

Questions

Projections and transformations

- What is a pinhole camera model?

A pinhole camera is a camera that uses a single small aperture to create an image. It is a simple camera with few parts and no lens, and it works by projecting an inverted image of its surroundings onto a piece of film or a digital sensor. The pinhole camera is used in computer vision to capture images of scenes in order to recognize objects and track motion.

On the camera chip we will get a blurred version of the image instead, flipped and scaled down. Adding a lens, allows to have a larger hole without blurring the image.

- What is the difference between intrinsic and extrinsic camera parameters?

Camera's intrinsic parameters:

- Focal lengths f_u , f_v are often assumed to be equal, $f = f_u = f_v$
- Skew γ captures small rotations and is often close to zero.
- Principal point (u_0, v_0) is often close to image centre.

Intrinsic camera parameters are properties of the camera itself, such as the focal length, sensor size, and principal point. These parameters are specific to the camera and can be used to determine the projection of light from 3D space onto the 2D image plane. Extrinsic camera parameters are related to the pose of the camera relative to the scene. These parameters include the camera's orientation, translation, and distortion coefficients.

- How does a 3D point get projected to a pixel with a perspective projection?

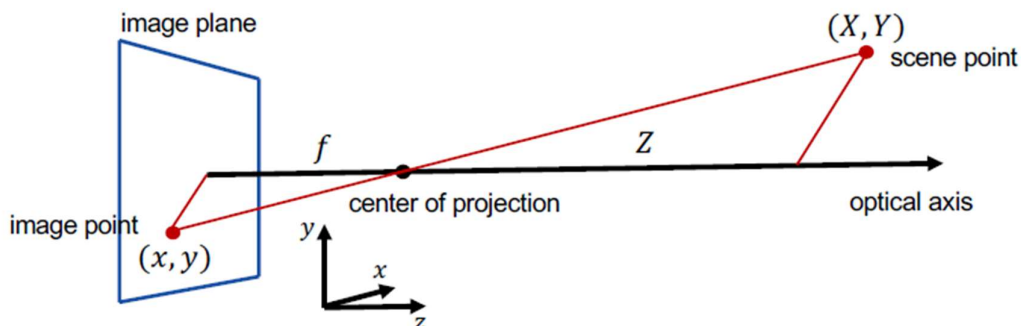
A 3D point can be projected to a 2D pixel with a perspective projection using the following formula:

$$\text{2D Pixel Position} = (F_x * X / Z, F_y * Y / Z)$$

Where F_x and F_y are the focal lengths of the camera in the x and y directions respectively (in pixels), X and Y are the 3D point's x and y coordinates and Z is the 3D point's z coordinate.



Perspective projection

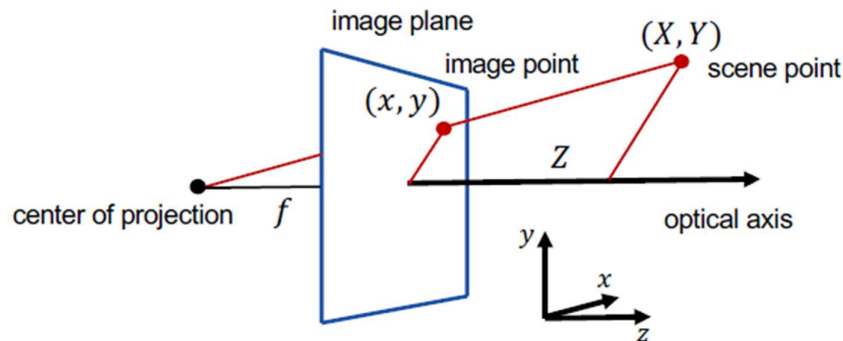


- A scene point is projected to an image point with all light passing through the pinhole (centre of projection).
- We have two triangles of equal shape.

$$\frac{x}{f} = \frac{X}{Z} \Rightarrow x = f \frac{X}{Z}$$
$$\frac{y}{f} = \frac{Y}{Z} \Rightarrow y = f \frac{Y}{Z}$$



Perspective projection



- An equivalent model that is easier to draw.
- As if you are looking through a window.

$$\frac{x}{f} = \frac{X}{Z} \Rightarrow x = f \frac{X}{Z}$$

$$\frac{y}{f} = \frac{Y}{Z} \Rightarrow y = f \frac{Y}{Z}$$

A perspective transformation has three components:

- Rotation - from world to camera coordinate system
- Translation - from world to camera coordinate system
- Perspective projection - from camera to image coordinates

Basic properties which are preserved:

- lines project to lines
- tangencies
- intersections

- What are homogeneous coordinates and what are they good for?

Homogeneous coord. are used for going from Euclidean to homogeneous coord., avoiding to make a division, which can lead to a division by zero.

Instead of having $(x, y) = f^*(X, Y)/Z$, in homogeneous coord. we have:

$$(x, y, 1) = (f^*X, f^*Y, Z)$$

In homogeneous coord. we won't have to deal with divisions, but only with matrices.

- How is a perspective projection expressed in homogeneous coordinates?

$$(x, y, 1) = (f^*X, f^*Y, Z)$$

- What is a vanishing point and how do you find it?

A vanishing point in computer vision is a point in an image where all parallel lines appear to converge. It is used in computer vision algorithms to indicate the direction of lines and edges in the image, and can be used to detect the orientation of objects.

- What is an affine camera model?
- When to choose to a perspective model vs an affine one?

An affine camera model is a mathematical model of a camera that is used to project 3D points onto a 2D image plane. This model assumes that the camera is affine-invariant, meaning that the intrinsic parameters and the projection matrix remain constant regardless of the image scale or the

camera pose. The affine camera model is used to simplify the calculation of the projection of 3D points onto the 2D image plane.

When choosing between a perspective or an affine model, the perspective model is usually preferable, as it provides a more accurate representation of 3D points onto a 2D image plane. The affine model is most useful for applications that require an approximate projection of 3D points onto a 2D image. For example, the affine model can be used for image registration or image stitching.

Image filtering

- What properties does a linear shift-invariant filter have?

The property of linear shift-invariant filter is that, given the same input but at different time instants, the system will output always the same signal but shifted.

Having an input signal, we can consider it as a combination of different impulses shifted in time and the output will be the sum of each signals related to each impulses.

$$g(x, y) = \sum_{i, j} w(i, j) f(x + i, y + j)$$

This property applied to a linear filter means that the weights are the same for each pixel. The output is a weighted sum neighbours.

- How do you define a convolution?

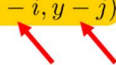
Way of considering convolution: weighted sum of shifted copies of one function, with weights given by the function value of the second function.

So far we have used a cross-correlation

$$g(x, y) = \sum_{i, j} h(i, j) f(x + i, y + j)$$

$$g = h \otimes f$$

But usually we do convolutions instead

$$g(x, y) = \sum_{i, j} h(i, j) f(x - i, y - j)$$


$$g = h * f$$

Cross-correlation and convolution are both operations applied to images. Cross-correlation means sliding a kernel (filter) across an image. Convolution means sliding a flipped kernel across an image. Same filter but flipped x-wise and y-wise.

Convolution is commutative, so we can apply the filter in both ways.

- Why are convolutions important in linear filtering?

Convolutions are important in linear filtering because they allow linear filters to be applied to signals or images in a localized manner, meaning that the output of the filter at any point in the image or signal is determined only by the values at a small region around the point. This localized approach makes it possible for linear filters to extract features from signals or images, such as edges or textures, and to reduce noise.

- How do you define a 2D Fourier transform?

We can represent an image (signal) as the sum of sinusoids. What we get is a new representation in the frequency domain.

Fourier Transform in 2D

$$\mathcal{F}(f(x)) = \int_{x \in \mathbb{R}^2} f(x) e^{-i\omega^T x} dx = \hat{f}(\omega)$$

$$\mathcal{F}^{-1}(\hat{f}(\omega)) = \frac{1}{(2\pi)^2} \int_{\omega \in \mathbb{R}^2} \hat{f}(\omega) e^{i\omega^T x} d\omega = f(x)$$

$$e^{i\omega^T x} = \cos \omega^T x + i \sin \omega^T x, \quad \omega^T x = \omega_1 x_1 + \omega_2 x_2$$

Terminology:

- Frequency spectrum: $\hat{f}(\omega) = \text{Re}[\hat{f}(\omega)] + i \text{Im}[\hat{f}(\omega)] = |\hat{f}(\omega)| e^{-i\phi(\omega)}$
- Angular frequency: $\omega = \begin{pmatrix} \omega_1 \\ \omega_2 \end{pmatrix}$
- Frequency: $f = \frac{\omega}{2\pi}$
- Wavelength: $\lambda = \frac{2\pi}{\|\omega\|} = \frac{2\pi}{\sqrt{\omega_1^2 + \omega_2^2}}$

- If you apply a Fourier transform to an image, what do you get?

The Fourier transform of an image will be represented by a periodic function (sum of sinusoids) in the Fourier domain.

This representation consists of a set of values that represent the magnitude and phase of each frequency component of the image. The Fourier transform allows for detailed frequency analysis of the image, allowing you to identify and remove unwanted frequency components from the image.

- What information does the phase contain? And the magnitude?

The Magnitude determines the intensity of the image, while the phase contains important details about edges and the shape of the image.

Phase defines how waveforms are shifted along its direction.

- Where edges will end up in the image (most important).

Magnitude defines how large the waveforms are.

- What grey-levels are on either side of edges (less important).

- What is the Fourier transform of a convolution? Why important?

The Fourier transform of a convolution corresponds to a point-wise multiplication in the Fourier domain and viceversa.

Implementation: when size of the filter is too large it is more effective to use multiplication in the Fourier domain.

- What does separability of filters mean?

It means having a filter $h(x,y)$ s.t. $h(x,y) = h(x)h(y)$. This is important because we can split the 2D convolution in 2 1D convolutions.

A Fourier transform in 2D can always be performed as a series of two 1D Fourier transforms

$$h(x, y) = h_1(x)h_2(y)$$

$$(h*f)(x, y) = \sum_{i=0}^m \sum_{j=0}^m h(i, j)f(x-i, y-j) = \sum_{i=0}^m h_1(i) \left(\sum_{j=0}^m h_2(j)f(x-i, y-j) \right)$$

- If convolution mask $h(x, y)$ can be separated as above, 2D convolution can be performed as a series of 1D convolutions.
- Discrete case: If the mask is m^2 in size $\rightarrow 2m$ operations per pixel, instead of m^2 operations per pixel.

- How do you interpret a point in the Fourier domain in the spatial domain?

A point in the Fourier domain represent a sinusoidal wave in the spatial domain.

In the Fourier domain, a point is a frequency component of a signal. To interpret a point in the spatial domain, you would need to use the inverse Fourier transform to transform the frequency components back into the spatial domain. This would allow you to see the time-domain signal, which gives you an idea of what the signal looks like in the spatial domain.

- How do you apply a discrete Fourier transform?

Discrete Fourier Transform (DFT)

$$\hat{f}(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-2\pi i \left(\frac{xu}{M} + \frac{yv}{N} \right)}$$

$$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \hat{f}(u, v) e^{+2\pi i \left(\frac{xu}{M} + \frac{yv}{N} \right)}$$

Terminology:

- Frequency spectrum: $\hat{f}(u, v) = \text{Re}[\hat{f}(u, v)] + i \text{Im}[\hat{f}(u, v)] = |\hat{f}(u, v)| e^{-i\phi(u, v)}$
- Power spectrum: $P(u, v) = |\hat{f}(u, v)|^2 = \text{Re}[\hat{f}(u, v)]^2 + \text{Im}[\hat{f}(u, v)]^2$

The magnitude $|\hat{f}(u, v)|$ is simply the peak value, and the phase $\phi(u, v)$ determines where the origin is, or where the sinusoid starts.

- What happens to the Fourier transform, if you translate or rotate an image?

Rotation in the spatial domain determines a rotation in the fourier domain, but doesn't affect the magnitude and the phase. In the fourier domain the rotation determines some distortion.

Translation only affects phase, not magnitude.

- In what sense is the Fourier transform symmetric?

- The DFT and its inverse are periodic with period N , for an $N \times N$ image. This means:

$$\hat{f}(u, v) = \hat{f}(u + N, v) = \hat{f}(u, v + N) = \hat{f}(u + N, v + N)$$

- This is easy to understand since

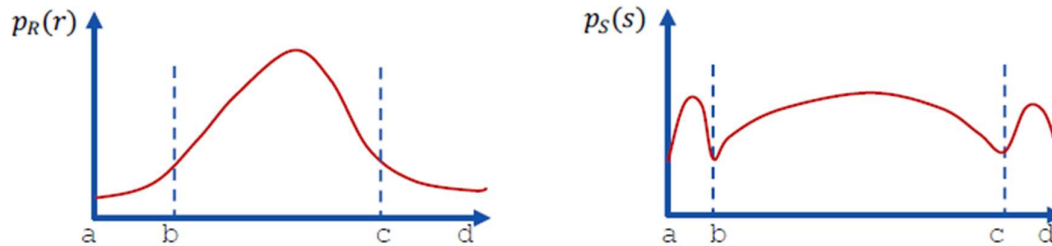
$$e^{-2\pi i \left(\frac{xu}{N} \right)} = e^{-2\pi i \left(\frac{x(u+kN)}{N} \right)}$$

- This property defines the implied symmetry in the Fourier spectrum as well as that $\hat{f}(u, v)$ repeats itself infinitely.
- However, only one period is enough to reconstruct the original image $f(x, y)$.

Image enhancement

- Mention a typical grey-level transformation. When to use it?

Grey level transformation: $s = T(r)$, where s and r are intensities and T the transformation.



An example of transformation is using a logarithmic function:

$$s = T(r) = c \cdot \log(r+1)$$

A grey-level transformation of this type can be used to stretch the interval of values on the histogram and make details more visible.

- What do histogram stretching and compression mean? Look Above
- What are the principles of histogram equalization?

Histogram equalization (continuous case)

Idea: Find transformation $s = T(r)$ such that the distribution $p_S(s)$ of pixel values is uniform, $p_S(s) = 1$, given a distribution from an image $p_R(r)$, $r \in [0,1]$.

A small range in R of width Δr is mapped to a small range in S of width Δs . The number of pixels in these ranges must be the same.

$$N = p_R(r) \Delta r = p_S(s) \Delta s$$

For really small intervals we have

$$\frac{p_R(r)}{p_S(s)} = p_R(r) = \frac{\Delta s}{\Delta r} \rightarrow \frac{ds}{dr} = T'(r), \text{ as } \Delta r \rightarrow 0$$

Thus the transformation is given by

$$s = T(r) = \int_0^r p_R(r') dr'$$

- What are the differences between lowpass, bandpass and highpass filters?

The main difference is the range of frequencies they are retaining. Lowpass filter eliminates high frequencies and results in eliminating noise and sharp edges. If we filter too much will result in blurring the image. Highpass filter eliminates lower frequencies, corresponding with shadow and slow changing values. Bandpass combines both.

- Why are ideal lowpass filters rarely used in practice?

Ideal lowpass filters are rarely used in image analysis due to their inability to effectively filter out high-frequency noise. Ideal lowpass filters tend to blur the edges of objects in an image, which can negatively affect the accuracy of any analysis performed on the image. Additionally, the ideal lowpass filter does not account for any non-linearities that may exist in the image, which can lead to inaccurate results.

- What characteristics does a Gaussian filter have?

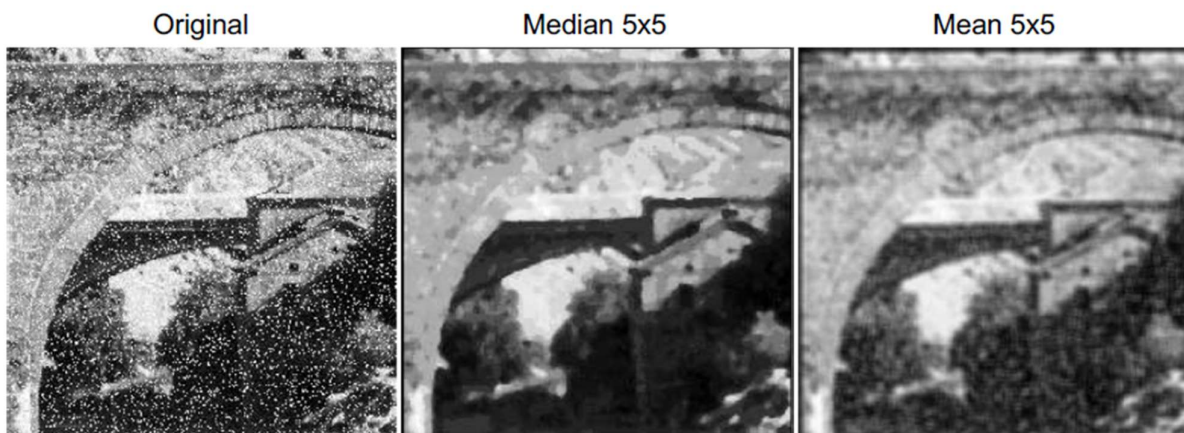
A Gaussian filter is a linear filter used for smoothing or blurring an image. It is based on the Gaussian function, which is a bell-shaped curve that tapers off gradually. The Gaussian filter has the following characteristics:

- It is a low pass filter, meaning it attenuates high frequency components of an image.
- It is non-linear, meaning it does not have a constant output for a given input.
- It is isotropic, meaning it does not favor any particular direction.
- It has a variable width, meaning the blurriness of the output can be adjusted.
- It has a Gaussian distribution, meaning it has a bell-shaped curve that tapers off gradually.

- What is the difference between mean and median filters?

First of all Mean filtering is a linear filter, while median is non-linear.

Mean filters are used to smooth out noise or unwanted details in an image. They work by replacing each pixel in the image with the average value of the surrounding pixels. Median filters are used to reduce random noise in an image. They work by replacing each pixel in the image with the median value of the surrounding pixels. The median filter is usually more effective at preserving edges and details than the mean filter.



Median filtering can be very good at ignoring occasional really wrong pixels.

- How can you do sharpening?

Methods:

- Unsharp masking (spatial): Blur image → subtract from original → weight the difference → add to original
- Highpass filtering (spectral): similar to unsharp masking but in fourier domain
- Differentiation (image derivatives)

- How can you approximate a first order derivative?

- But for discrete images h has to be an integer. The best we can do is

$$f'(x) \approx \frac{f(x+1) - f(x-1)}{2}$$

- This corresponds to a filter kernel

$$\delta_x \quad \frac{1}{2} \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

Why does the 1 come first?

The Sobel filter

- Problem: A derivative is a highpass filter and will enhance noise.
- Solution: Add some smoothing, but do it in the opposite direction.

$$\frac{1}{8} \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \frac{1}{2} \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

- A derivative in the y-direction can be implemented correspondingly.

- What is a Laplacian?

Commonly used differential operators

- We apply a differential operator δ_x to an image f and get

$$f_x = \delta_x * f$$

- Gradient: $\nabla f = (f_x, f_y)$. often use to compute edge direction
 - First order, linear, non-isotropic
- Gradient magnitude: $|\nabla f| = \sqrt{f_x^2 + f_y^2}$ often used to detect edges
 - First order, non-linear, isotropic
- Laplacian: $\Delta f = f_{xx} + f_{yy}$ often used to detect blobs
 - 2nd order, linear, isotropic
- Isotropic means that the result does not depend on the orientation. It is good if we want to detect an edge and want it to behave similarly for all edge directions.

Image features I

- Why do we get image aliasing when subsampling and what to do about it?

Image aliasing is a visual artifact that occurs when the resolution of an image is too low for the level of detail being captured. It appears as jagged edges or stair-stepping along curves and diagonal lines, and is most commonly seen in digital images.

By just subsampling you get image aliasing due to undersampling.

- Why is the notion of scale important in image analysis and computer vision?

In image matching the scale represents an important factor when you want to find common features.

Scale is important in image analysis and computer vision because it allows us to measure features and objects in an image with respect to a known size. It allows us to compare images of different sizes and resolutions, and to accurately calculate distances, areas and other properties of objects

in the image. Scale can also be used to identify objects in an image by determining their relative size.

- What is a scale-space representation? On what basis is it constructed?

A scale-space representation is a way of representing an image or signal in multiple scales, or levels of detail. It is constructed by convolving the image or signal with a set of filters at different scales. This is done in order to capture the different features of the data at different levels of detail. This helps to identify features at different scales, allowing for more accurate analysis and comparison of data.

- What structural requirements are natural to impose on early visual operations?
- What is meant by a Gaussian derivative? Why are these important?

Note: since both operations are linear, we may either blur first and then compute derivatives, or vice versa.

These can be used as a general basis for expressing image operations such as feature detection, feature classification, surface shape, image matching and recognition.

- Why is edge detection important for image understanding?

Edge detection is important because edges determine surface in real world, and region/object in the image itself.

Why edges: Edge features constitute important features to humans, Independent of illumination, