorithm that uses the relative pose (rotation and translation) between the two cameras by considering Essential matrix.

Relative pose and point correspondences can then be used to retrieve the position of the points in 3D by recovering their depths relative to each camera frame.

- Consider the basic rigid-body equation, where the pose (R, T) has been recovered, in terms of the **images** and the **depths**, it is given by

λx=RX+t $\quad \lambda_2^j \boldsymbol{x}_2^j = \lambda_1^j R \boldsymbol{x}_1^j + \gamma T, \quad j = 1, 2, \ldots, n.$

- Notice that since (R, T) are known, the equations are linear in both the depth λ's and the ***scale*** γ.

79

- For each point, λ_1, λ_2 are its depths with respect to the first and second camera frames, respectively. One of them is redundant, it is simply a function of (R, T).
- Hence we can eliminate, say, λ_2 from the above equation by multiplying both sides by $\widehat{x}_2$(cross product as matrix), which yields

$$\lambda_1^j \widehat{\boldsymbol{x}}_2^j R \boldsymbol{x}_1^j + \gamma \widehat{\boldsymbol{x}}_2^j T = 0, \quad j = 1, 2, \ldots, n.$$

- This is equivalent to solve

$$M^j \bar{\lambda}^j \doteq \left[\widehat{\boldsymbol{x}}_2^j R \boldsymbol{x}_1^j, \ \widehat{\boldsymbol{x}}_2^j T \right] \begin{bmatrix} \lambda_1^j \\ \gamma \end{bmatrix} = 0,$$

for all *n* equations.

- The linear least squares estimate of $\vec{\lambda}$ is simply the eigenvector of M^TM that corresponds to its smallest eigenvalue.

- Note that this scale ambiguity is intrinsic, since without any prior knowledge about the scene and camera motion, one cannot disambiguate whether the camera moved twice the distance while looking at a scene twice larger but two times further away.
  This scale can be determined if we know the distance between two points in the observed scene or the distance between the two cameras.

4 Homography , SLAM and AR

Now we consider computer vision algorithms for the use in AR. We focus in particular on visual tracking and 3D scene reconstruction.

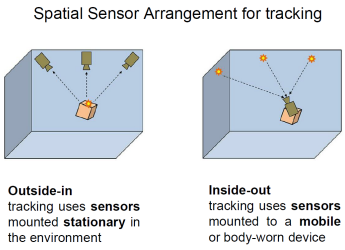
AR necessitates real-time approaches. This requirements is reflected in the subset of computer vision algorithms we consider here.

4.1 Model-Based vs Model-Free Tracking

Using images obtained by a camera requires comparing these images to some reference model. If such a model is obtained prior to starting the tracking system , we refer to the approach as model-based tracking. The alternative is called model-free tracking , a name that is slightly misleading , because a temporary model is actually acquired on the fly during the tracking.

4.1.1 Markers vs Natural Features tracking

We classify tracking targets into those natural features and those with artificial features. The former type can be , for instance , corners or SIFT. The latter type is often called a marker or fiducial.

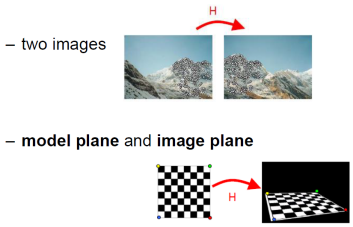


4.2 Marker Tracking

Marker tracking is computationally inexpensive and can deliver useful results even with rather poor cameras. Detecting the four corners of a flat marker in an image from a single calibrated camera delivers just enough information to recover the pose of the relative to the marker.

4.3 Homography

To estimate homography H from point correspondences between planar surfaces :



A homography is a non singular , line preserving , projective mapping H : $P^n -> P^n$ It is represented by a square (n + 1) - dim matrix with $(n+1)^2 - 1$ DOF

We consider a mapping between planes.

4.4 The marker tracking pipeline

1. Capture an image by using a calibrated camera
2. Marker detection by searching for quadrilateral shapes
3. Homography Estimation
4. Pose estimation from homography
5. Pose refinement by nonlinear reprojection error minimization
6. AR rendering with the recovered camera pose.

4.4.1 Marker Detection

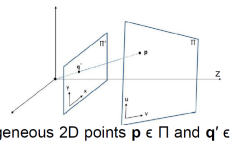
We assume a single input marker.

