Computer Vision

Computer Science & Information Technology
Yung-Yu Chuang
2009/04/10

with slides by Li Zhang, Szymon Rusinkiewicz, Srinivasa Narasimhan

What is computer vision ?

- The goal of computer vision is to write computer programs that can interpret images and understand the scene. The holy grail is to mimic human vision system.

OUTPUT

INPUT		descriptions	images
	descriptions		Computer Graphics
	images	Computer Vision	Image Processing

Can computer match human perception?

- Yes and no (but mostly no!)
  - computers can be better at “easy” things
  - humans are much better at “hard” things

Computer vision vs Human Vision

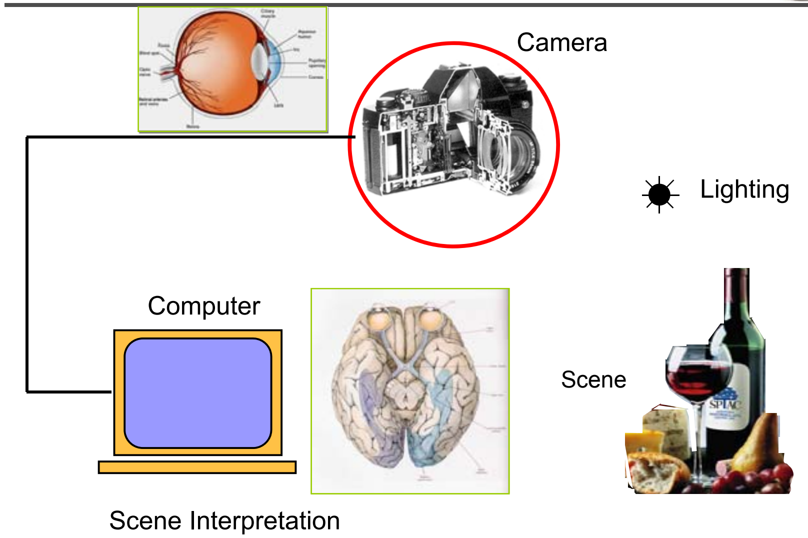
What we see

0	3	2	5	4	7	6	9	8
3	0	1	2	3	4	5	6	7
2	1	0	3	2	5	4	7	6
5	2	3	0	1	2	3	4	5
4	3	2	1	0	3	2	5	4
7	4	5	2	3	0	1	2	3
6	5	4	3	2	1	0	3	2
9	6	7	4	5	2	3	0	1
8	7	6	5	4	3	2	1	0

What a computer sees

Srinivasa Narasimhan’s slide

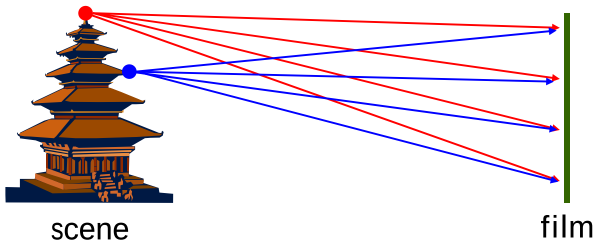
Components of a computer vision system



Srinivasa Narasimhan's slide

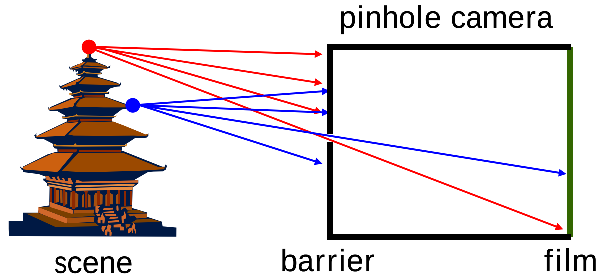
Camera

Camera trial #1



Put a piece of film in front of an object.

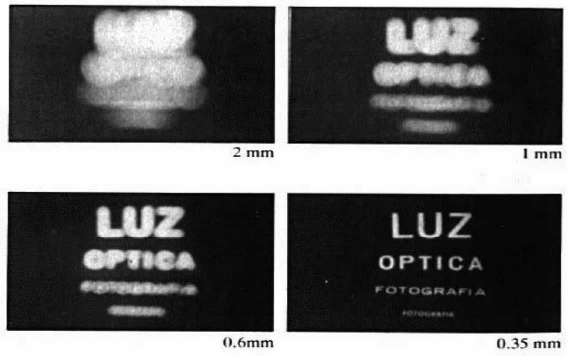
Pinhole camera



Add a barrier to block off most of the rays.

- It reduces blurring
- The pinhole is known as the aperture
- The image is inverted

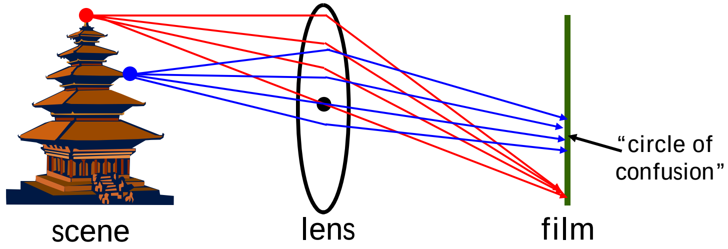
Shrinking the aperture



Why not making the aperture as small as possible?

- Less light gets through
- Diffraction effect

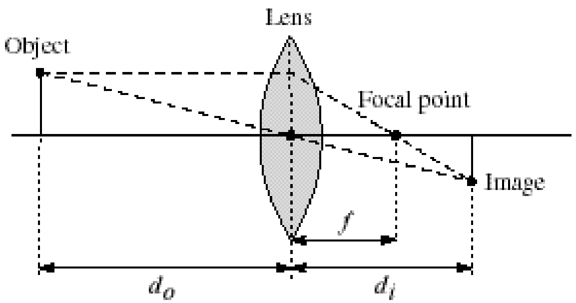
Adding a lens



A lens focuses light onto the film

- There is a specific distance at which objects are "in focus"
- other points project to a "circle of confusion" in the image

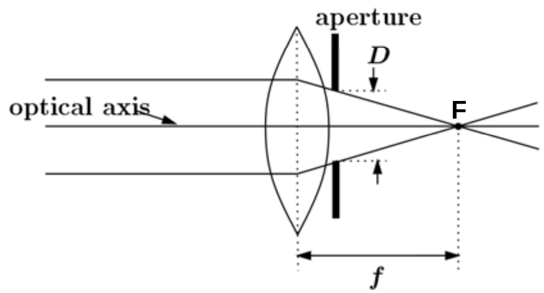
Lenses



Thin lens equation: $\frac{1}{d_o} + \frac{1}{d_i} = \frac{1}{f}$

- Any object point satisfying this equation is in focus
- Thin lens applet: http://www.phy.ntnu.edu.tw/java/Lens/lens_e.html

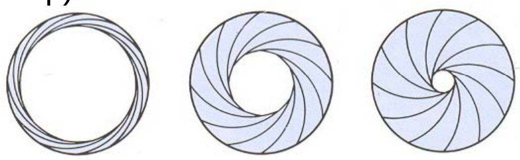
Exposure = aperture + shutter speed



- Aperture of diameter D restricts the range of rays (aperture may be on either side of the lens)
- Shutter speed is the amount of time that light is allowed to pass through the aperture

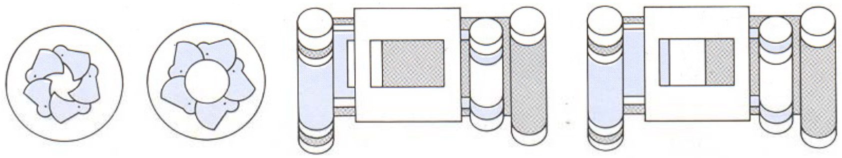
Exposure

- Two main parameters:
  - Aperture (in f stop)



Full aperture Medium aperture Stopped down

  - Shutter speed (in fraction of a second)



Blade (closing) Blade (open) Focal plane (closed) Focal plane (open)

Exposure

- Two main parameters:
  - Aperture (in f stop)
  - Shutter speed (in fraction of a second)
- Reciprocity

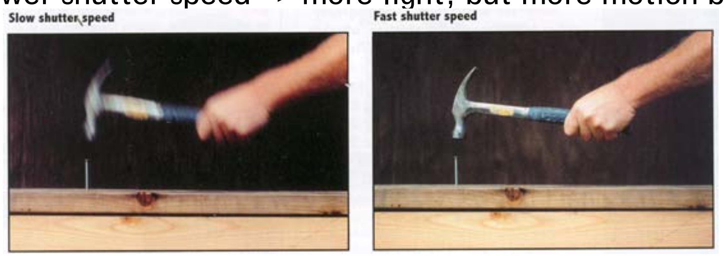
  The same exposure is obtained with an exposure twice as long and an aperture *area* half as big

  - Hence square root of two progression of f stops vs. power of two progression of shutter speed
  - Reciprocity can fail for very long exposures

From Photography, London et al.

Effects of shutter speeds

- Slower shutter speed => more light, but more motion blur



From Photography, London et al.

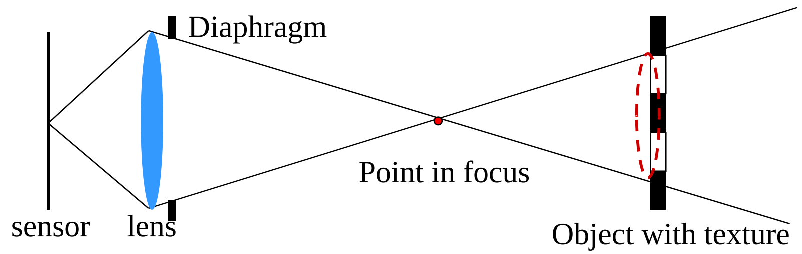
- Faster shutter speed freezes motion

Walking people Running people Car Fast train

1/125 1/250 1/500 1/1000

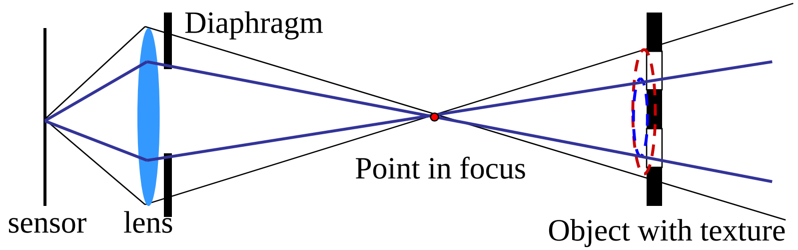
Depth of field

Changing the aperture size affects depth of field. A smaller aperture increases the range in which the object is approximately in focus

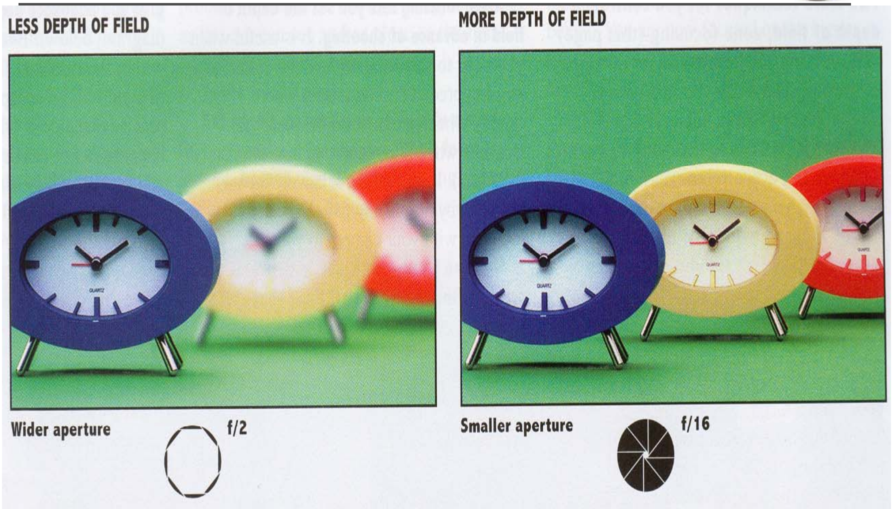


Depth of field

Changing the aperture size affects depth of field. A smaller aperture increases the range in which the object is approximately in focus

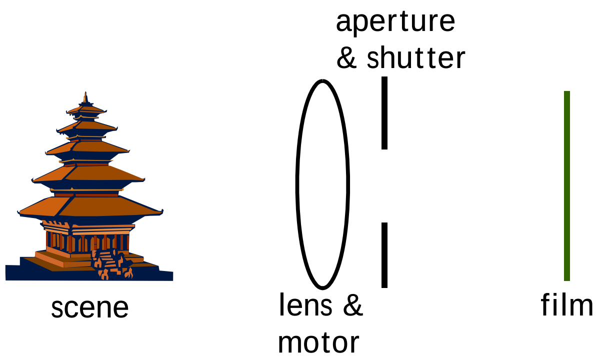


Depth of field

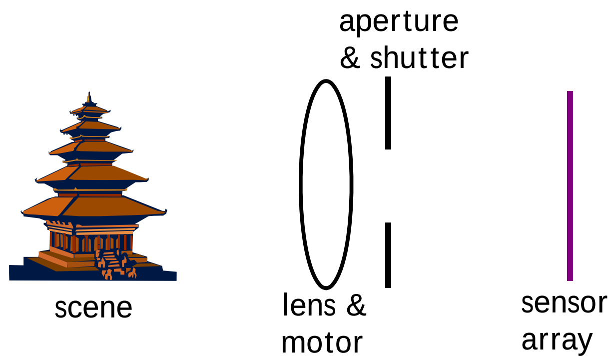


From Photography, London et al.

Film camera



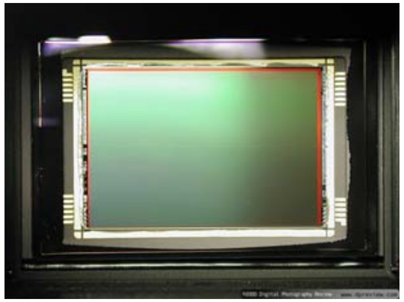
Digital camera



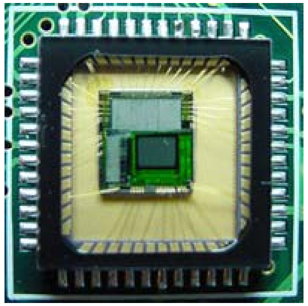
- A digital camera replaces film with a sensor array
- Each cell in the array is a light-sensitive diode that converts photons to electrons

CCD v.s. CMOS

- CCD is less susceptible to noise (special process, higher fill factor)
- CMOS is more flexible, less expensive (standard process), less power consumption



CCD

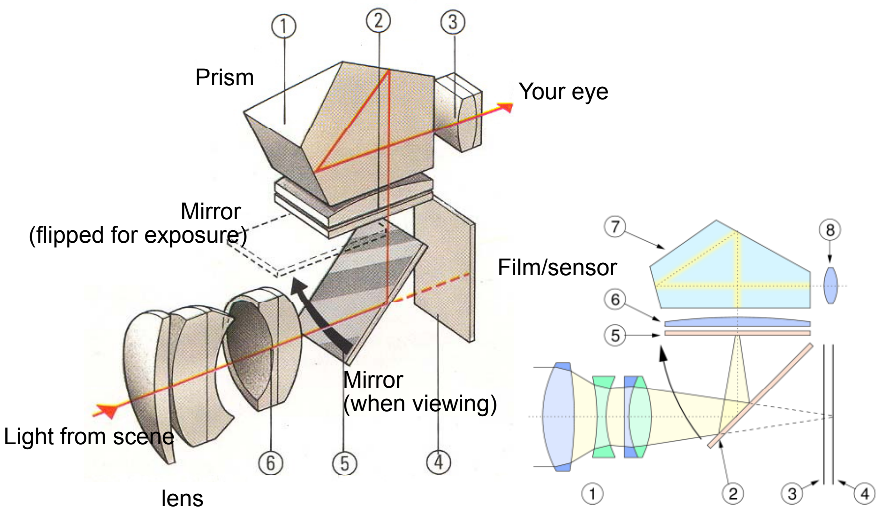


CMOS

SLR (Single-Lens Reflex)

- Reflex (R in SLR) means that we see through the same lens used to take the image.
- Not the case for compact cameras

SLR view finder

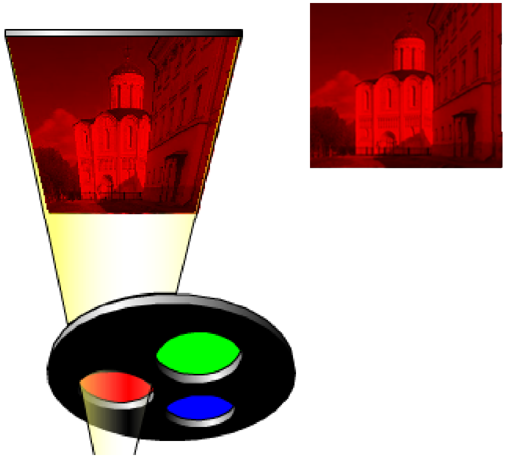


Color

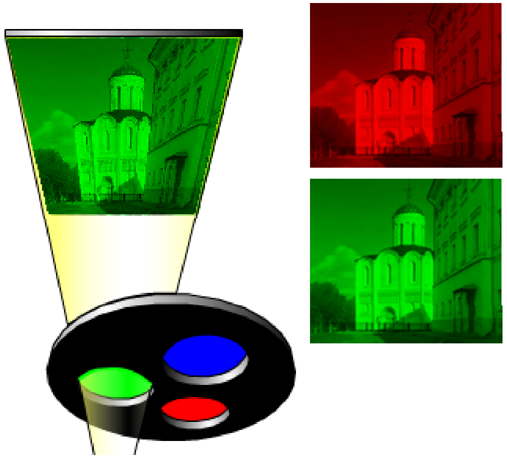
So far, we've only talked about monochrome sensors. Color imaging has been implemented in a number of ways:

- Field sequential
- Multi-chip
- Color filter array
- X3 sensor

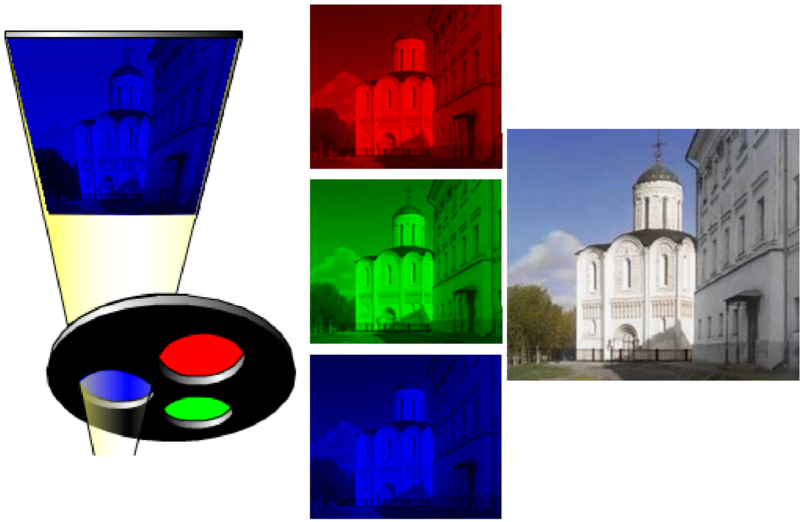
Field sequential



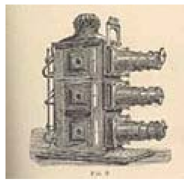
Field sequential



Field sequential



Prokudin-Gorskii (early 1900's)

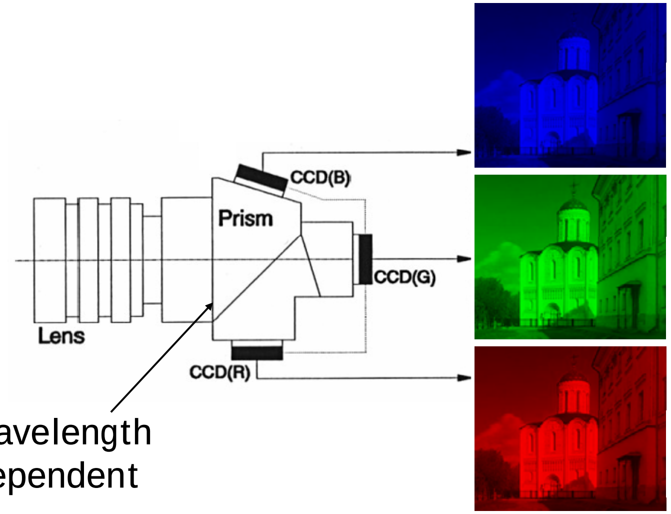


Lantern projector

http://www.loc.gov/exhibits/empire/

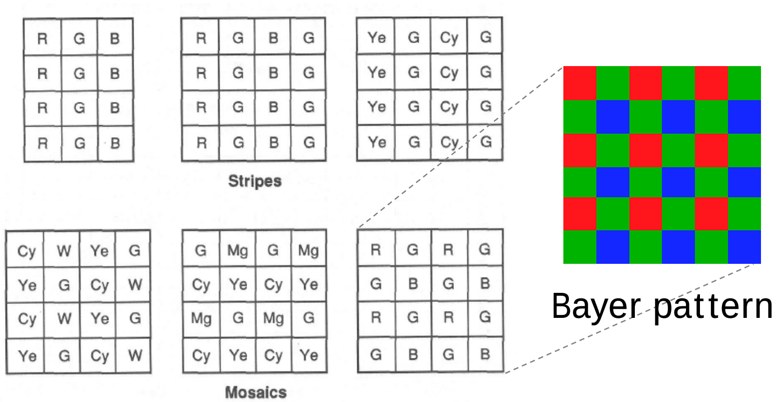
Prokudin-Gorskii (early 1990's)

Multi-chip



wavelength dependent

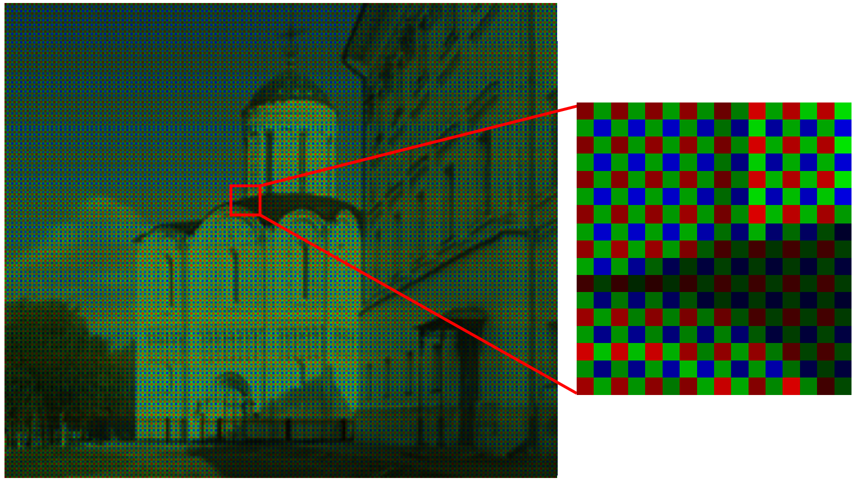
Color filter array



Bayer pattern

Color filter arrays (CFAs)/color filter mosaics

Bayer's pattern



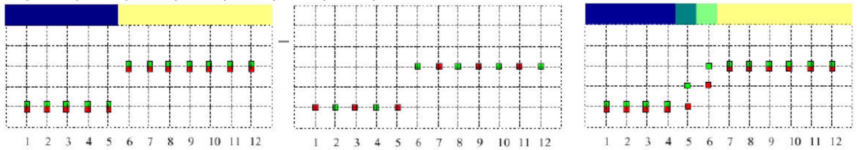
Demosaicking CFA’s

R_{11}	G_{12}	R_{13}	G_{14}	R_{15}	G_{16}	R_{17}
G_{21}	B_{22}	G_{23}	B_{24}	G_{25}	B_{26}	G_{27}
R_{31}	G_{32}	R_{33}	G_{34}	R_{35}	G_{36}	R_{37}
G_{41}	B_{42}	G_{43}	B_{44}	G_{45}	B_{46}	G_{47}
R_{51}	G_{52}	R_{53}	G_{54}	R_{55}	G_{56}	R_{57}

bilinear interpolation

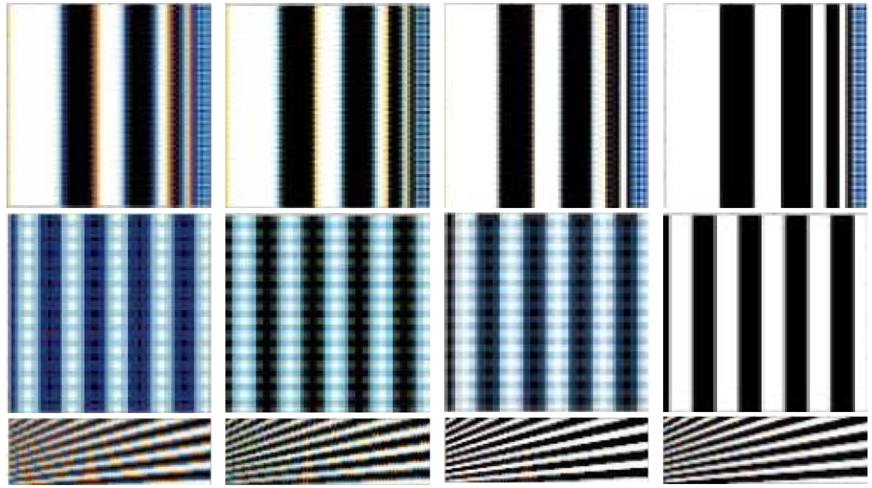
$$G_{44}=(G_{34}+G_{43}+G_{45}+G_{54})/4$$

$$R_{44}=(R_{33}+R_{35}+R_{53}+R_{55})/4$$



original input linear interpolation

Demosaicking CFA’s



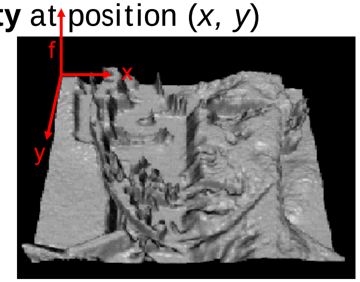
bilinear Cok Freeman LaRoche

Digital camera review website

- [A cool video of digital camera illustration]
- [http://www.dpreview.com/]

Now, we have images

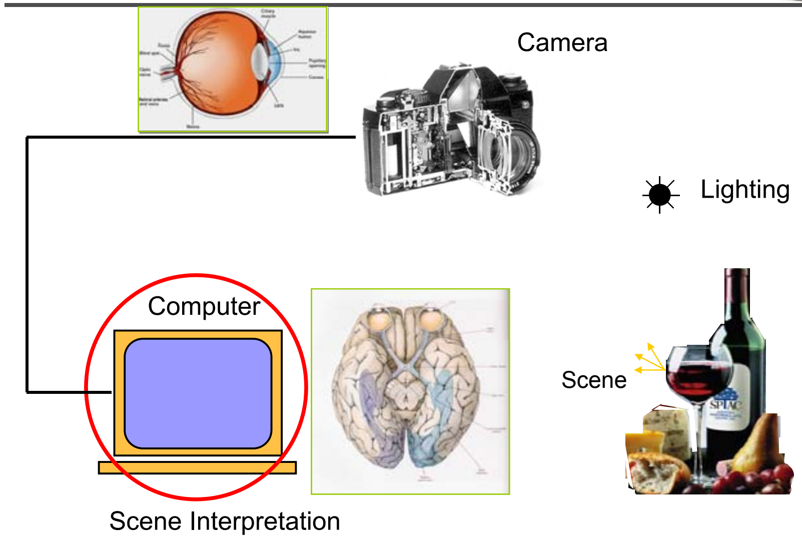
- We can think of an **image** as a function, f: $R^2 \rightarrow R$:
  - $f(x, y)$ gives the **intensity** at position (x, y)



62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

- What about color images?

Write a program to interpret images



Srinivasa Narasimhan's slide

Computer vision programs

Low-level vision (early vision)

- Considers local properties of an image

"There's an edge!"

Mid-level vision

- Grouping and segmentation

"There's an object and a background!"

High-level vision

"It's a chair!"

- Recognition

Low-level vision

Detection

- Edges
- Lines
- Corners

Image filtering

- Convolution with a mask

62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

0	0	1	0	0
0	1	1	1	0
1	1	1	1	1
0	1	1	1	0
0	0	1	0	0

Image filtering (motion blur)

- Convolution with a mask

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

Image filtering (sharpening)

- Convolution with a mask

0	0	0	0	0
0	-1	-1	-1	0
0	-1	9	-1	0
0	-1	-1	-1	0
0	0	0	0	0

Demo with PaintShop Pro

Gaussian filters

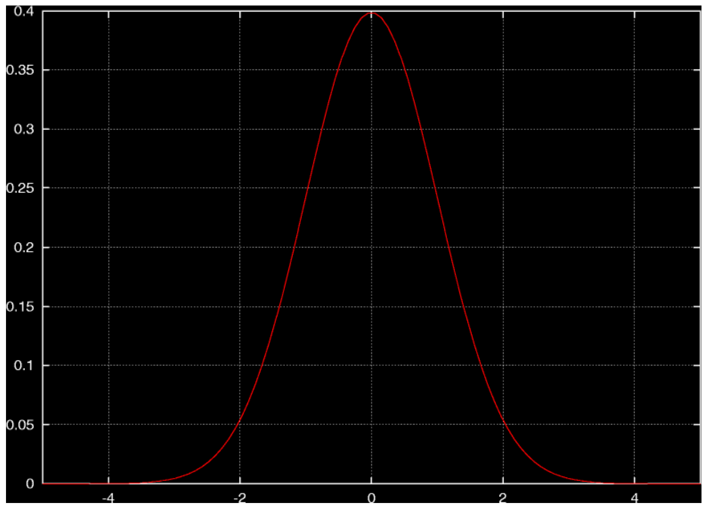
- One-dimensional Gaussian

$$G_1(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$$

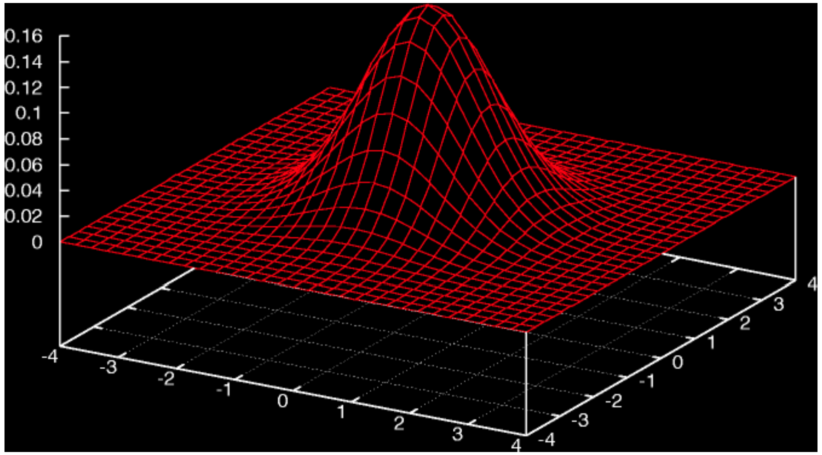
- Two-dimensional Gaussian

$$G_2(x, y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Gaussian filters



Gaussian filters



Computing Discrete Convolutions

$$Out(x,y) = \sum_i \sum_j f(i,j) \cdot In(x-i, y-j)$$

- If *In* is $n \times n$, f is $m \times m$, takes time $O(m^2 n^2)$
- OK for small filter kernels, bad for large ones

Example: smoothing

Original

Smoothed with Gaussian kernel

Canny edge detector

- Smooth
- Find derivative
- Thresholding
- Thinning

Canny edge detector

- First, smooth with a Gaussian of some width σ

Original Image

blurred Image

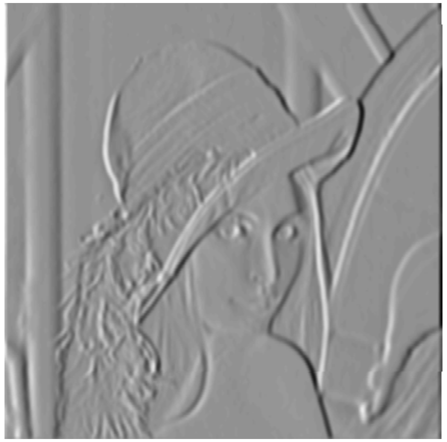
Canny edge detector

- Next, find "derivative"
- What is derivative in 2D? Gradient:

$$\nabla f(x,y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)$$

$$\frac{1}{2}\begin{array}{|c|c|c|c|c|}\hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & -1 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline \end{array} \qquad \frac{1}{2}\begin{array}{|c|c|c|c|c|}\hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & -1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline \end{array}$$

Canny edge detector



Horizontal gradient

Vertical gradient

Canny edge detector

Original Image

Smoothed Gradient Magnitude

Canny edge detector

- thresholding

Original Image

Threshold Gradient Magnitude

Canny edge detector

- Thinning

Original Image

Edges

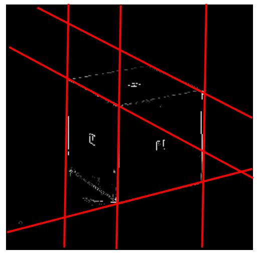
Canny edge detector

- Nonmaximum suppression
  - Eliminate all but local maxima in *magnitude* of gradient
  - At each pixel look along *direction* of gradient: if either neighbor is bigger, set to zero
  - In practice, quantize direction to horizontal, vertical, and two diagonals
  - Result: "thinned edge image"

[Canny demo]

Detecting lines

- What is the difference between line detection and edge detection?
  - Edges = local
  - Lines = nonlocal
- Line detection usually performed on the output of an edge detector



Hough transform

- General idea: transform from image coordinates to parameter space of feature
  - Need parameterized model of features
  - For each pixel, determine all parameter values that might have given rise to that pixel; vote
  - At end, look for peaks in parameter space

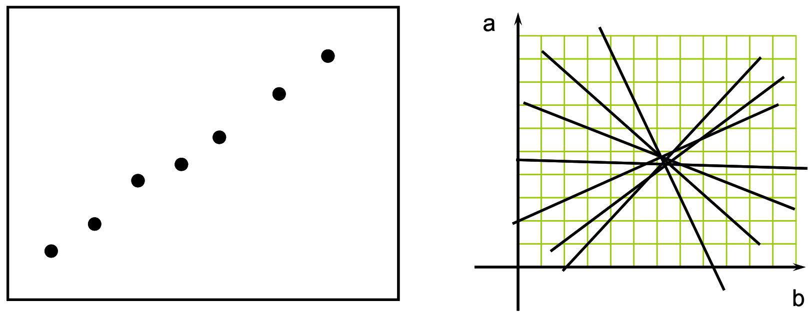
Hough transform for lines

- Generic line: $y = ax+b$
- Parameters: a and b

Hough transform for lines

1. Initialize table of *buckets*, indexed by a and b, to zero
2. For each detected edge pixel (x,y):
   a. Determine all (a,b) such that $y = ax+b$
   b. Increment bucket (a,b)
3. Buckets with many votes indicate probable lines

Hough transform for lines

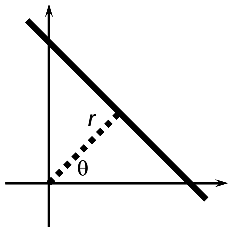


Issues

- Slope / intercept parameterization not ideal
  - Non-uniform sampling of directions
  - Can't represent vertical lines
- Angle / distance parameterization
  - Line represented as (r, θ) where $x \cos \theta + y \sin \theta = r$

Hough transform demo

Hough transform demo2



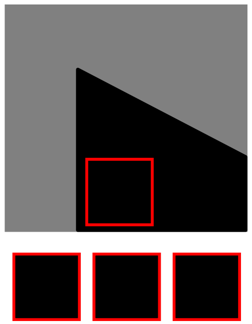
Detection of corners

- Also known as featurs, interesting points, salient points or keypoints. Points that you can easily point out their correspondences in multiple images using only local information.

Moravec corner detector (1980)

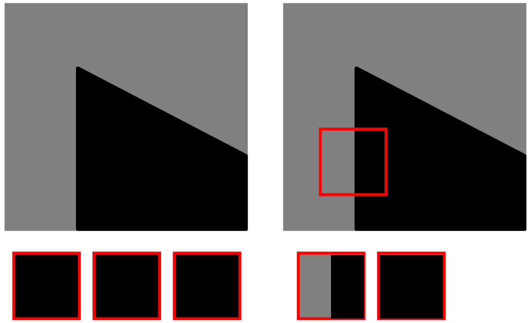
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity

Moravec corner detector



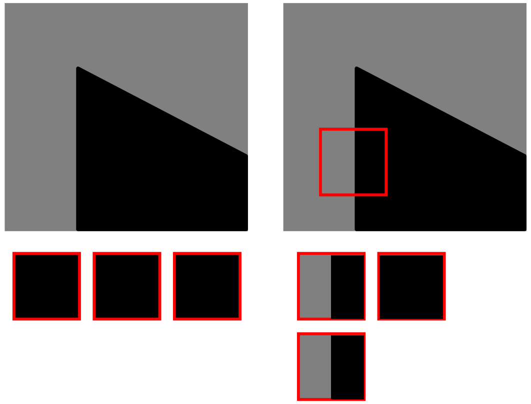
flat

Moravec corner detector



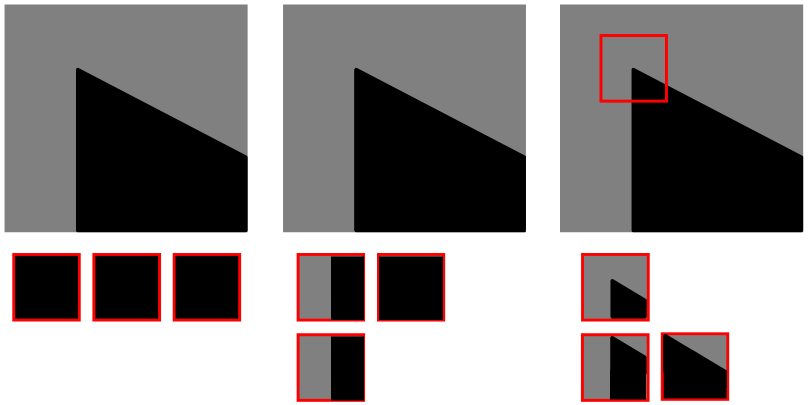
flat

Moravec corner detector



flat edge

Moravec corner detector

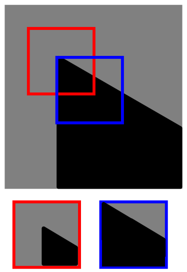


flat edge corner
isolated point

Moravec corner detector

Change of intensity for the shift [*u*,*v*]:

$$E(u,v)=\sum_{x,y}\underbrace{w(x,y)}_{\text{window function}}\Big[\underbrace{I(x+u,y+v)}_{\text{shifted intensity}}-\underbrace{I(x,y)}_{\text{intensity}}\Big]^2$$



Window function $w(x,y)$ =

Four shifts: (u,v) = (1,0), (1,1), (0,1), (-1, 1)
Look for local maxima in *min{E}*

Problems of Moravec detector

- Noisy response due to a binary window function
- Only a set of shifts at every 45 degree is considered
- Only minimum of E is taken into account

⇨ Harris corner detector (1988) solves these problems.

