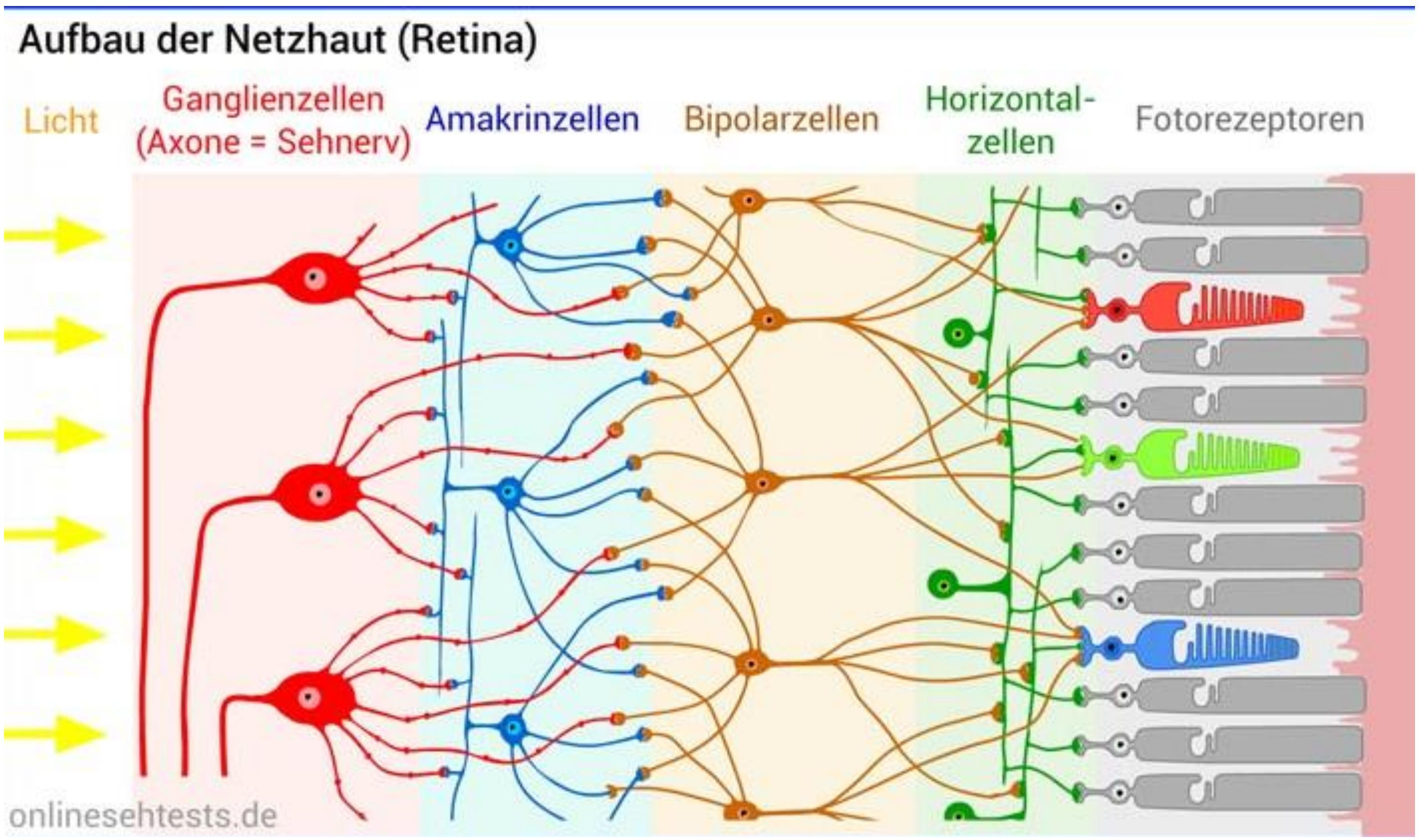
Computer Vision

Robotics

Image Processing

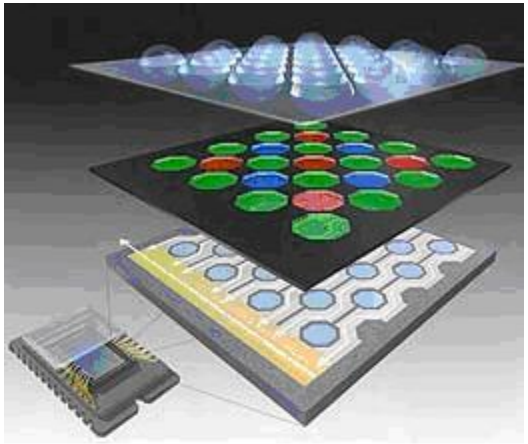
- Bayer filter
- RGB color space
- YUV
- HSI, HSV
- Convolution operators
- Binarization: Threshold, OTSU
- Erosion, dilation
- Line detection: RANSAC