We compute the four corners of the flat marker.

4.4.2 Homograohy Estimation

For tracking applications , one plane is the model plane , and the other plane is the image plane that contains the known points in the world , such as the marker's corners.

Because the four points are constrained to lie in a plane , they can be expressed with only 2DOF. Consequently , a 3 x 3 matrix with 8DOF is sufficient



- Homogeneous 2D points **p** ϵ Π and **q′** ϵ Π′ can be related by a homography **H** as follows: $\mathrm{p} = \mathrm{Hq}'$

- We assume that the **marker** defines the **plane** Π′: $\mathbf{q}_z = 0$ in *world coordinates*, and that marker corners have the coordinates $[0\ 0\ 0]^T$, $[1\ 0\ 0]^T$, $[1\ 1\ 0]^T$, and $[0\ 1\ 0]^T$.
- We can then express a 3D point **q′** ϵ Π′ as a *homogeneous 2D point* $\mathbf{q}' = [q_x\ q_y\ 1]^T$.
- Mapping from one plane to another can be mathematically modeled as a **homography** defined by a 3 × 3 matrix **H.**

to relate them to the image plane. H can be estimated from 2D-2D correspondences using direct linear transformation.

Because we are using homogeneous coordinates , two points are related by a homography only up to scale. When interpreted as vectors , however , they point in the same direction. Thus , the cross product is zero : p x Hq' = 0

- From one 2D–2D correspondence, we have now obtained three equations in the unknown coefficients of **h**.
- These **equations** are linearly dependent, so we retain only the **first two**.
- We require a minimum of **four** input **points** to determine **eight unknowns**.

- From *N* pairs $\mathbf{p}_i = [p_{i,u}\ p_{i,v}\ p_{i,w}]^T$ and $\mathbf{q}_i = [q_{i,u}\ q_{i,v}\ q_{i,w}]^T$, we set up a $2N \times 9$ matrix:

$$\mathrm{A} = \begin{bmatrix} 0 & 0 & 0 & -p_{1,w}q_{1,u}^T & -p_{1,w}q_{1,v}^T & -p_{1,w}q_{1,w}^T & p_{1,v}q_{1,u}^T & p_{1,v}q_{1,v}^T & p_{1,v}q_{1,w}^T \\ p_{1,w}q_{1,u}^T & p_{1,w}q_{1,v}^T & p_{1,w}q_{1,w}^T & 0 & 0 & 0 & -p_{1,u}q_{1,u}^T & -p_{1,u}q_{1,v}^T & -p_{1,u}q_{1,w}^T \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & -p_{N,w}q_{N,u}^T & -p_{N,w}q_{N,v}^T & -p_{N,w}q_{N,w}^T & p_{N,v}q_{N,u}^T & p_{N,v}q_{N,v}^T & p_{N,v}q_{N,w}^T \\ p_{N,w}q_{N,u}^T & p_{N,w}q_{N,v}^T & p_{N,w}q_{N,w}^T & 0 & 0 & 0 & -p_{N,u}q_{N,u}^T & -p_{N,u}q_{N,v}^T & -p_{N,u}q_{N,w}^T \end{bmatrix}$$

- The homogeneous equation system **A h** = 0, which is overdetermined for $N > 4$, can be solved using singular value decomposition (SVD), i.e. *the eigenvector with smallest eigenvalue.*
- Now we have the **homography estimation**. We can exploit it **to estimate the camera pose**.

4.4.3 Pose Estimation from Homography

Recall that our points lie in the z-plane and assume that K is known : we derive the 3D camera pose from H , since the third column R_C3 of the rotation matrix R has no effect :

- We see that $\mathbf{H} = \mathbf{K}[\mathbf{R}_{C1} \mid \mathbf{R}_{C2} \mid \mathbf{t}]$. Therefore, the camera pose can be computed from

  $\mathbf{H}^K = \mathrm{K}^{-1}\,\mathrm{H}$, thus $\mathbf{H}^K = [\mathbf{R}_{C1} \mid \mathbf{R}_{C2} \mid \mathbf{t}]$

  by recovering the third column of the rotation matrix as the cross-product of the first two columns: $\mathbf{R}_{C1} \times \mathbf{R}_{C2}$

- However, the first two columns of $\mathbf{H}^K$ will usually not be truly orthonormal *because of noise* in the point correspondences, and we know them *up to a scale*. Therefore, a proper **normalization** needs to be **enforced**.

- To enforce a proper **normalization,** we consider

  $$\boldsymbol{K}^{-1}\boldsymbol{H} = \lambda\,[\boldsymbol{r}_1\boldsymbol{r}_2\boldsymbol{t}]$$

- r_1 and r_2 are unit vectors -> **find lambda**
- Use this **to compute t**
- Rotation matrices are **orthogonal** -> find r_3 as cross product between r_1 and r_2

- Thus, we have

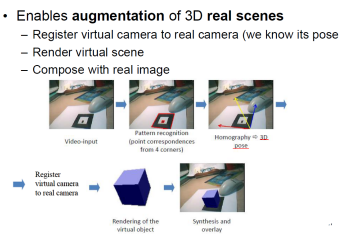
  $$\boldsymbol{P} = \boldsymbol{K}\begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & (\boldsymbol{r}_1 \times \boldsymbol{r}_2) & \boldsymbol{t} \end{bmatrix}$$

Problem : The vectors r1 and r2 might not yield the same lambda
Solution : use the average value
Problem : The estimated rotation matrix might not be orthogonal
Solution : orthogonalize R'

4.4.4 Pose Refinement

Pose estimation cannot always be computed directly from imperfect point correspondences with desired accuracy Therefore , the pose estimation is refined by iteratively minimizing the reprojection error. When a first estimate of the

camera pose is known , we minimize the displacemetn of the known points q_i in 3D , projected using [R— t] from its known image location p_i



We can calibrate a camera by considering several images of a known planar pattern , such as a checkerboard.

We can follow the same approach used for the homography estimation , but now we have to estimate the matrix K too.

Thus , first we estimate H by using th 4-point algorithm , then we can obtain two equations that the calibration matrix K has to satisfy.

Since K has 5 parameters , we need at least three different images of the checkerboard.

4.5 Stereo (or multiple) camera tracking

Tracking uses an outside-in setup with multiple infrared cameras.

A minimum of two cameras in a known configuration , a calibrated stereo camera rig is required.

For tracking arbitrary objects , we require general pose estimation , which addresses the probelm of determining the camera pose from 2D-3D correspondences.

We describe an infrared tracking system designed also to track rigid body markers composed of four or more retro-reflective spheres.

4.6 The stereo camera tracking pipeline

1. Blob detection in all images to locate the spheres of the rigid body markers.
2. Establishment of point correspondences between blobs using epipolar geometry between the cameras.
3. Triangulation to obtain 3D candidate points from the multiple 2D points
4. Matching of 3D candidate points to 3D target points-
5. Determination of the target's pose using absolute orientation.

Blob detection : it is simplified , since the targets are composed of spheres covered with retro-reflective foil

Establishing Point Correspondences : The candidate 2D points in the two images p1 and p2 can be related using epipolar lines. Since the system is calibrated , we can use the Essential matrix.

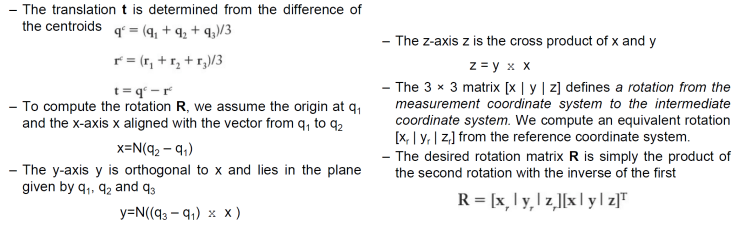
Triangulation from two cameras : We can perform , a 3D reconstruction from multiple 2D points , since the system is calibrated.

Matching Targets Consisting of Spherical Markers : There may be more candidte points than target points because of ambiguous observations. The association from candidate points to target points is resolved using the known geometric structure of the target

Absolute Orientation : After candidate point association , we are left with two sets of corresponding points , the observed points q_i and the target points r_i

The target points are specified in a reference coordinate system , and we would like to compute the pose of the observed target relative to the reference coordinate system.

It requires at least three points. The centroid of the three points can be used to determine the translation from the reference coordinate system to the measurement coordinate system. The rotation is computed from two parts. First , we define a rotation from the measurement coordinate system into an intermediate coordinate system defined by the q_i Second , we di the same for r_i. Finally we concatenate the two rotations to obtain R



4.7 Natural Feature Tracking

In the previous two case studies , we have considered artificial markers.

Here, we introduce the use of natural feature tracking to determine the camera pose from observations in the image without instrumenting the environment with markers.

We consider monocular tracking with a single camera. We describe tracking by detection : the camera pose is determined from matching interest points in every frame anew , without relying on prior information gleaned from previous frames.

4.7.1 Pipeline for natural feature tracking

A typical pipeline for tracking by detection of sparse interest point consists of five stages :

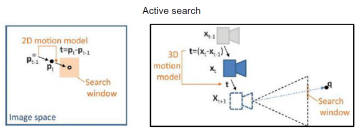
1. Interest point detection
2. Descriptor creation
3. Descriptor matching
4. Perspective-n-Point camera pose
5. Robust pose estimation (RANSAC Algorithm) : To estimate the model parameters x from a randomly chosen subset of data points. For every one of the remaining , potentially many , point correspondences , we compute the residual error, by assuming the camera pose computed. A data point with a residual smaller than a threshold counts as in inlier. If the ratio of inliers to outliers is not sufficient , the procedure is repeated.

4.8 Incremental Tracking

Since AR requires teal-time update rates , neither the camera pose nor the projection of feature points to the image will change drastically from one frame to the next. Tracking by detection ignores this coherence , so that the tracking problem becomes harder to solve than necessary. A tracking system that uses information from a previous step is said to use incremental tracking or recursive tracking.

If the last tracking iteration was successful , there is good reason to believe that we can be successful again by searching for the inliers from the last frame and searching close to their last known positions.

Incremental tracking requires two components : An interest point matching component and an incremental search component.



In the simplest case, if **no 3D tracking model** is available, a motion model can be obtained purely from the 2D positions of interest points.

If a **3D tracking model** can be obtained, a corresponding 3D motion model will usually give a better result.

The classic approach to incremental tracking is the Kanade-Lucas-Tomasi tracker, which extract keypoints from an initial image and then tracks them using optical flow.

Hierarchical Search :

It is usually sufficient to employ a simple image pyramid. Only a small number of strong features is tracked at this resolution using the predicted camera pose from the motion model.

The camera pose resulting from this coarse tracking step is not yet accurate enough , it is adequate for initialization of tracking at the full resolution by using a much smaller search window.

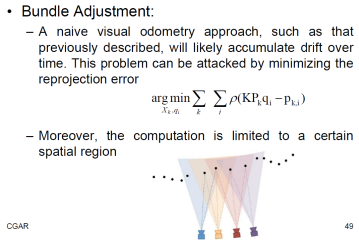
4.9 Simultaneous Localization and Mapping

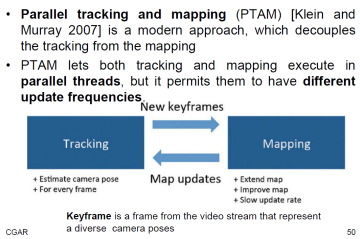
Localization : means continuous 6DOF tracking of a camera pose relative to an arbitrary starting point.

Mapping: is to create a map using consistent data association of observation to points in the scene.

4.9.1 SLAM pipeline

1. Detect interest points in a frame
2. Track the interest point in 2D from the previous frame
3. Determine the essential matrix between the current and previous frames
4. Recover the incremental camera pose from the essential matrix
5. The essential matrix determines the translation part of the pose only up to scale , but it must be consistent throughout the tracked image sequence. Thus , 3D point locations are triangulated from multiple 3D observations of the same image feature over time
6. Proceed to the next frame





4.10 Outdoor Tracking

The tracking methods we have described so far are primarily intended for indoor use.

Outdoor tracking is generally more difficult than indoor tracking for the following reasons :

- Mobility : the user is free to go anywhere. The algorithm has to run on mobile devices
- Environment : Many areas with poor or unusable textures. Variations can quickly make any tracking model outdated.
- Localization database : The tracking model cna grow very large
- The user : in general , we cannot expect a naive user of an AR system to understand the system's operation in depth.