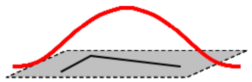
Harris corner detector

Noisy response due to a binary window function

➢ Use a Gaussian function

$$w(x,y) = \exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$$

Window function $w(x,y)$ =



Gaussian

Harris corner detector

Only a set of shifts at every 45 degree is considered

➢ Consider all small shifts by Taylor's expansion

Harris corner detector

Only a set of shifts at every 45 degree is considered

➢ Consider all small shifts by Taylor's expansion

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

$$E(u,v) = Au^2 + 2Cuv + Bv^2$$

$$A = \sum_{x,y} w(x,y) I_x^2(x,y)$$

$$B = \sum_{x,y} w(x,y) I_y^2(x,y)$$

$$C = \sum_{x,y} w(x,y) I_x(x,y) I_y(x,y)$$

Harris corner detector

Equivalently, for small shifts [*u*,*v*] we have a *bilinear* approximation:

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

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

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

Harris corner detector

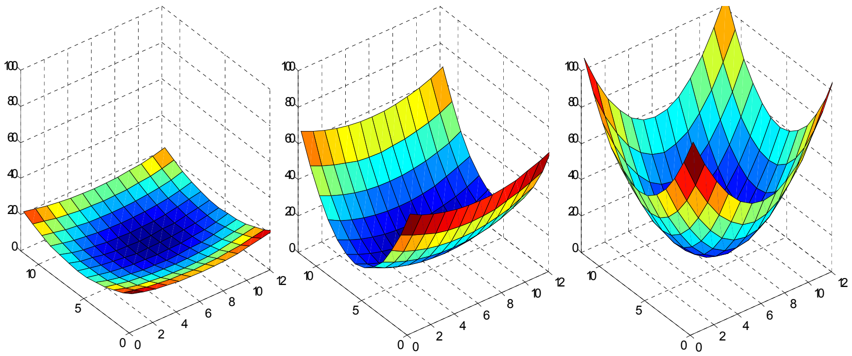
Only minimum of E is taken into account

➢A new corner measurement by investigating the shape of the error function

$\mathbf{u}^T\mathbf{M}\mathbf{u}$ represents a quadratic function; Thus, we can analyze E's shape by looking at the property of $\mathbf{M}$

Harris corner detector

High-level idea: what shape of the error function will we prefer for features?



flat edge corner

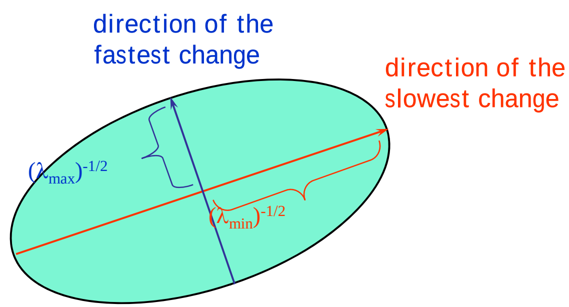
Harris corner detector

Intensity change in shifting window: eigenvalue analysis

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

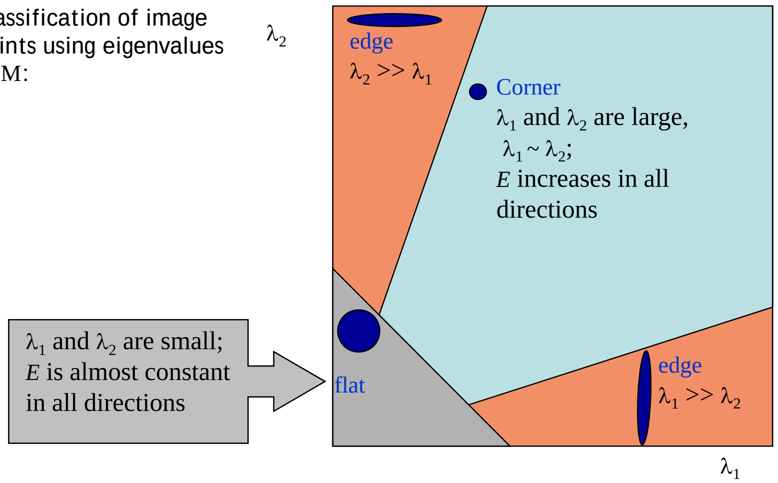
λ_1, λ_2 – eigenvalues of $\mathbf{M}$

Ellipse *E(u,v)* = const



Harris corner detector

Classification of image points using eigenvalues of M:



Harris corner detector

$$\lambda = \frac{a_{00} + a_{11} \pm \sqrt{(a_{00} - a_{11})^2 + 4a_{10}a_{01}}}{2}$$

Only for reference, you do not need them to compute R

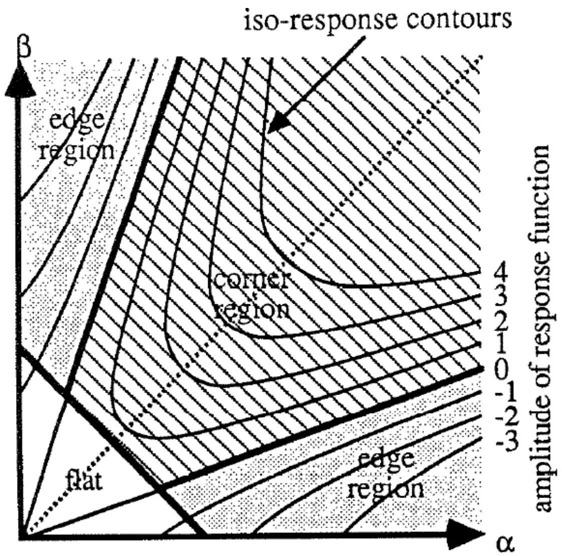
Measure of corner response:

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

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

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

Harris corner detector



Summary of Harris detector

1. Compute x and y derivatives of image

$$I_x = G_\sigma^x * I \qquad I_y = G_\sigma^y * I$$

2. Compute products of derivatives at every pixel

$$I_{x^2} = I_x \cdot I_x \qquad I_{y^2} = I_y \cdot I_y \qquad I_{xy} = I_x \cdot I_y$$

3. Compute the sums of the products of derivatives at each pixel

$$S_{x^2} = G_{\sigma'} * I_{x^2} \qquad S_{y^2} = G_{\sigma'} * I_{y^2} \qquad S_{xy} = G_{\sigma'} * I_{xy}$$

Summary of Harris detector

4. Define the matrix at each pixel

$$M(x,y) = \begin{bmatrix} S_{x^2}(x,y) & S_{xy}(x,y) \\ S_{xy}(x,y) & S_{y^2}(x,y) \end{bmatrix}$$

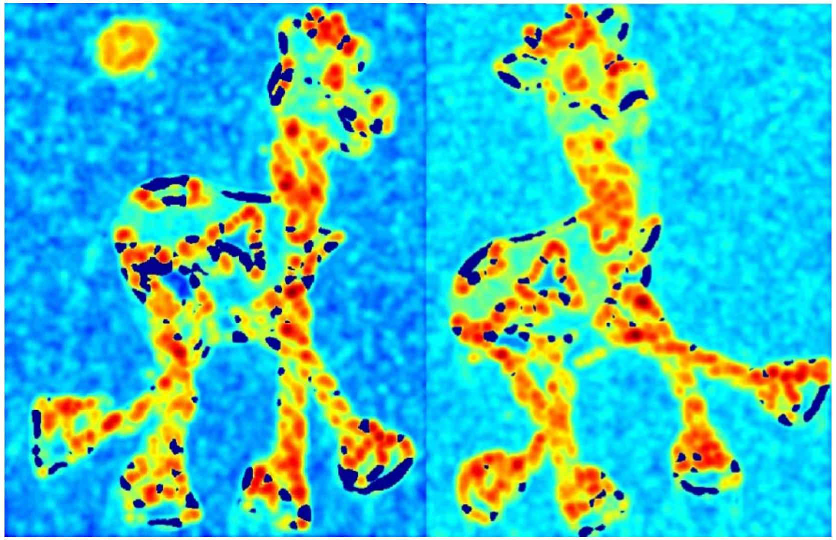
5. Compute the response of the detector at each pixel

$$R = \det M - k(\mathrm{trace} M)^2$$

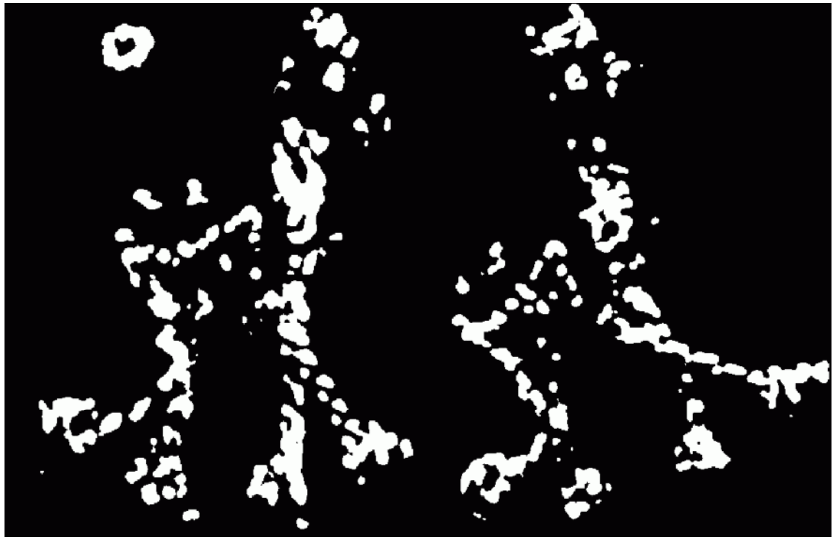
6. Threshold on value of R; compute nonmax suppression.

Harris corner detector (input)

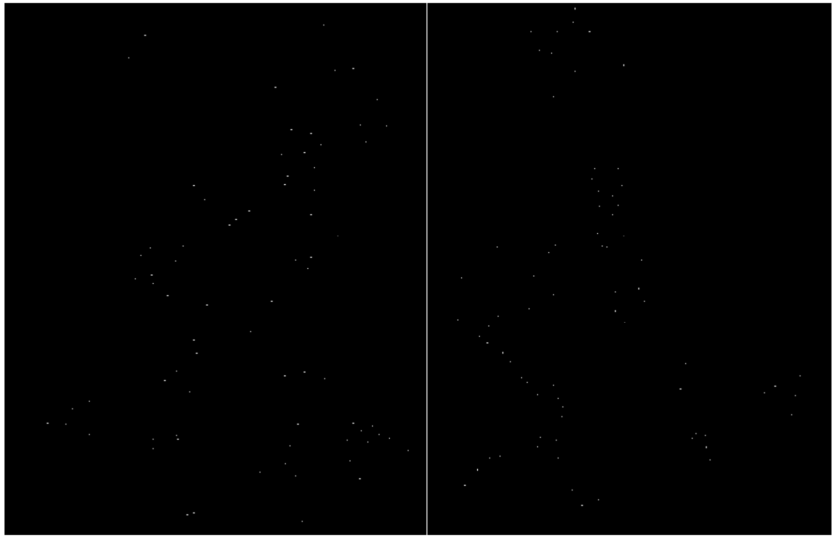
Corner response R



Threshold on R



Local maximum of R



Harris corner detector

Mid-level vision

Segmentation and clustering

- Defining regions
  - Should they be compact? Smooth boundary?
- Defining similarity
  - Color, texture, motion, …
- Defining similarity of regions
  - Minimum distance, mean, maximum

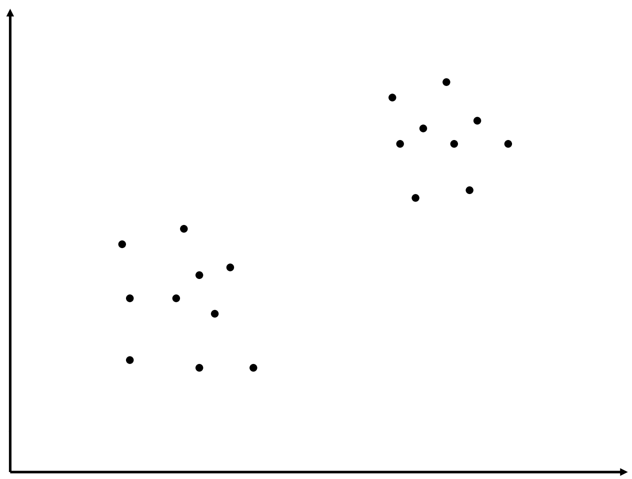
Clustering based on color

- Let’s make a few concrete choices:
  - Arbitrary regions
  - Similarity based on color only
  - Similarity of regions = distance between mean colors

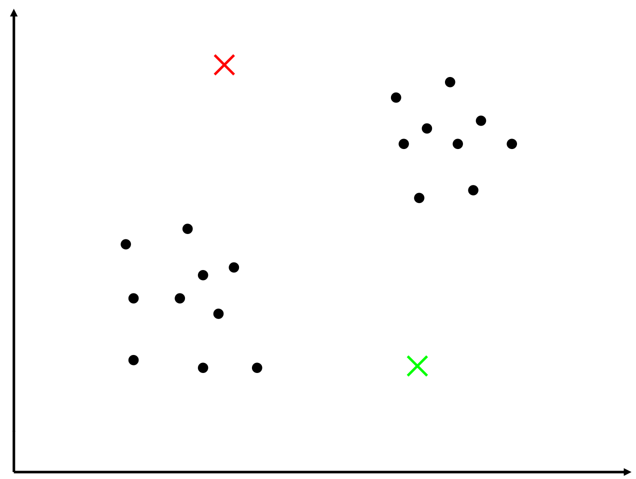
k-means Clustering

1. Pick number of clusters *k*
2. Randomly scatter *k* “cluster centers” in color space
3. Repeat:
   a. Assign each data point to its closest cluster center
   b. Move each cluster center to the mean of the points assigned to it

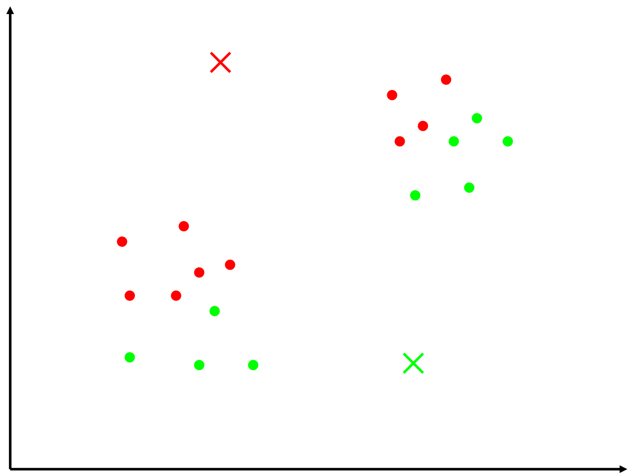
k-means Clustering



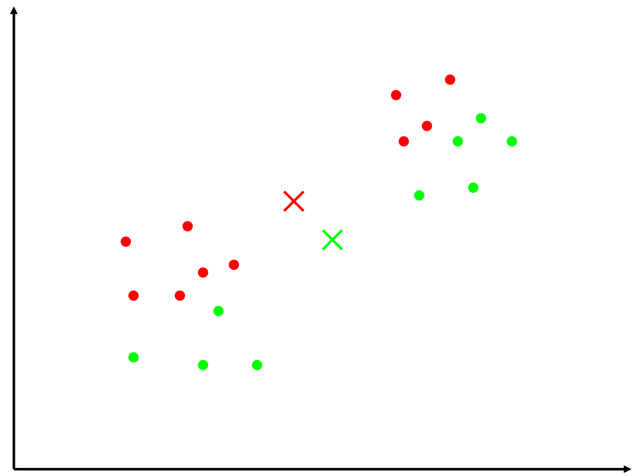
k-means Clustering



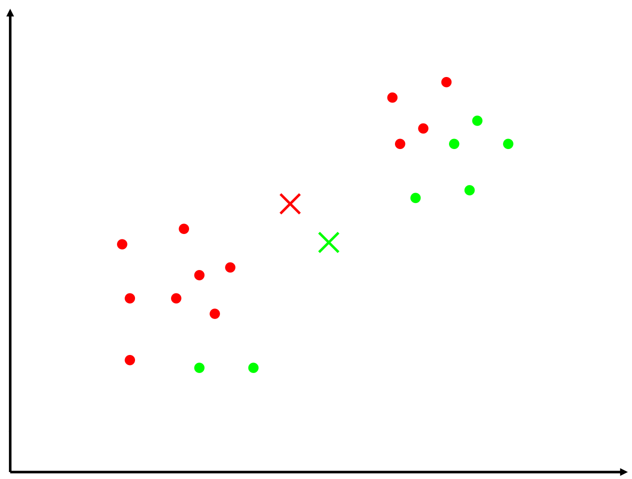
k-means Clustering



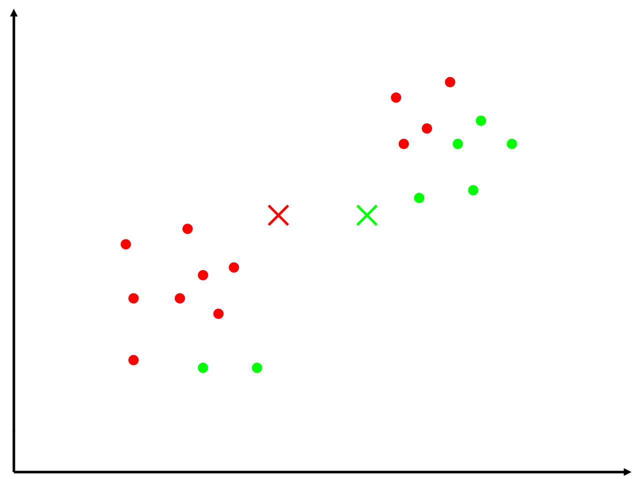
k-means Clustering



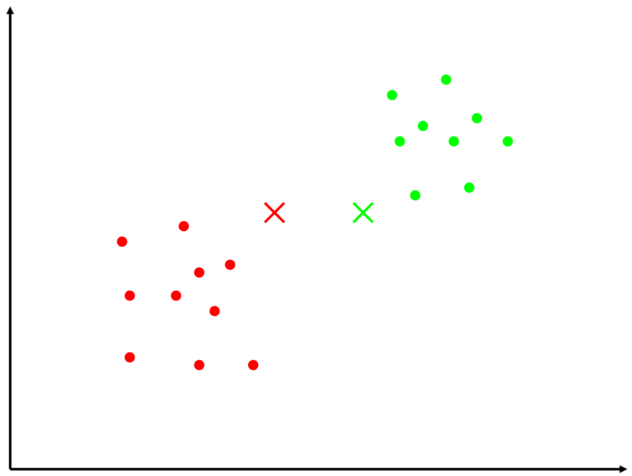
k-means Clustering



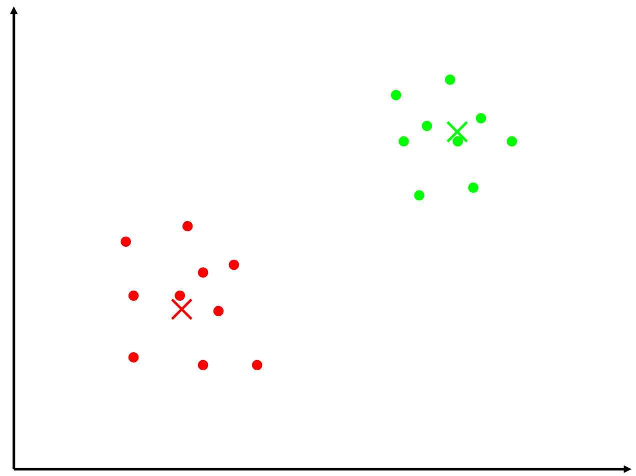
k-means Clustering



k-means Clustering



k-means Clustering



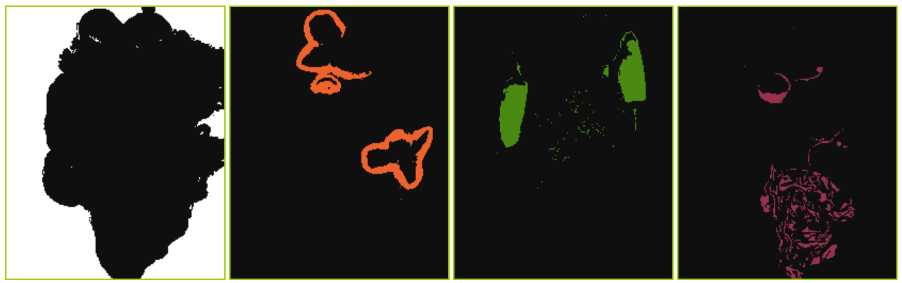
Results of Clustering

Original Image

k-means, k=5

k-means, k=11

Results of Clustering



Sample clusters with *k*-means clustering
based on color

Other Distance Measures

- Suppose we want to have compact regions
- New feature space: 5D
  (2 spatial coordinates, 3 color components)
- Points close in this space are close both in color and in actual proximity

Interactive segmentation

video1 video2

Matting

High-level vision

Recognition

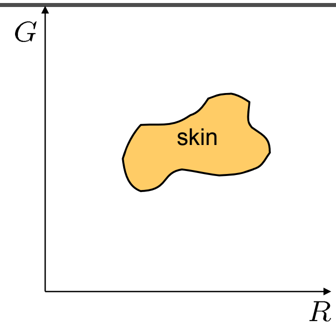
Recognition problems

- What is it?
  - Object detection
- Who is it?
  - Recognizing identity
- What are they doing?
  - Activities
- All of these are **classification** problems
  - Choose one class from a list of possible candidates

Face detection

- How to tell if a face is present?

One simple method: skin detection

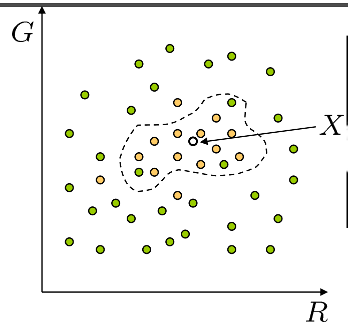


- Skin pixels have a distinctive range of colors
  - Corresponds to region(s) in RGB color space
    - for visualization, only R and G components are shown above

Skin classifier

- A pixel X = (R,G,B) is skin if it is in the skin region
- But how to find this region?

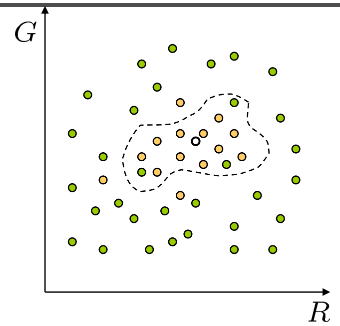
Skin detection



- **Learn** the skin region from examples
  - Manually label pixels in one or more “training images” as skin or not skin
  - Plot the training data in RGB space
    - skin pixels shown in orange, non-skin pixels shown in blue
    - some skin pixels may be outside the region, non-skin pixels inside. Why?

Skin classifier
- Given X = (R,G,B): how to determine if it is skin or not?

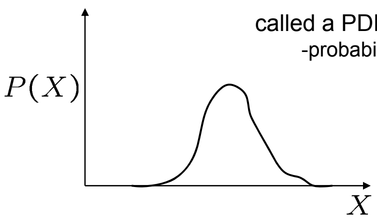
Skin classification techniques



Skin classifier
- Given X = (R,G,B): how to determine if it is skin or not?
- Nearest neighbor
  - find labeled pixel closest to X
  - choose the label for that pixel
- Data modeling
  - fit a model (curve, surface, or volume) to each class
- Probabilistic data modeling
  - fit a probability model to each class

Probability

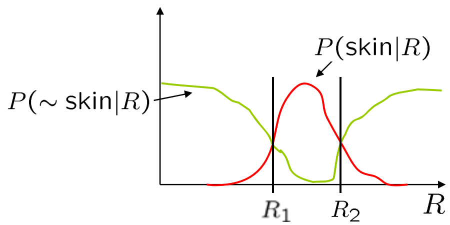
- Basic probability
  - X is a random variable
  - P(X) is the probability that X achieves a certain value



called a PDF
-probability distribution/density function

  - Conditional probability: P(X | Y)
    - probability of X given that we already know Y

Probabilistic skin classification

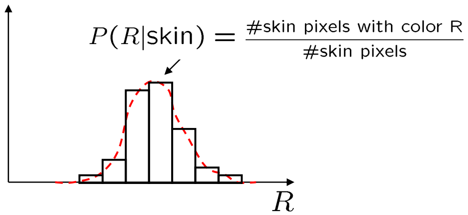


- Now we can model uncertainty
  - Each pixel has a probability of being skin or not skin
    - $P(\sim \text{skin}|R) = 1 - P(\text{skin}|R)$

Skin classifier
- Given X = (R,G,B): how to determine if it is skin or not?
- Choose interpretation of highest probability
  - set X to be a skin pixel if and only if $R_1 < X \leq R_2$

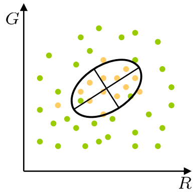
Where do we get $P(\text{skin}|R)$ and $P(\sim \text{skin}|R)$?

Learning conditional PDF's



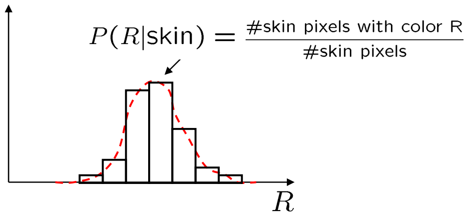
- We can calculate P(R | skin) from a set of training images
  - It is simply a histogram over the pixels in the training images
    - each bin R_i contains the proportion of skin pixels with color R_i

This doesn't work as well in higher-dimensional spaces. Why not?



Approach: fit parametric PDF functions
- common choice is rotated Gaussian
  - center $\mathbf{c} = \overline{X}$
  - covariance $\sum_X (X - \overline{X})(X - \overline{X})^T$
    - » orientation, size defined by eigenvecs, eigenvals

Learning conditional PDF's



- We can calculate P(R | skin) from a set of training images
  - It is simply a histogram over the pixels in the training images
    - each bin R_i contains the proportion of skin pixels with color R_i

But this isn't quite what we want
- Why not? How to determine if a pixel is skin?
- We want P(skin | R) not P(R | skin)
- How can we get it?

Bayes rule

$$P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)}$$

- In terms of our problem:

what we measure (**likelihood**) — domain knowledge (**prior**)

$$P(\text{skin}|R) = \frac{P(R|\text{skin})\ P(\text{skin})}{P(R)}$$

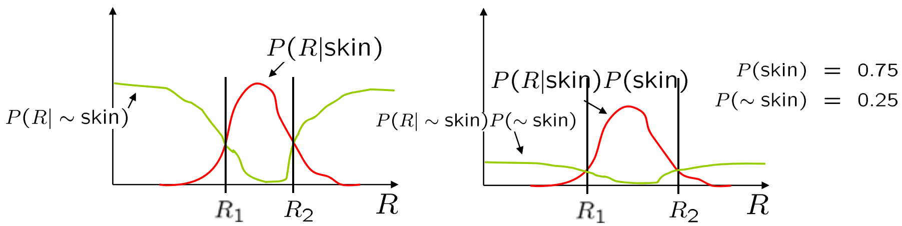
what we want (**posterior**)

normalization term

$$P(R) = P(R|\text{skin})P(\text{skin}) + P(R|\sim\text{skin})P(\sim\text{skin})$$

The prior: P(skin)
- Could use domain knowledge
  - P(skin) may be larger if we know the image contains a person
  - for a portrait, P(skin) may be higher for pixels in the center
- Could learn the prior from the training set. How?
  - P(skin) may be proportion of skin pixels in training set

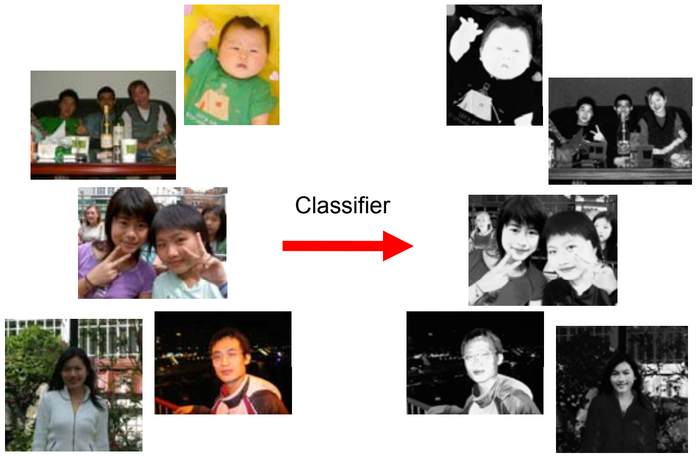
Bayesian estimation



likelihood — posterior (unnormalized)

- Bayesian estimation = minimize probability of misclassification
  - Goal is to choose the label (skin or ~skin) that maximizes the posterior
    - this is called **Maximum A Posteriori (MAP) estimation**

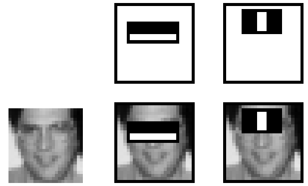
Skin detection results



Viola/Jones: features

"Rectangle filters"

Differences between sums of pixels in adjacent rectangles



$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

$$60{,}000 \times 100 = 6{,}000{,}000$$

Unique Features

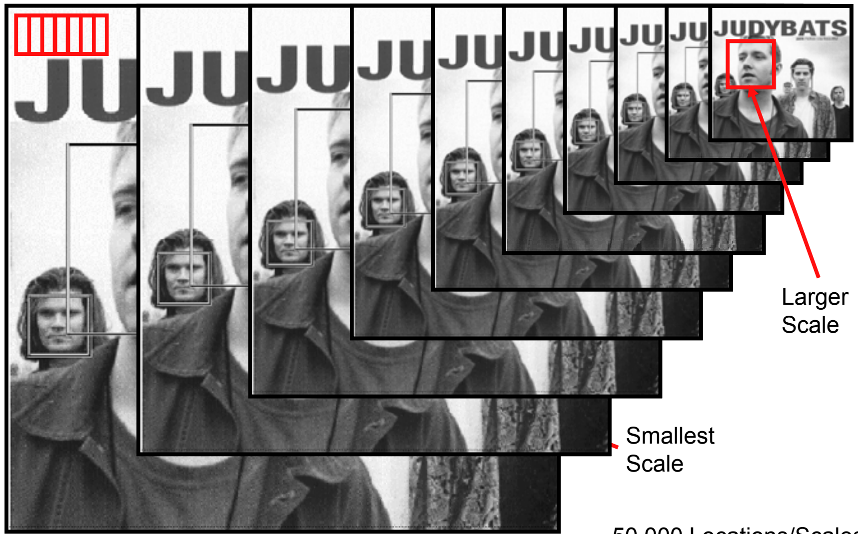
$Y(x)=\sum \alpha_t h_t(x)$

Select 200 by Adaboost

$$\text{Detection} = \begin{cases} \text{face,} & \text{if } Y(x) > 0 \\ \text{non-face,} & \text{otherwise} \end{cases}$$

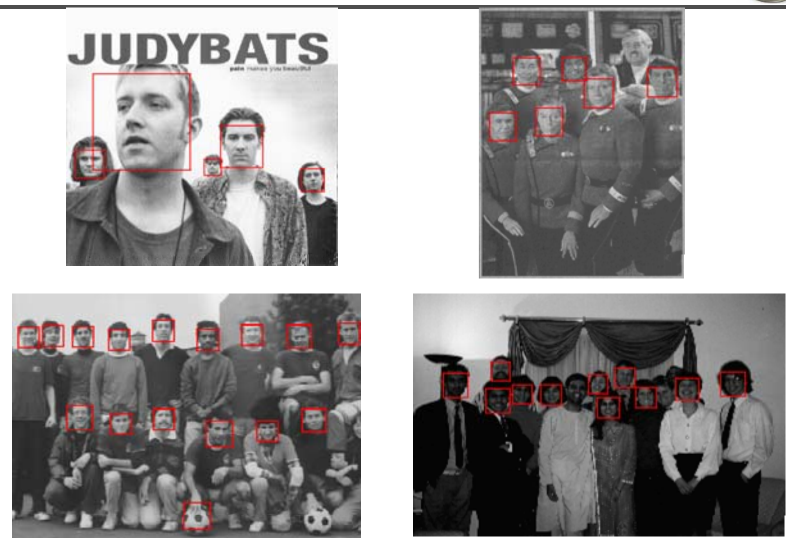
Robust Realtime Face Detection, IJCV 2004, Viola and Jones

Viola/Jones: handling scale



50,000 Locations/Scales

Viola/Jones results:



Run-time: 15fps (384x288 pixel image on a 700 Mhz Pentium III)

Application

Smart cameras: auto focus, red eye removal, auto color correction

Application

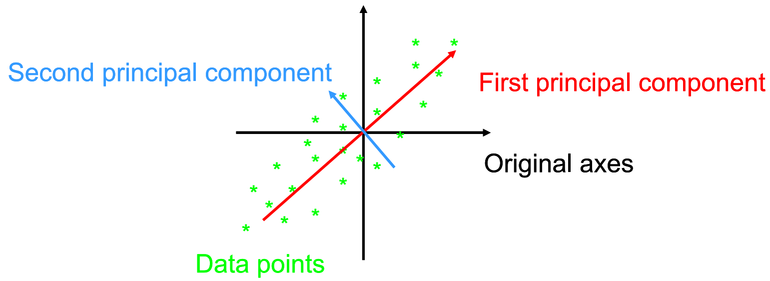
Lexus LS600 Driver Monitor System

Face recognition

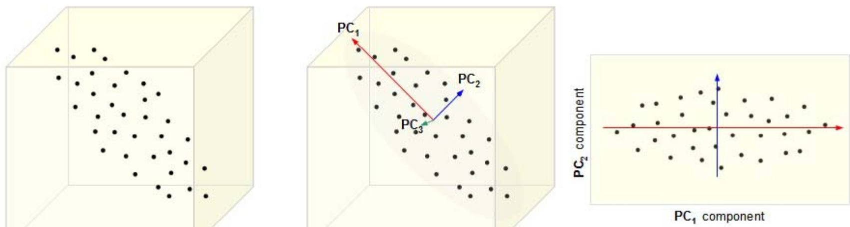
- Suppose you want to recognize a *particular* face
- How does *this* face differ from average face
- Consider variation from average face
- Not all variations equally important
  - Variation in a single pixel relatively unimportant
- If image is high-dimensional vector, want to find directions in this space with high variation

PCA

- Principal Components Analysis (PCA): approximating a high-dimensional data set with a lower-dimensional subspace



PCA

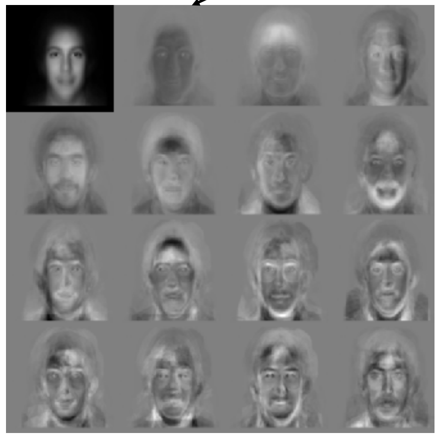


a b c

PCA on Faces: "Eigenfaces"

Average face

First principal component



Other components

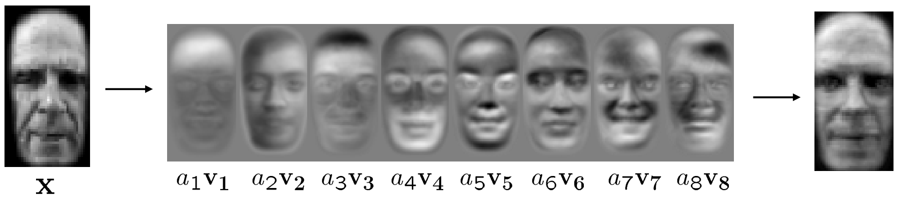
For all except average,
"gray" = 0,
"white" > 0,
"black" < 0

Using PCA for Recognition

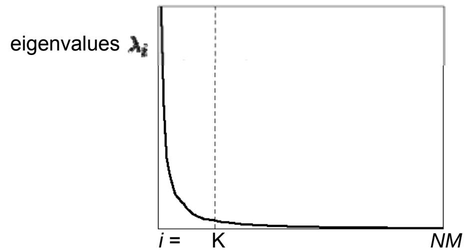
- Store each person as coefficients of projection onto first few principal components

$$\text{image} = \sum_{i=0}^{i_{\max}} a_i \text{Eigenface}_i$$

- Compute projections of target image, compare to database ("nearest neighbor classifier")



Choosing the dimension K



- How many eigenfaces to use?
- Look at the decay of the eigenvalues
  - the eigenvalue tells you the amount of variance "in the direction" of that eigenface
  - ignore eigenfaces with low variance

Advanced topics

High dynamic range imaging/display

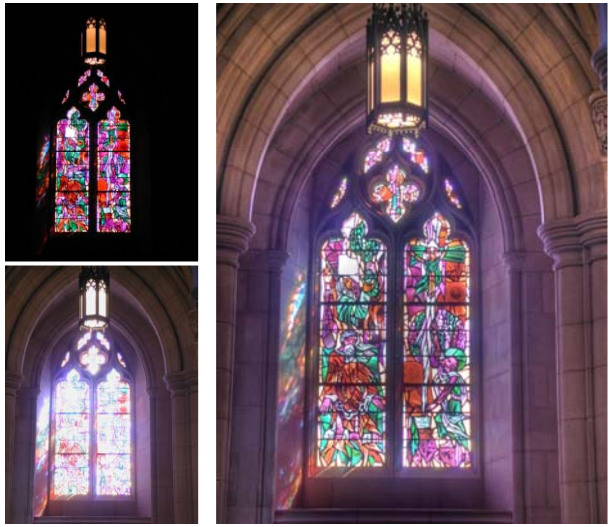


Image warping/morphing

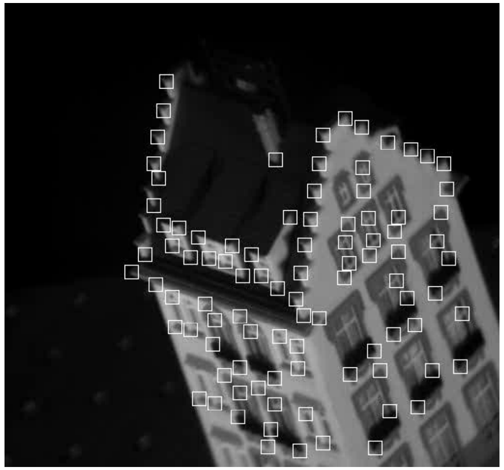
someone not that famous

someone very famous

video

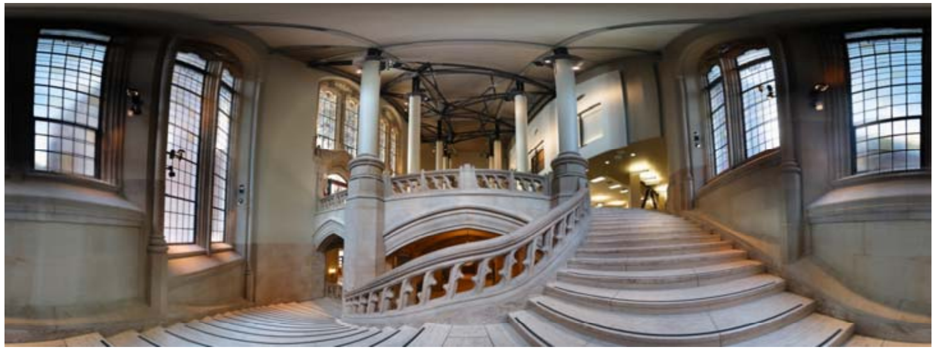
Image warping/morphing

Tracking



Feature tracking

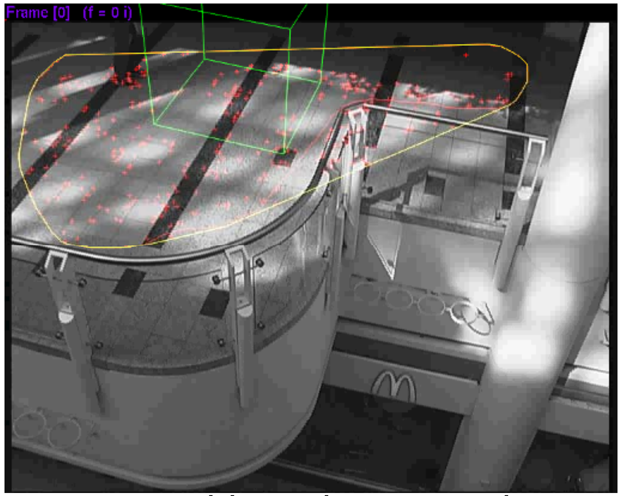
Image stitching



MatchMove

Move matching using scene planes

Matchmove



Move matching using scene planes

Matchmove

Move matching using scene planes

Photo tourism

Video matching

Matrix *MOCO (Motion control camera)*

Video matching

Video matching

Matting and compositing

Titanic

Matting

Image manipulation

GraphCut Texture

Image manipulation

Poisson blending

Image-based modeling

photogrammetric modeling and projective texture-mapping

Image-based modeling

photogrammetric modeling and projective texture-mapping

Image-based modeling

photogrammetric modeling and projective texture-mapping

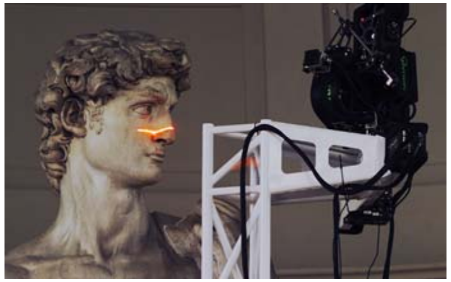
Image-based modeling

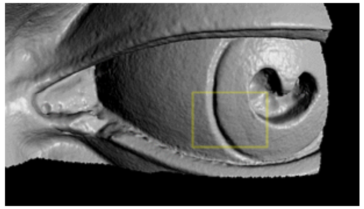
Tour into a picture

Image-based modeling

Tour into a picture

Structured Light and Ranging Scanning



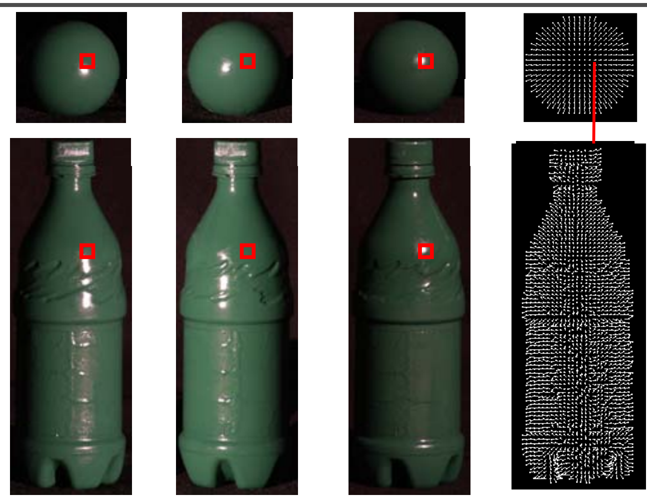


http://graphics.stanford.edu/projects/mich/

3D photography (active)

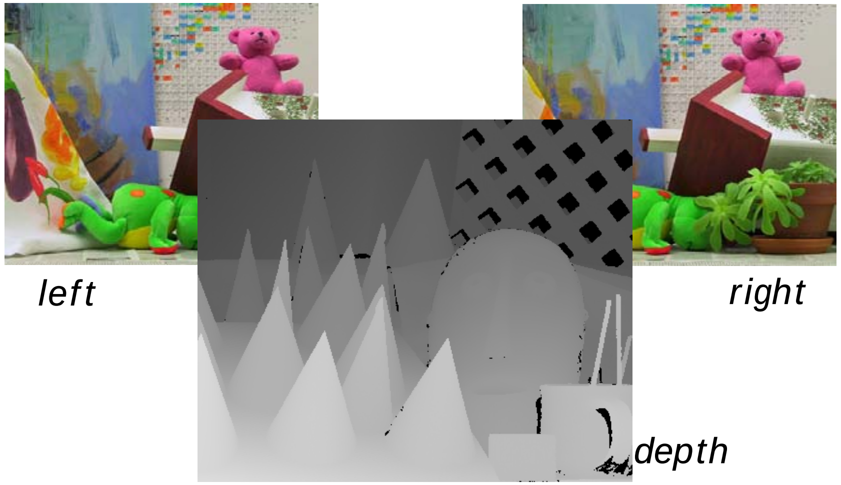
Cyberware whole body scanner

3D photography (active)



Photometric stereo

3D photography (passive)



Stereo

Image-based rendering

Surface lightfield

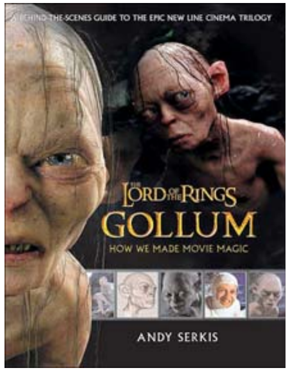
View interpolation

Bullet time video

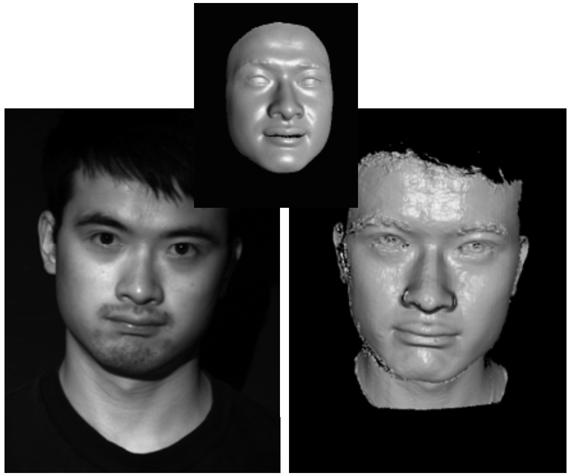
View interpolation

High-Quality Video View Interpolation

Making face



Gollum



Spacetime face

Video rewrite

Trainable videorealistic speech animation

Inpainting (wire removal)

Inpainting

Texture synthesis/replacement

Texture replacement

Semi-automatic matting painting

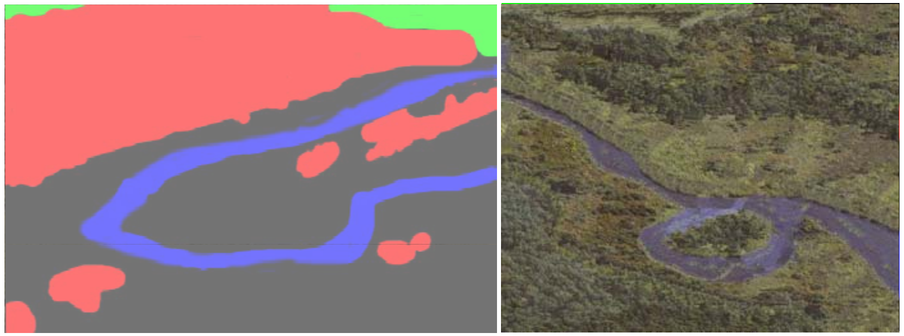
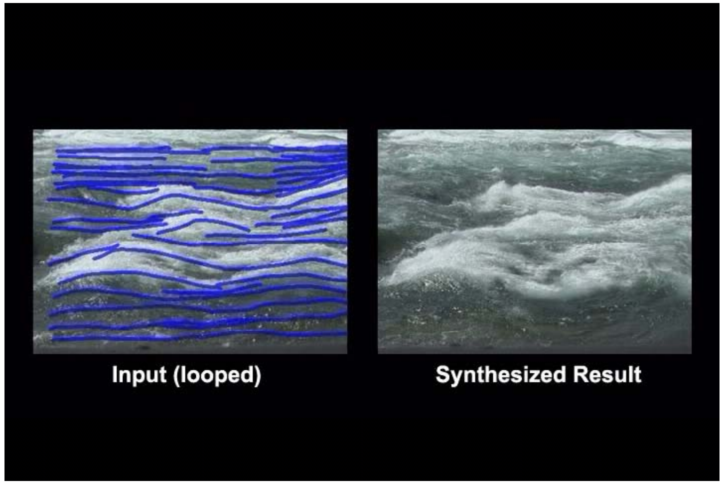


Image analogies

Video editing



Flow-based video editing

Face Detection and Recognition

Motion Estimation

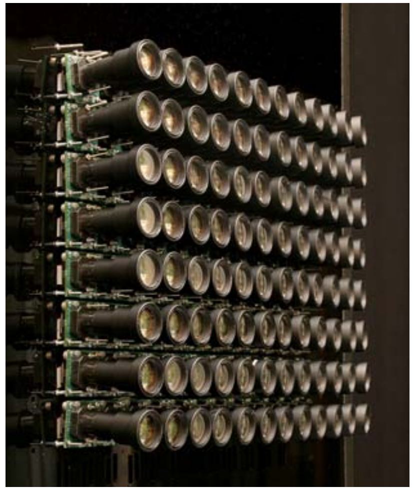
Application

Andy Serkis, Gollum, Lord of the Rings

Novel Cameras and Displays

http://www1.cs.columbia.edu/CAVE/projects/cc.htm

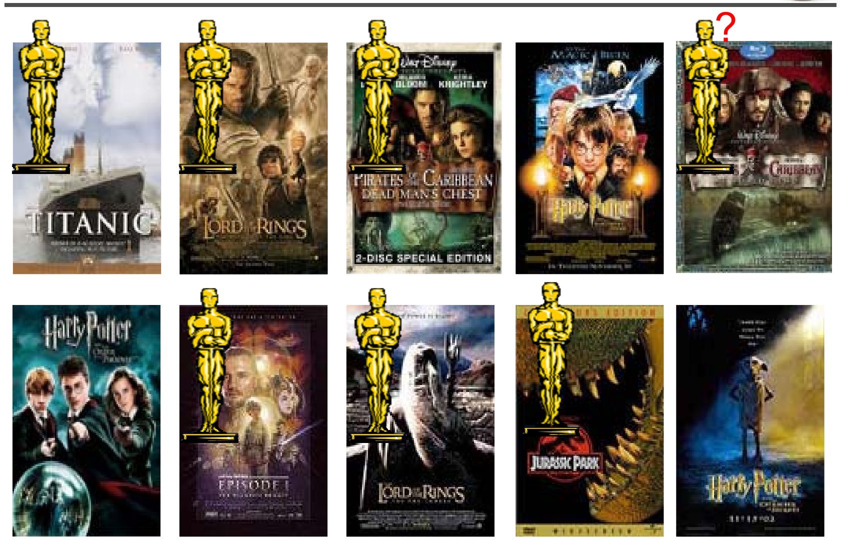
Capturing Light Field



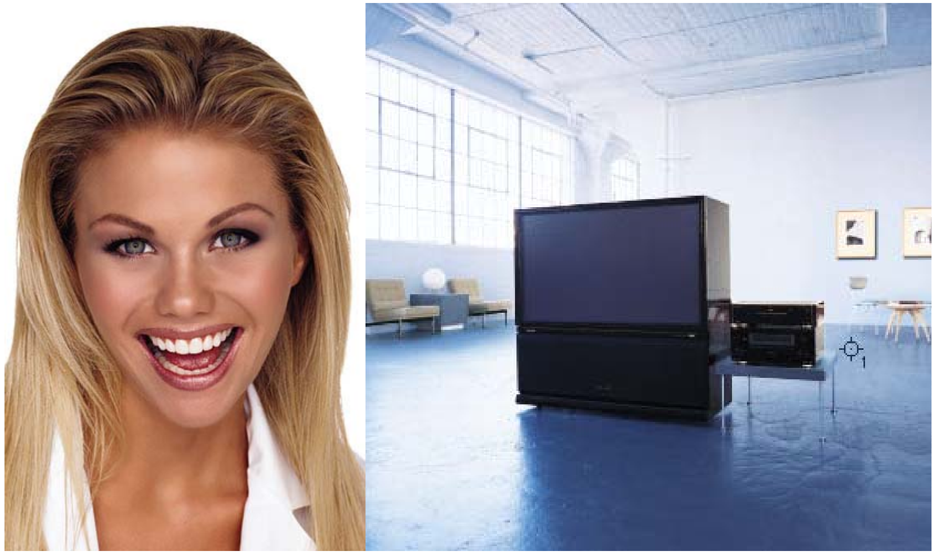
Camera Arrays, Graphics Lab, Stanford University

Digital visual effects

Digital Visual Effects



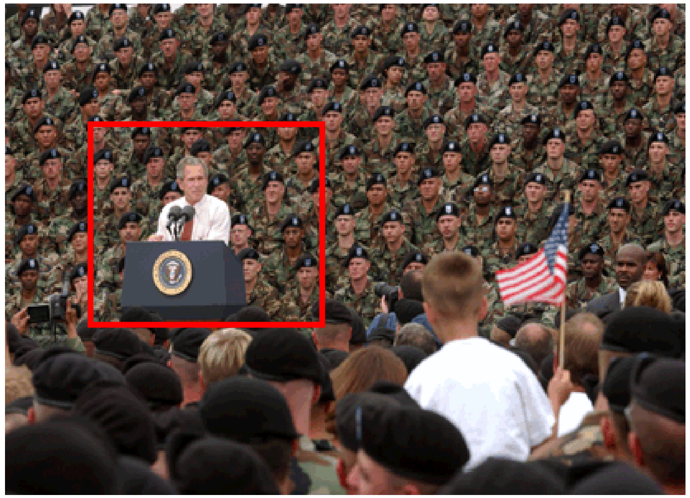
Reality?



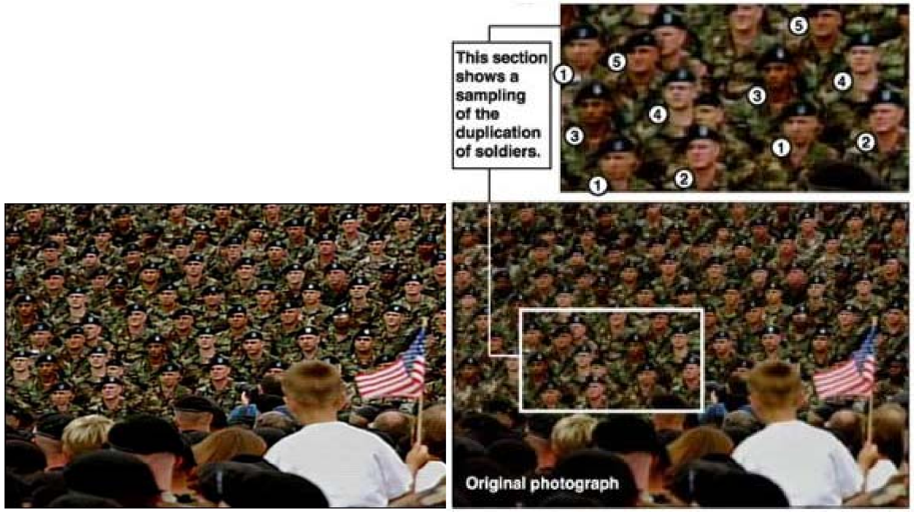
Retouching

Iraq War, LA Times, April 2003

Bush campaign's TV AD, 2004

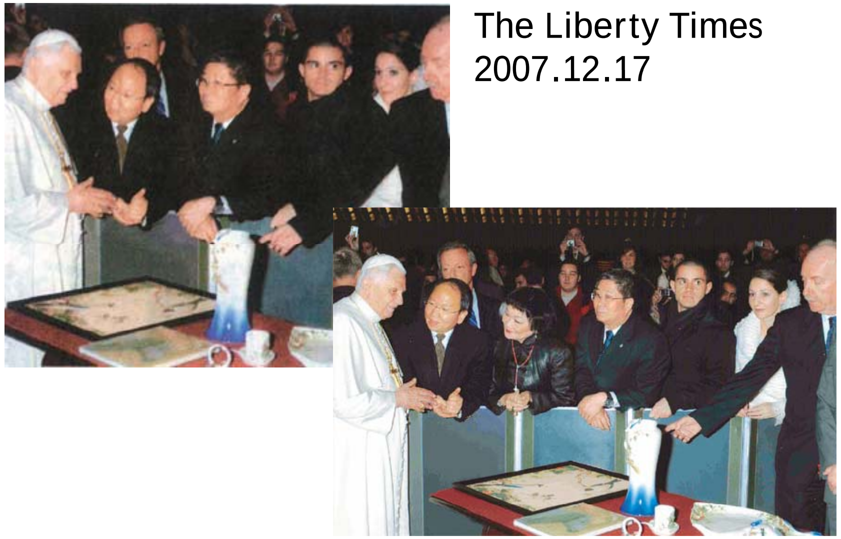


Texture synthesis and inpainting



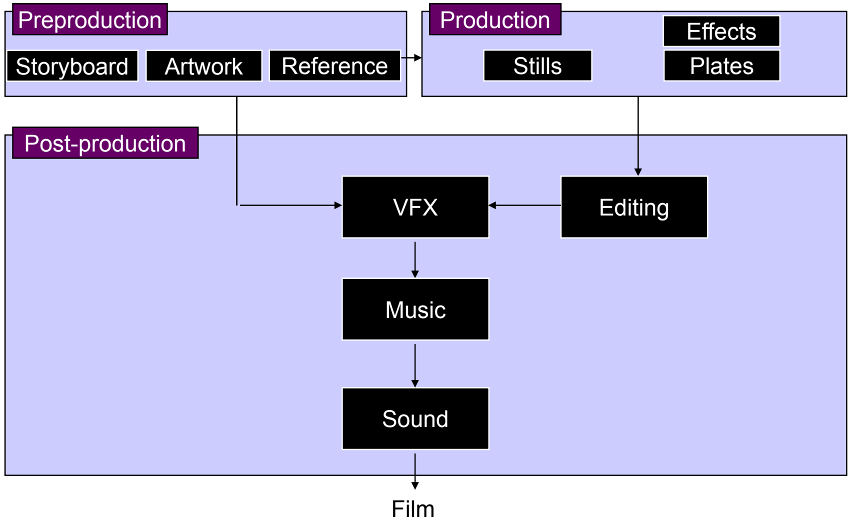
Domestic example

The Liberty Times
2007.12.17



Production pipeline

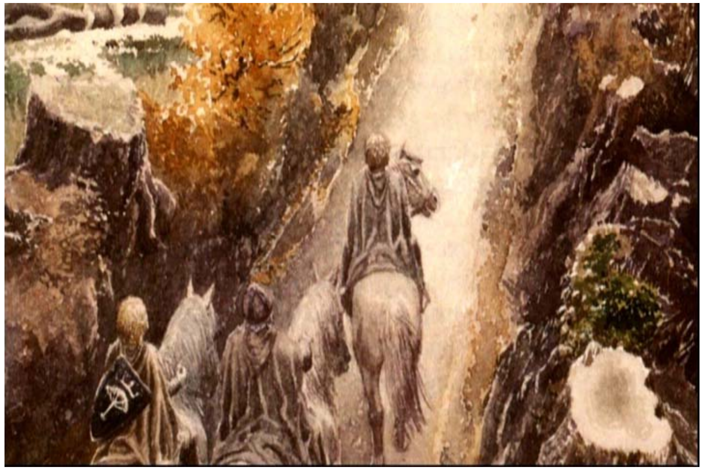
Production pipeline



Preproduction

Storyboard

Preproduction



Artwork

Preproduction

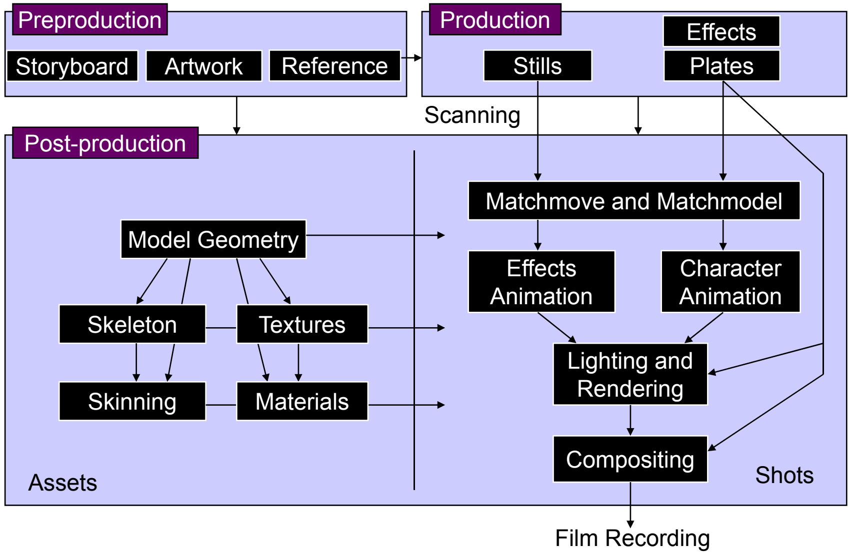
Reference & Research

Production

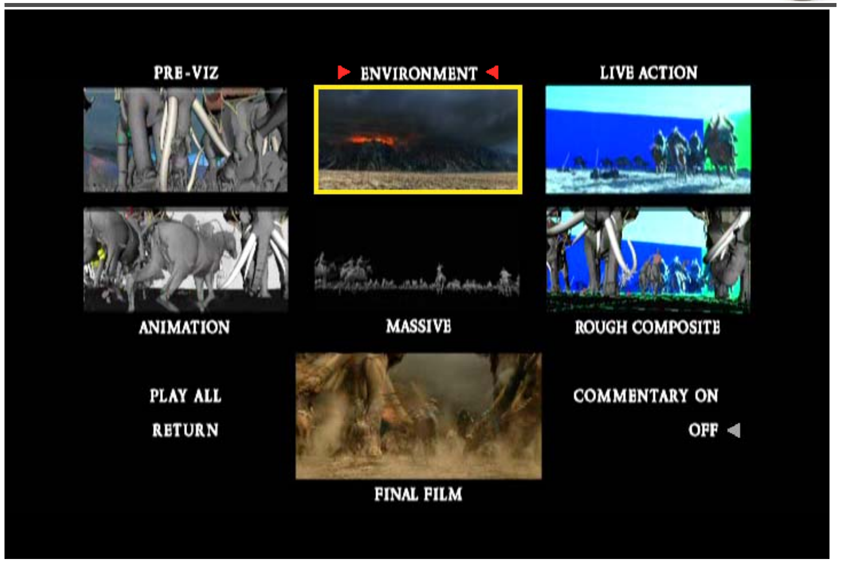
Shooting

Post-production

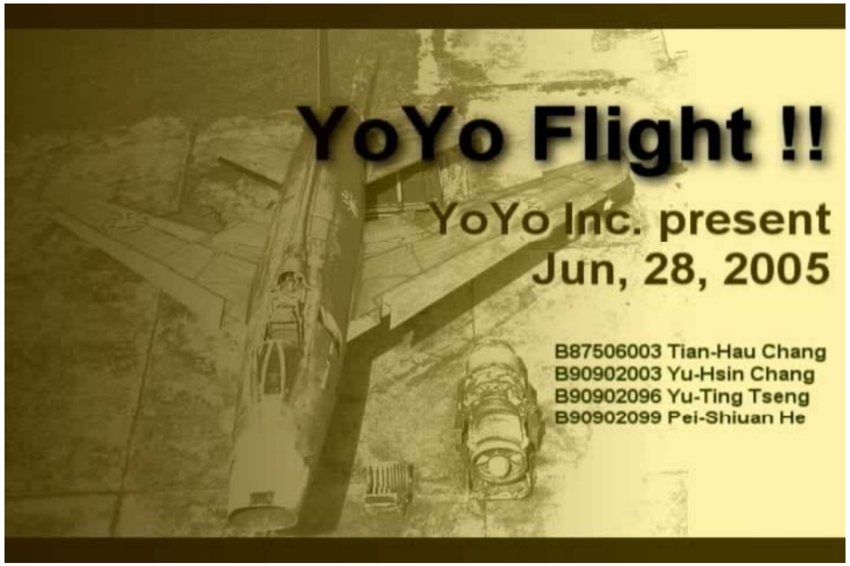
Visual effects production



Visual effects post-production

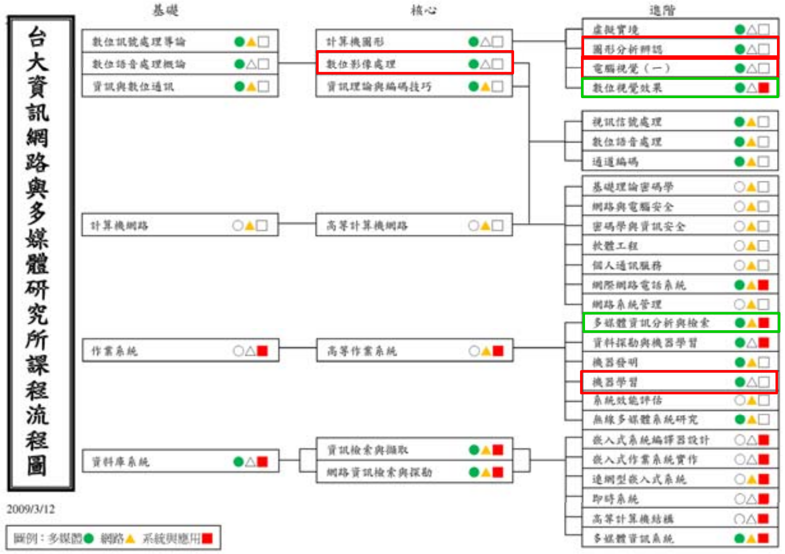


YoYo Flight



What's next?

Related courses



Related courses

專題

多媒體	影像式塑模與描繪 ●△□	數位影像生成 ●△□	視訊壓縮技術、標準與實務 ●▲■
	遊戲設計 ●△■	幾何模型 ●△□	電腦視覺（二） ●△□
	機器學習和電腦視覺專題 ●△□	高等多媒體資訊分析與檢索 ●▲□	視訊壓縮技術與研發專題 ●△□
	資訊與歷史資料分析 ●△■	醫學影像處理 ●△□	機器學習於多媒體之應用 ●△■
	數位圖書館與博物館研討 ●△■	多媒體安全 ●△■	統計人工智慧 ●▲■
	高等人機互動介面 ●▲■		
網路	網路資訊安全（二） ○▲□	行動通訊服務系統效能評估 ○▲□	網路與多媒體之技術與產業 ●▲□
	高等資訊網技術 ○▲□	網路模擬與測試 ○▲□	軟體產業之經營策略 ●▲■
	賽局理論 ○▲■	智慧型代理程式設計 ○▲■	電子商務系統 ●▲■
	網路藝術 ●▲■	無線感測網路與實作 ○▲□	
系統與應用	分散式代理人系統 ○▲■	嵌入式處理器設計 ○△■	生物資訊演算法 ○△■
	計算生物基礎數學 ○△■	生醫資料探勘演算法 ○△■	平行計算多處理機系統晶片 ○▲■
	計算機效能最佳化 ○△■	低功率系統設計 ○△■	輸出裝置與驅動程式設計 ○△■
	嵌入式系統概論 ○△■	低功率嵌入式系統設計 ○△■	嵌入式多核心系統與軟體 ○△■
	生物資訊之統計與計算方法 ○△■	軟硬體共同設計 ○△■	電腦對局理論 ○△■
	國外專業實習 ○△■	科技英文寫作與研究方法 ○△■	模糊系統與應用 ○△■
	人工智慧 ●▲■	機器人知覺與學習 ●△■	高等行動機器人學 ●▲■
	高等人工智慧 ●▲■	多媒體晶片系統設計 ●△■	即時系統開發 ○▲■
	普及計算 ●▲■	軟體專利實務基礎 ●▲■	多核處理器及編譯 ○△■
	嵌入式中介軟體設計 ○△■		

2009/3/12

圖例：多媒體● 網路▲ 系統與應用■