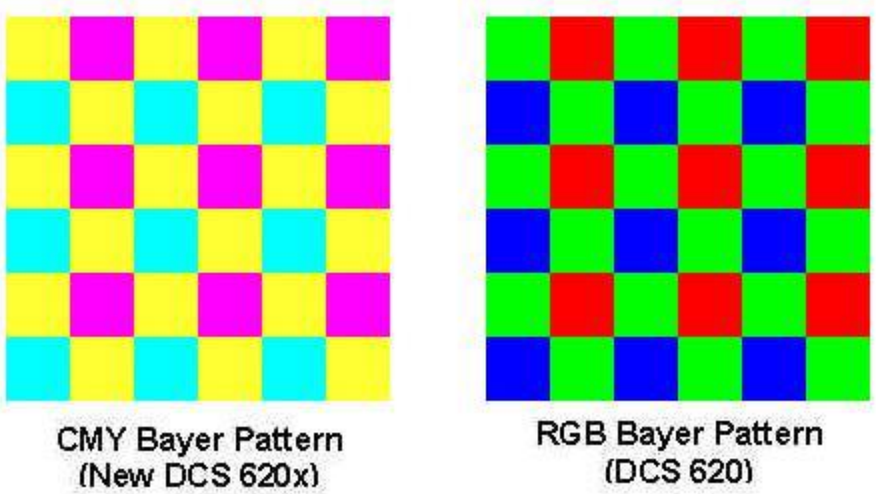
Retina

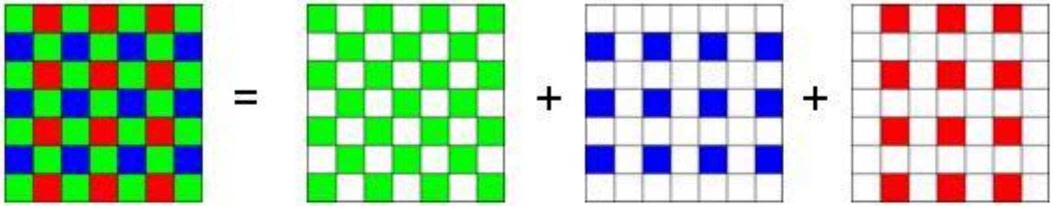


https://youtu.be/GOlGsDS1ZBk

Bayer Pattern







Color constancy

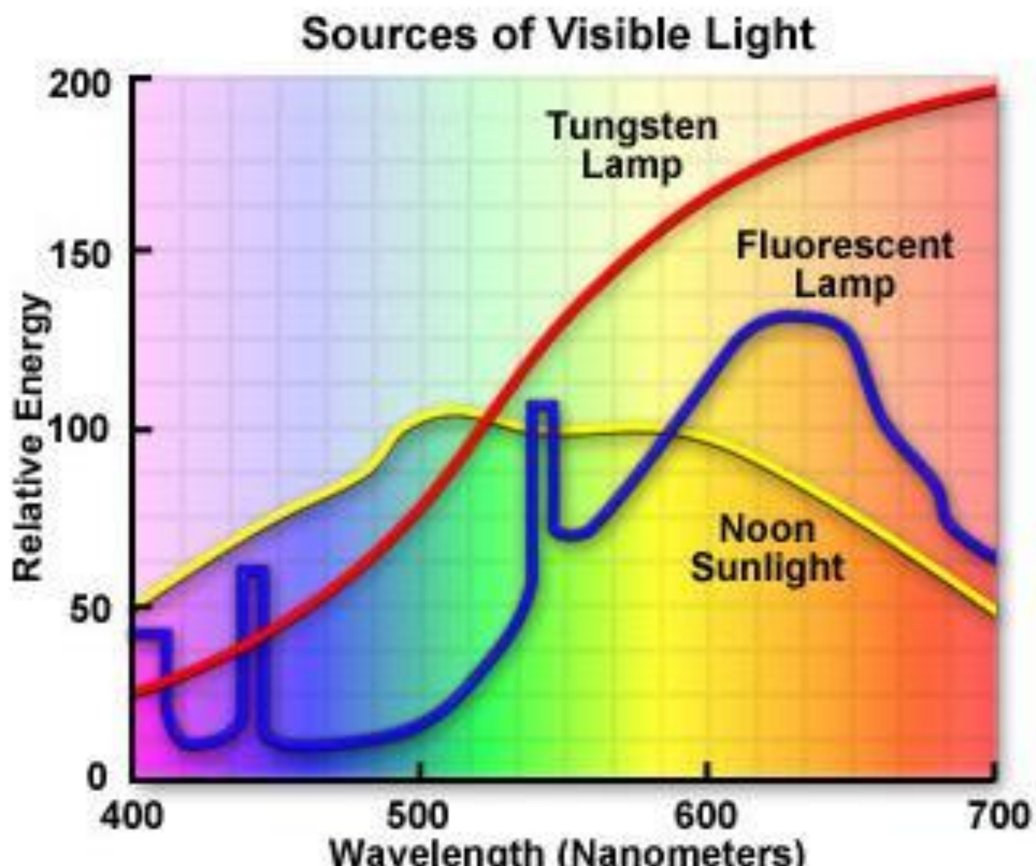
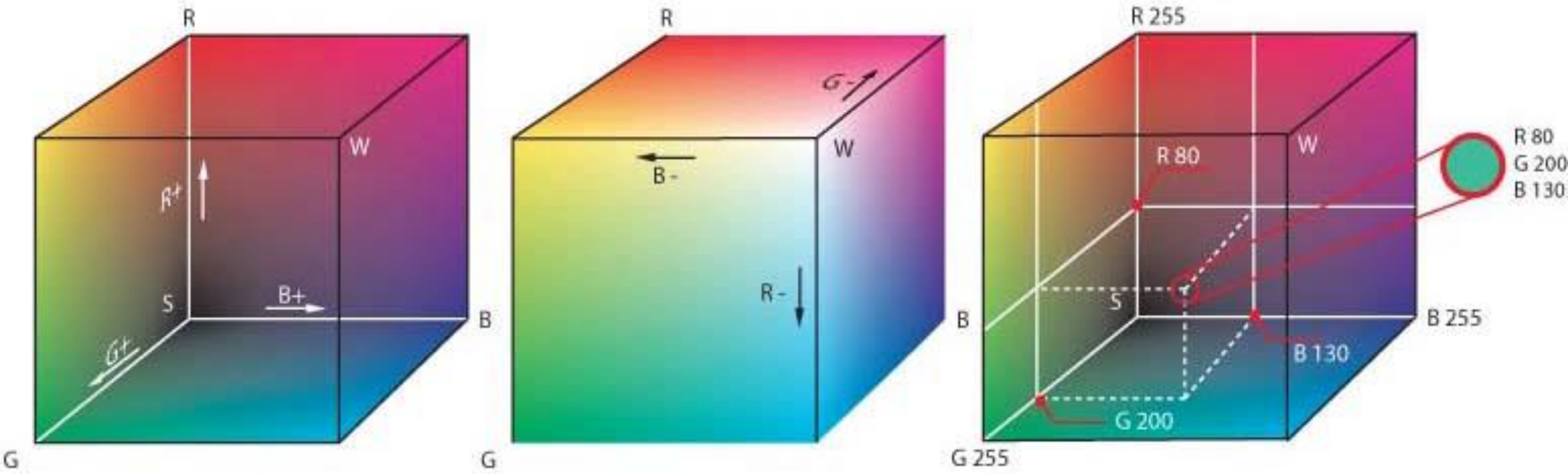
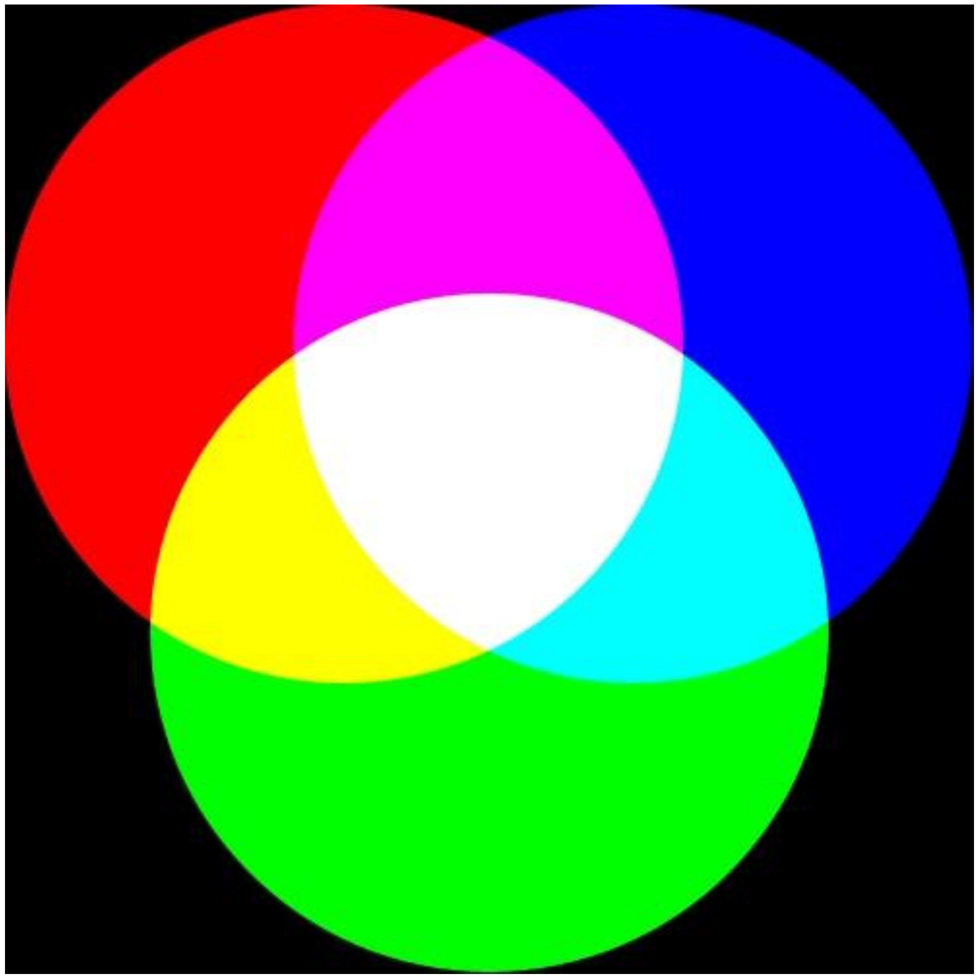


Figure 2

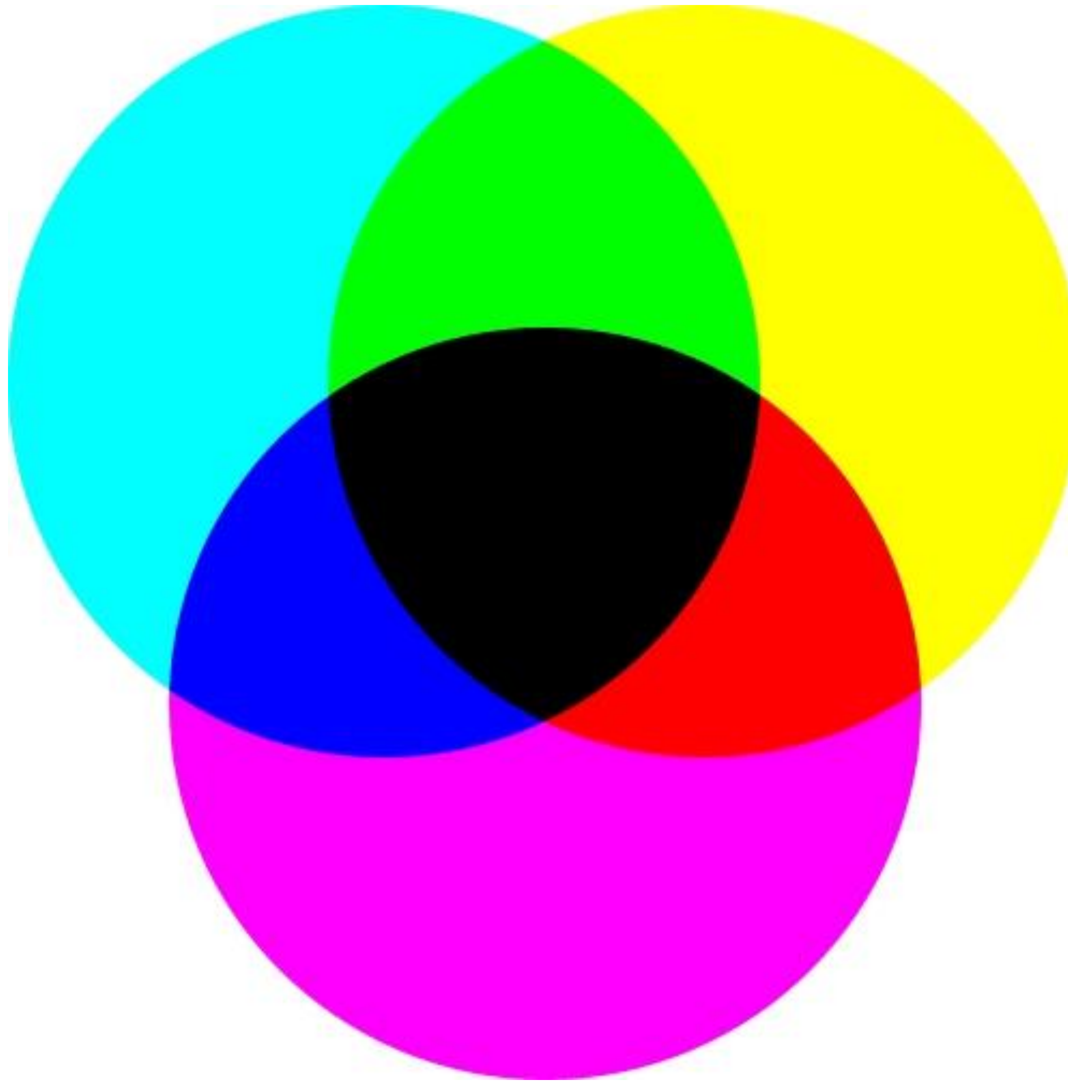
Farbräume: RGB



Additive color mixing



Subtractive color mixing



YUV color model

$$\boxed{Y := 0{,}299 \cdot R + 0{,}587 \cdot G + 0{,}114 \cdot B}$$

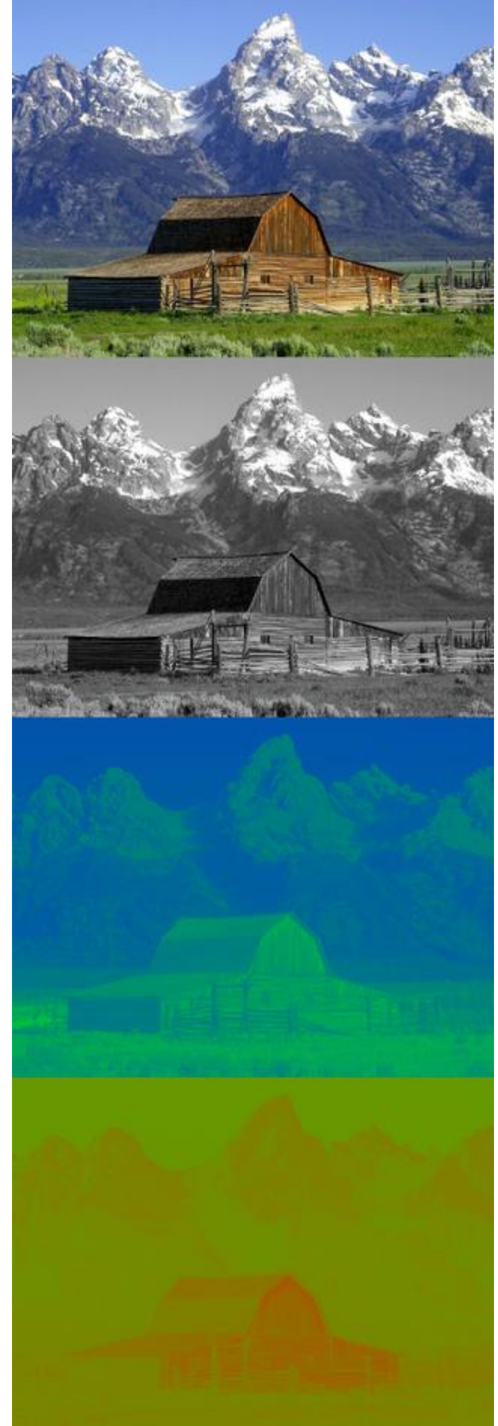
$$\boxed{\begin{aligned} B &= Y + U/0{,}493 \\ R &= Y + V/0{,}877 \\ G &= \frac{1}{0{,}587} \cdot Y - \frac{0{,}299}{0{,}587} \cdot R - \frac{0{,}114}{0{,}587} \cdot B \\ &\approx 1{,}704 \cdot Y - 0{,}509 \cdot R - 0{,}194 \cdot B \end{aligned}}$$

Linear transformations

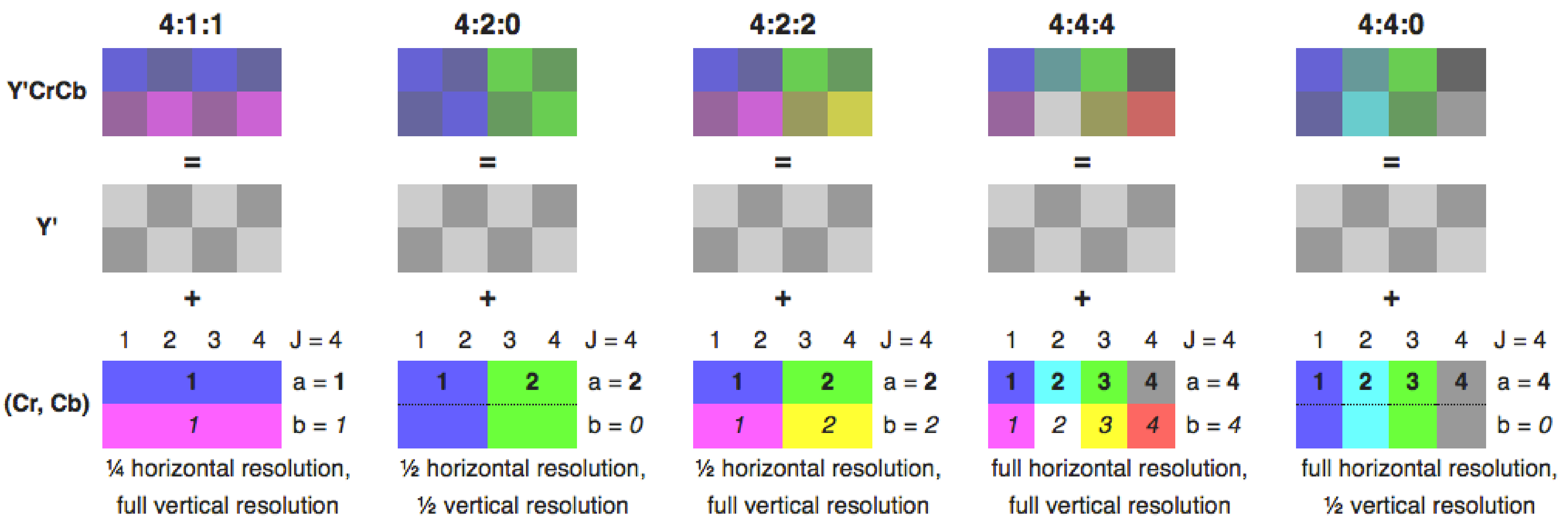
$$\begin{bmatrix} Y' \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix},$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.13983 \\ 1 & -0.39465 & -0.58060 \\ 1 & 2.03211 & 0 \end{bmatrix} \begin{bmatrix} Y' \\ U \\ V \end{bmatrix}.$$

YUV example



Chroma subsampling



HSV

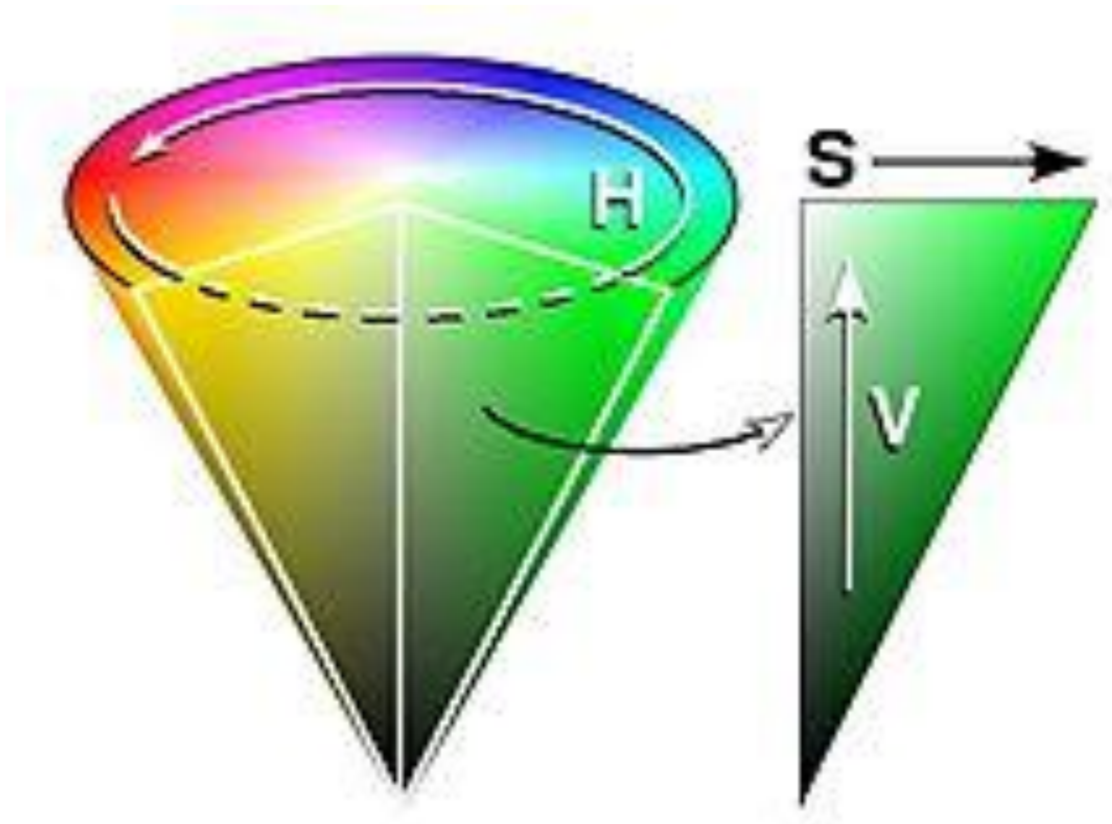
$$MAX := \max(R, G, B),\ MIN := \min(R, G, B)$$

$$H := \begin{cases} 0, & \text{falls } MAX = MIN \Leftrightarrow R = G = B \\ 60^\circ \cdot \left(0 + \frac{G-B}{MAX-MIN}\right), & \text{falls } MAX = R \\ 60^\circ \cdot \left(2 + \frac{B-R}{MAX-MIN}\right), & \text{falls } MAX = G \\ 60^\circ \cdot \left(4 + \frac{R-G}{MAX-MIN}\right), & \text{falls } MAX = B \end{cases}$$

$$S_{\text{HSV}} := \begin{cases} 0, & \text{falls } MAX = 0 \Leftrightarrow R = G = B = 0 \\ \frac{MAX-MIN}{MAX}, & \text{sonst} \end{cases}$$

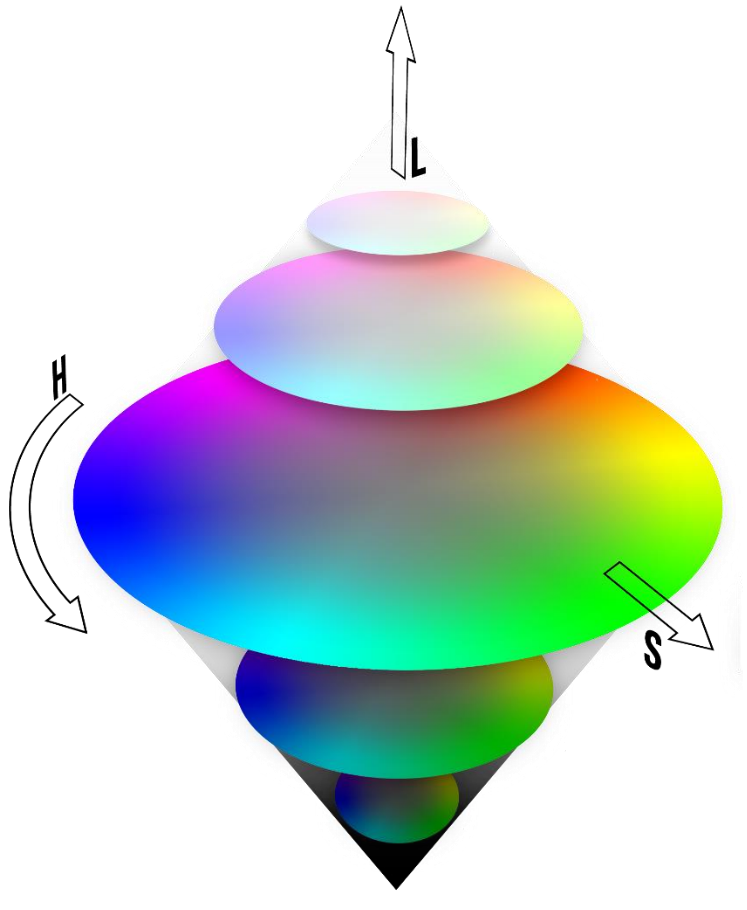
$$V := MAX$$

HSV

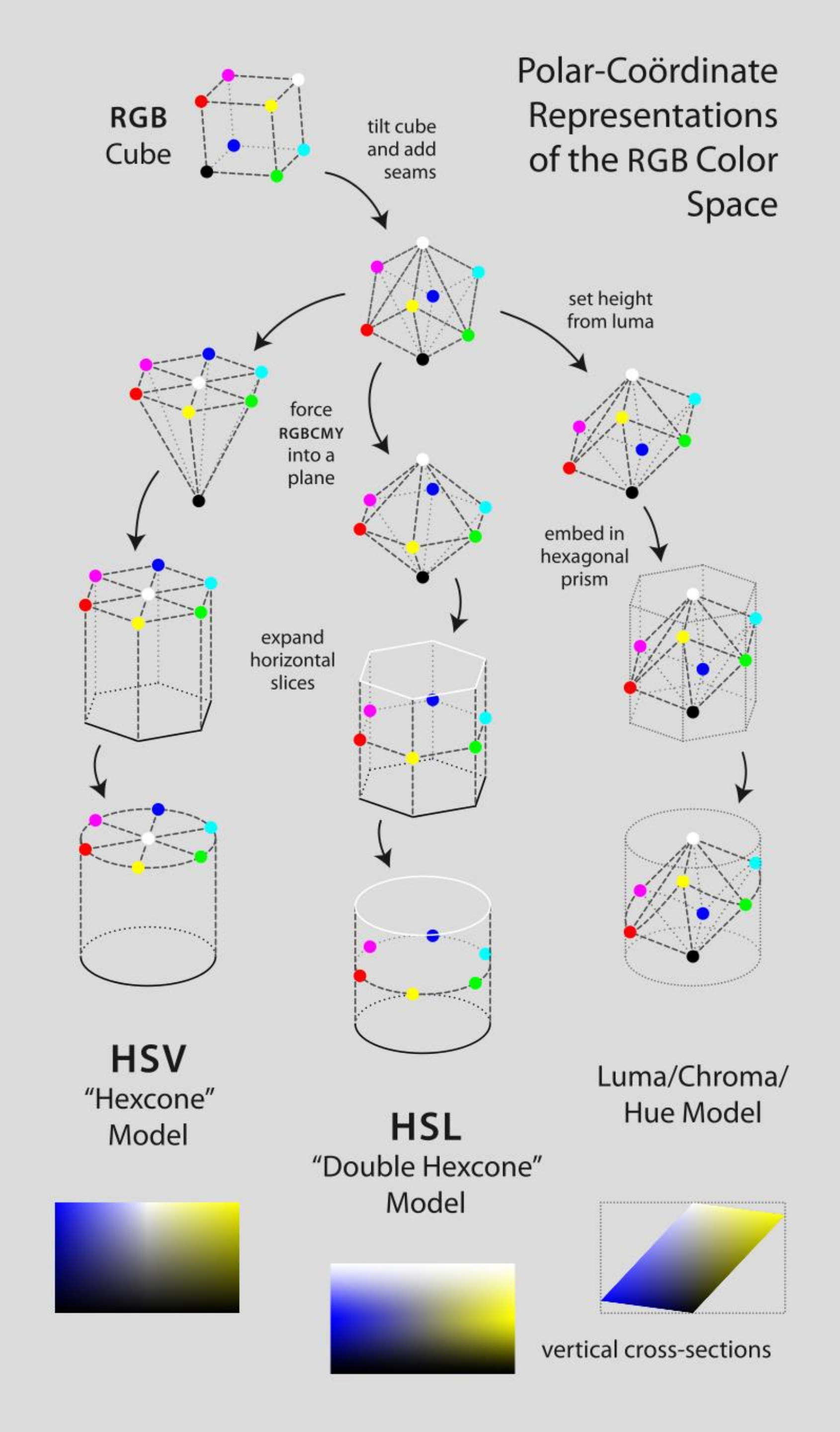


Source: Wikipedia

HSL



Source: Wikipedia



Polar-Coördinate Representations of the RGB Color Space
RGB Cube
tilt cube and add seams
set height from luma
force RGBCMY into a plane
embed in hexagonal prism
expand horizontal slices
HSV "Hexcone" Model
HSL "Double Hexcone" Model
Luma/Chroma/ Hue Model
vertical cross-sections

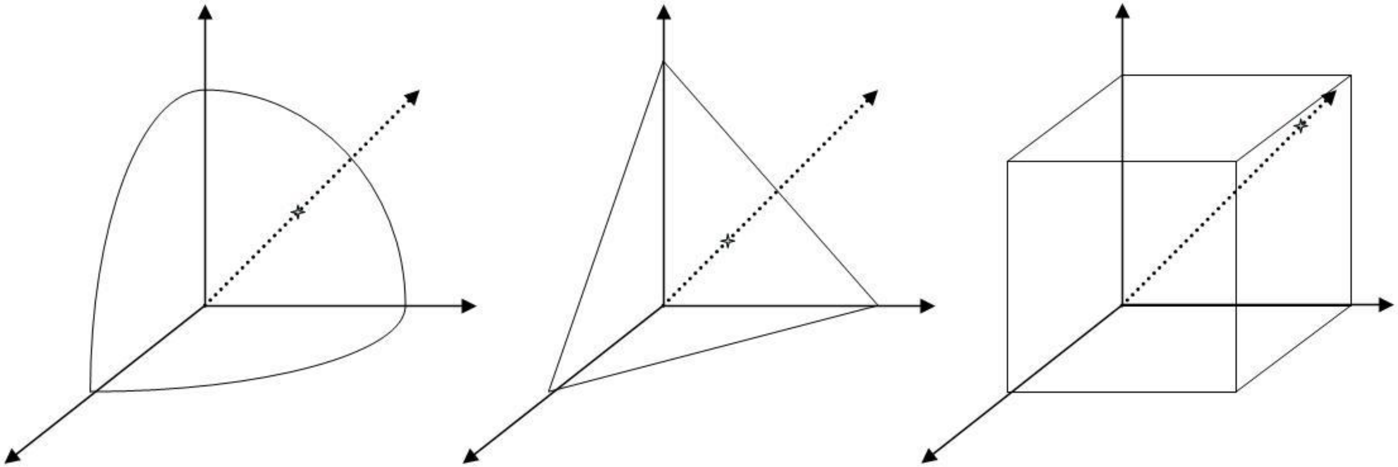
Intensity

$$I_1 = \sqrt{R^2 + G^2 + B^2)}$$

$$I_2 = (R + G + B)$$

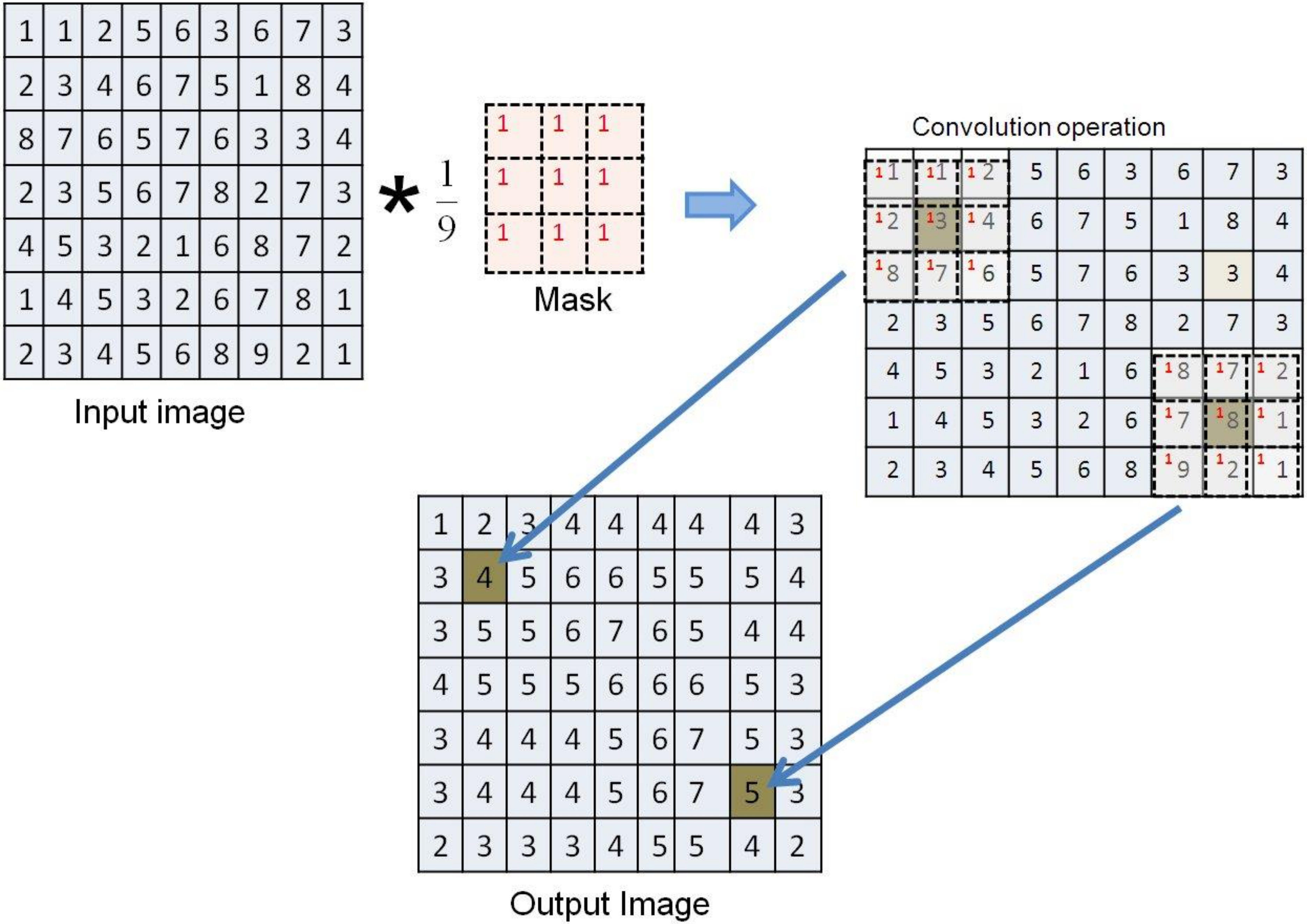
$$I_3 = \max(R, G, B)$$

Isoluminosity



Convolution operators

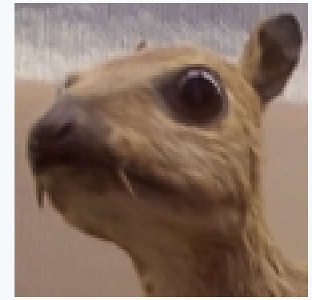
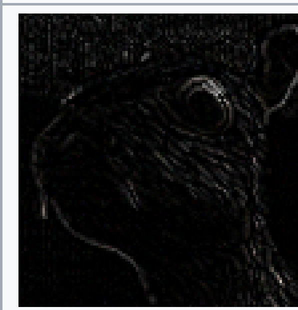
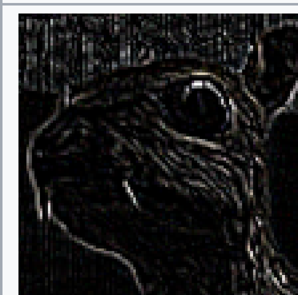
- Box blurring
- Gaussian blurring
- Edge detection



Input image
1/9
Mask
Convolution operation
Output Image

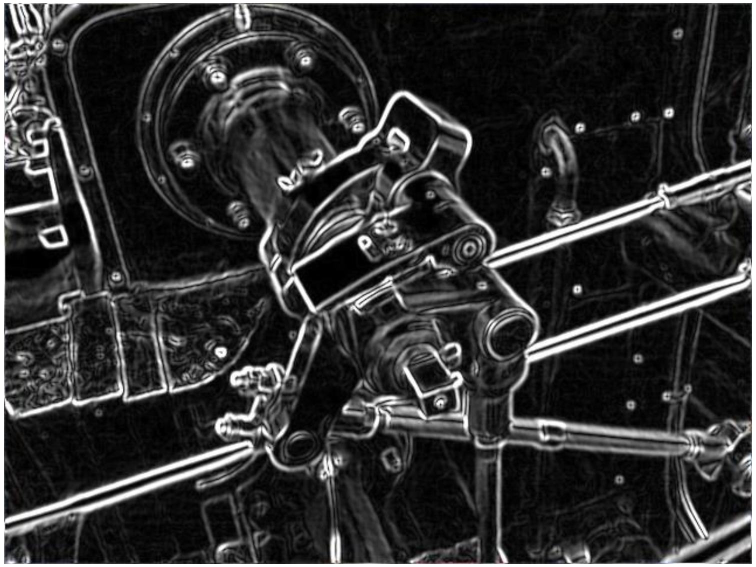
Blurring

Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur 3 × 3 (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Operation	Kernel	Image result
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	

Sobel operator

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

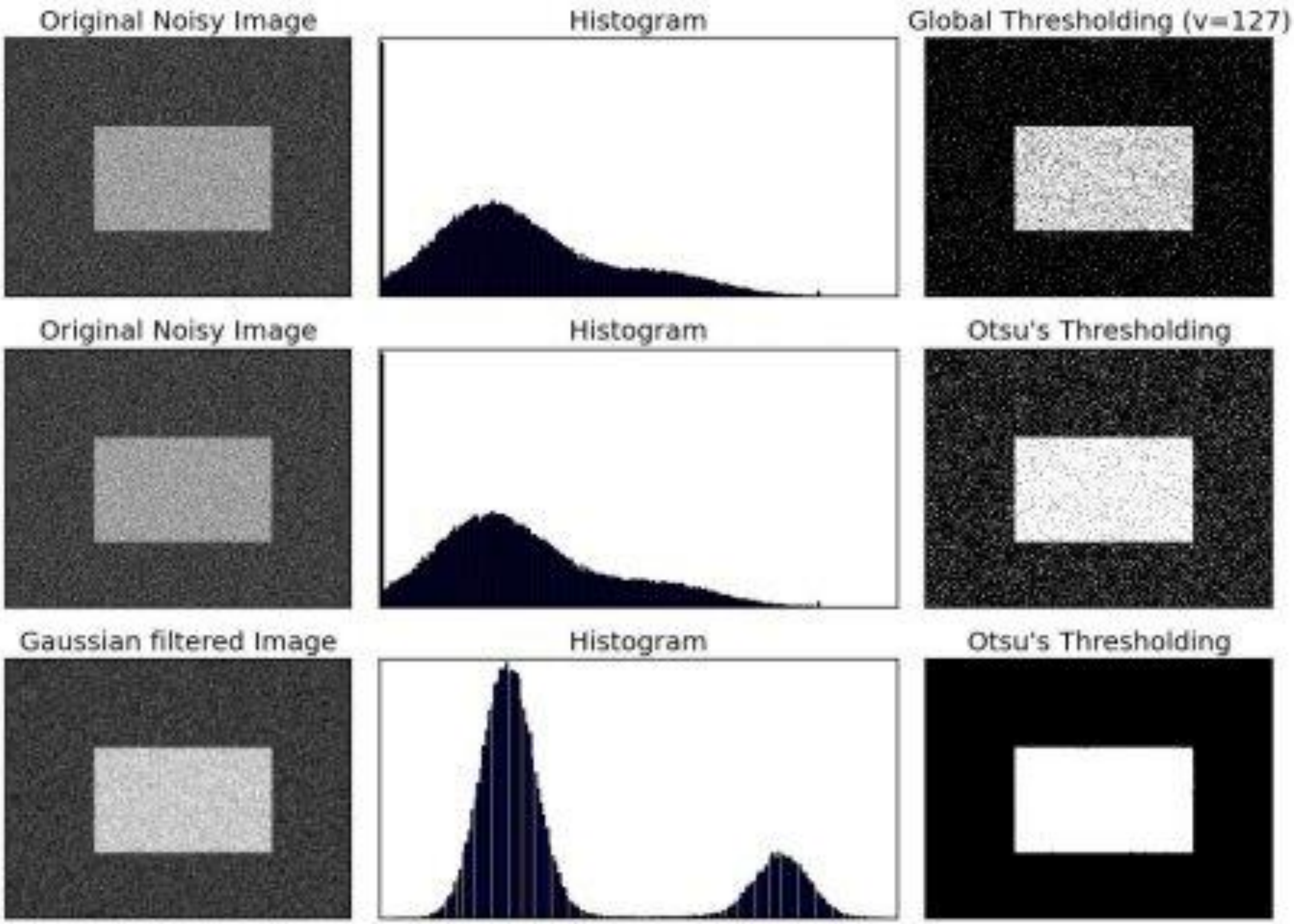


	Prewitt	**Sobel**	**Kirsch**
East	$\begin{vmatrix} 1 & 1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & -1 \end{vmatrix}$	$\begin{vmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{vmatrix}$	$\begin{vmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{vmatrix}$
Northeast	$\begin{vmatrix} 1 & -1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & 1 \end{vmatrix}$	$\begin{vmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{vmatrix}$	$\begin{vmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{vmatrix}$
North	$\begin{vmatrix} -1 & -1 & -1 \\ 1 & -2 & 1 \\ 1 & 1 & 1 \end{vmatrix}$	$\begin{vmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{vmatrix}$	$\begin{vmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{vmatrix}$
Northwest	$\begin{vmatrix} -1 & -1 & 1 \\ -1 & -2 & 1 \\ 1 & 1 & 1 \end{vmatrix}$	$\begin{vmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{vmatrix}$	$\begin{vmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{vmatrix}$
West	$\begin{vmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{vmatrix}$	$\begin{vmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{vmatrix}$	$\begin{vmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{vmatrix}$
Southwest	$\begin{vmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{vmatrix}$	$\begin{vmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{vmatrix}$	$\begin{vmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{vmatrix}$
South	$\begin{vmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{vmatrix}$	$\begin{vmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{vmatrix}$	$\begin{vmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{vmatrix}$
Southeast	$\begin{vmatrix} 1 & 1 & 1 \\ 1 & -2 & -1 \\ 1 & -1 & -1 \end{vmatrix}$	$\begin{vmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{vmatrix}$	$\begin{vmatrix} 5 & 5 & -3 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{vmatrix}$

Examples

- https://pixlr.com/editor/

Otsu’s thresholding



Niblak binarization

$$T_{Niblack} = m + k * s$$

$$T_{Niblack} = m + k\sqrt{\frac{1}{NP}\sum (p_i - m)^2}$$

$$= m + k\sqrt{\frac{\sum p_i^2}{NP} - m^2} = m + k\sqrt{B}$$

Morphological: Erosion

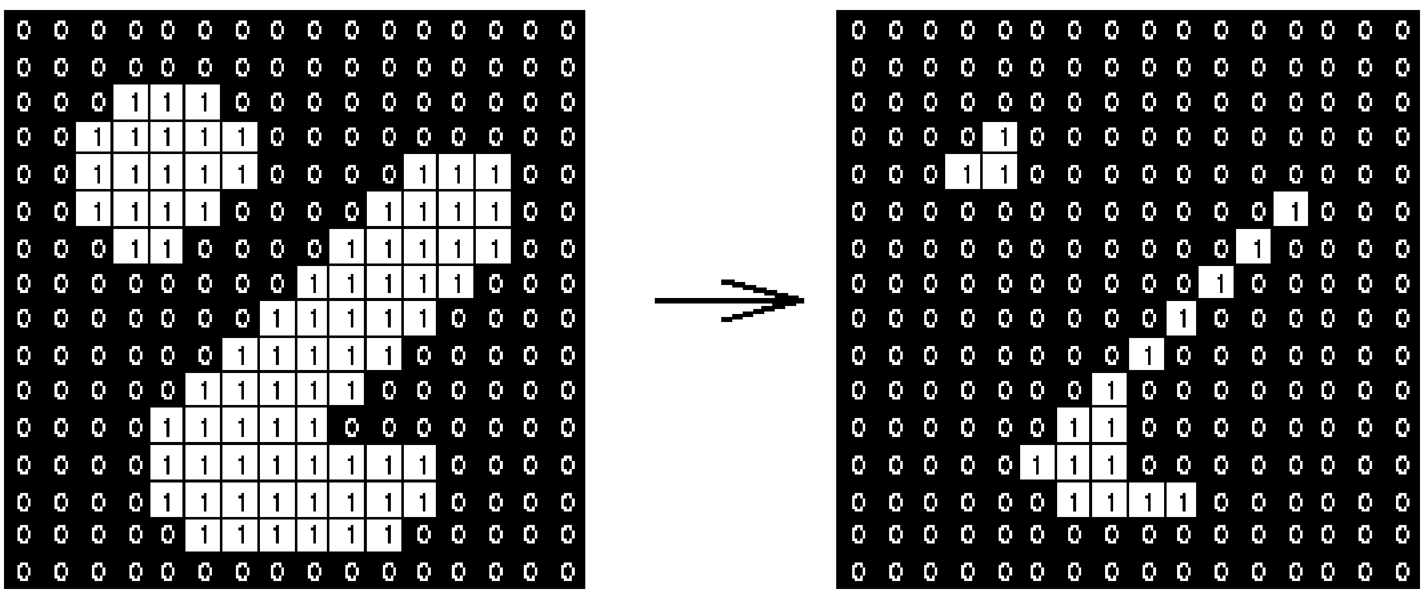
1	1	1
1	1	1
1	1	1

Set of coordinate points =

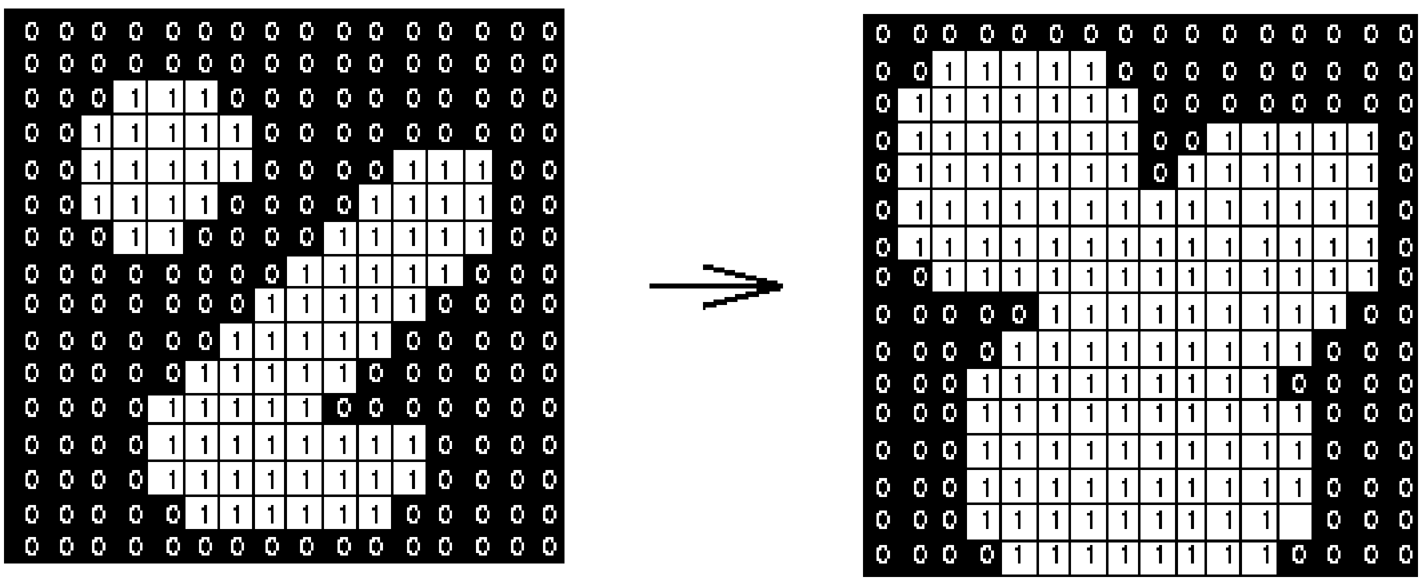
{ (-1, -1), (0, -1), (1, -1),

(-1, 0), (0, 0), (1, 0),

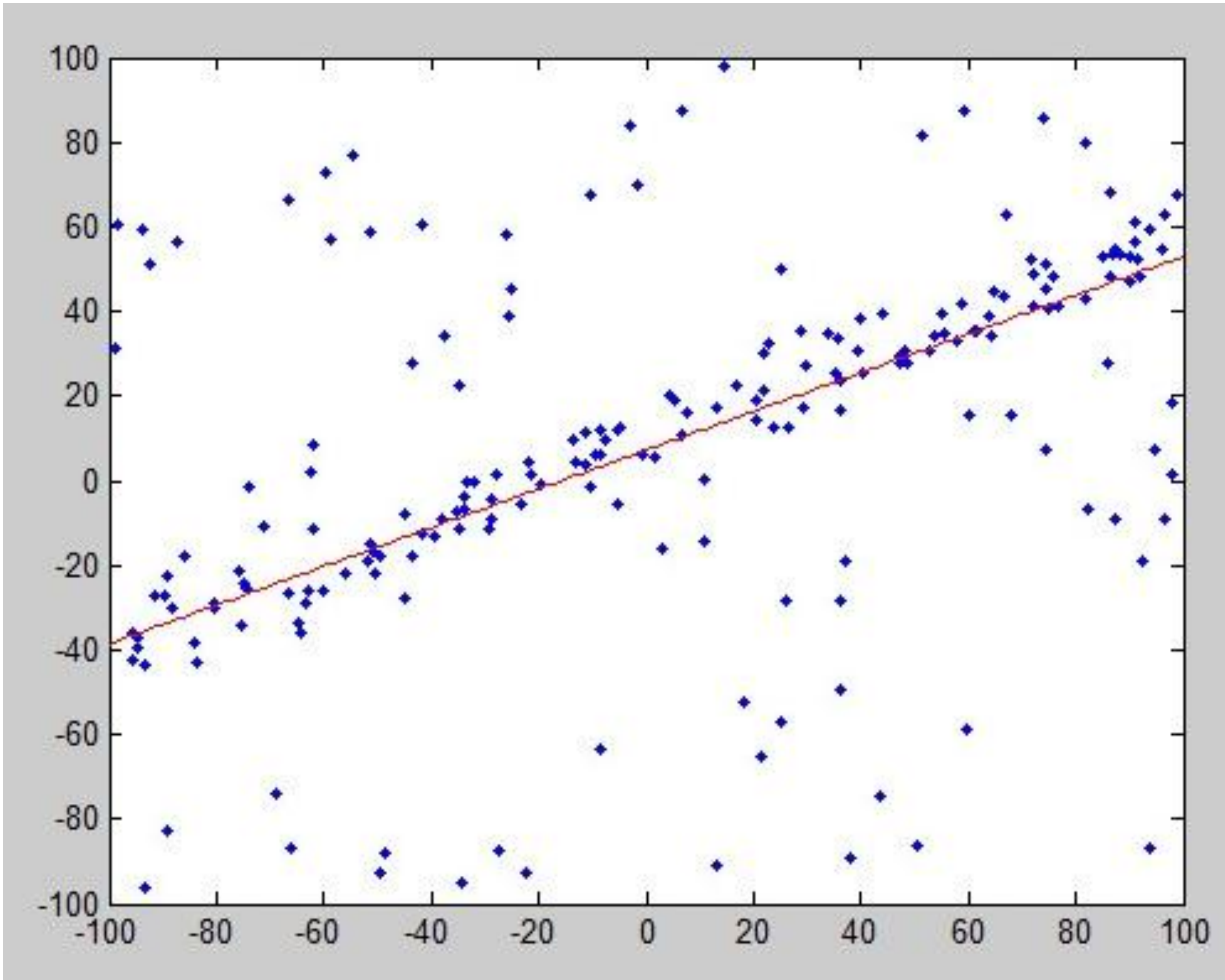
(-1, 1), (0, 1), (1, 1) }



Dilation



Line fit: Random Sample Consensus



RANSAC

- Robust fitting can deal with a few outliers – what if we have very many?
- Random sample consensus (RANSAC):
  Very general framework for model fitting in the presence of outliers
- Outline
  - Choose a small subset of points uniformly at random
  - Fit a model to that subset
  - Find all remaining points that are “close” to the model and reject the rest as outliers
  - Do this many times and choose the best model

M. A. Fischler, R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM, Vol 24, pp 381-395, 1981.

Slide: S. Lazebnik

RANSAC for line fitting

- Repeat ***N*** times:
- Draw ***s*** points uniformly at random
- Fit line to these ***s*** points
- Find inliers to this line among the remaining points (i.e., points whose distance from the line is less than ***t***)
- If there are ***d*** or more inliers, accept the line and refit using all inliers

Source: M. Pollefeys

Choosing the parameters

- Initial number of points *s*
  - Typically minimum number needed to fit the model
- Distance threshold *t*
  - Choose *t* so probability for inlier is *p* (e.g. 0.95)
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$
- Number of samples *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)

Source: M. Pollefeys

Choosing the parameters

- Initial number of points *s*
  - Typically minimum number needed to fit the model
- Distance threshold *t*
  - Choose *t* so probability for inlier is *p* (e.g. 0.95)
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$
- Number of samples *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)

$$\boxed{\left(1-\left(1-e\right)^s\right)^N = 1-p}$$

$$N = \log\left(1-p\right)/\log\left(1-\left(1-e\right)^s\right)$$

	proportion of outliers *e*						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

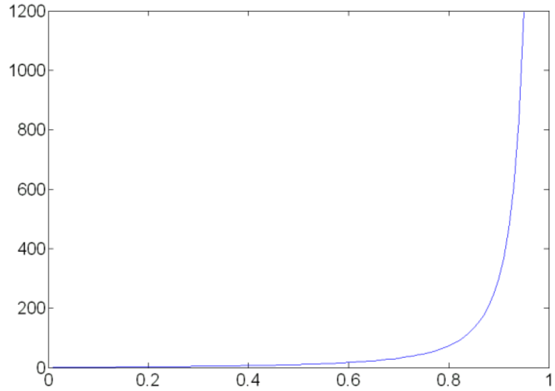
Source: M. Pollefeys

Choosing the parameters

- Initial number of points *s*
  - Typically minimum number needed to fit the model
- Distance threshold *t*
  - Choose *t* so probability for inlier is *p* (e.g. 0.95)
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$
- Number of samples *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)

$$\boxed{\left(1-\left(1-e\right)^s\right)^N=1-p}$$

$$N=\log\left(1-p\right)/\log\left(1-\left(1-e\right)^s\right)$$



Source: M. Pollefeys

Choosing the parameters

- Initial number of points s
  - Typically minimum number needed to fit the model
- Distance threshold t
  - Choose t so probability for inlier is p (e.g. 0.95)
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$
- Number of samples N
  - Choose N so that, with probability p, at least one random sample is free from outliers (e.g. p=0.99) (outlier ratio: e)
- Consensus set size d
  - Should match expected inlier ratio

Source: M. Pollefeys

Adaptively determining the number of samples

- Inlier ratio *e* is often unknown a priori, so pick worst case, e.g. 50%, and adapt if more inliers are found, e.g. 80% would yield *e*=0.2
- Adaptive procedure:
  - *N*=∞, *sample_count* =0
  - While *N* >*sample_count*
    - Choose a sample and count the number of inliers
    - Set e = 1 – (number of inliers)/(total number of points)

    Recompute *N* from *e:*

$$N = \log(1-p)/\log\left(1-(1-e)^s\right)$$

    Increment the *sample_count* by 1

Source: M. Pollefeys

RANSAC pros and cons

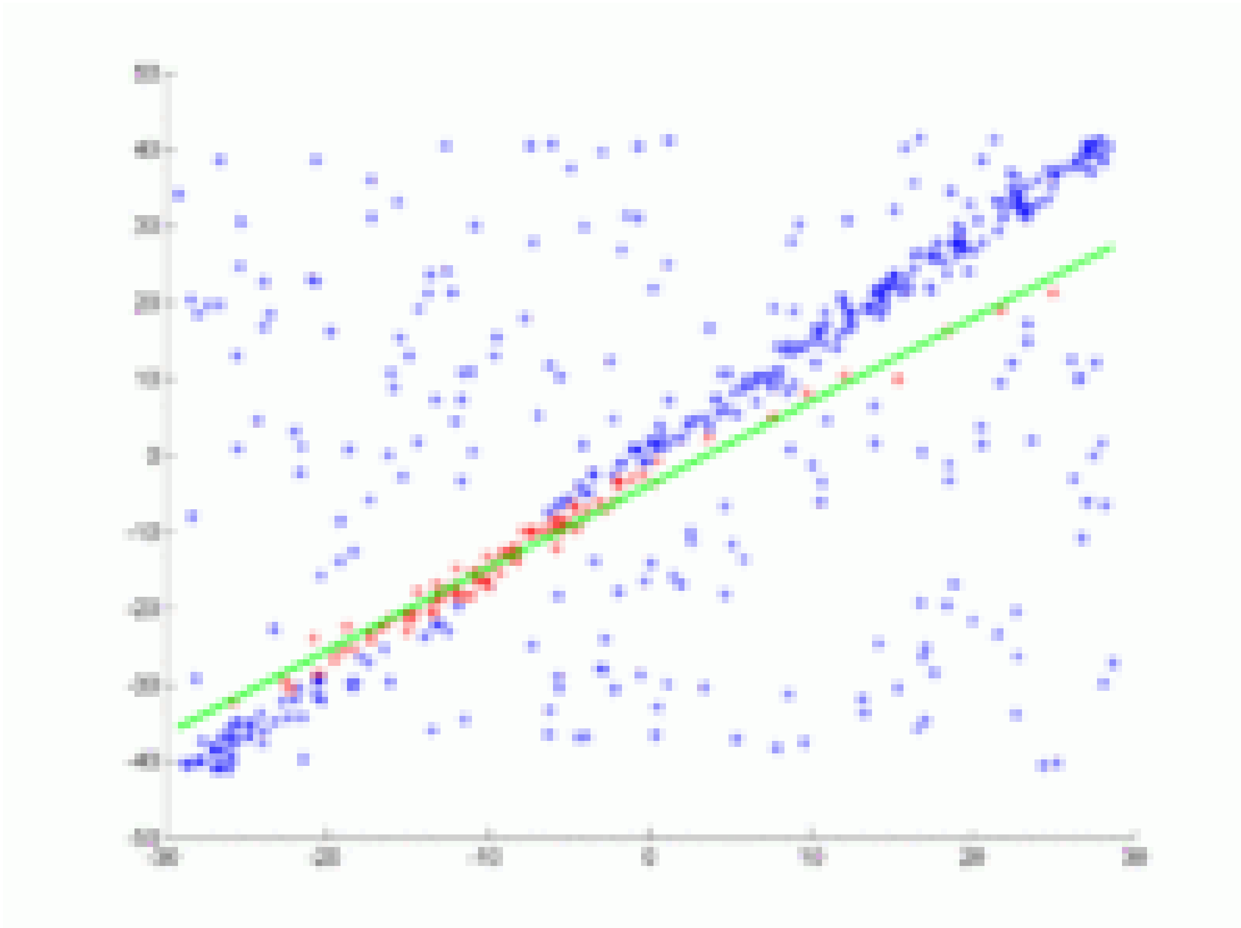
- Pros
  - Simple and general
  - Applicable to many different problems
  - Often works well in practice
- Cons
  - Lots of parameters to tune
  - Can't always get a good initialization of the model based on the minimum number of samples
  - Sometimes too many iterations are required
  - Can fail for extremely low inlier ratios
  - We can often do better than brute-force sampling

Source: M. Pollefeys

Voting schemes

- Let each feature vote for all the models that are compatible with it
- Hopefully the noise features will not vote consistently for any single model
- Missing data doesn’t matter as long as there are enough features remaining to agree on a good model

Example



Source: Wikipedia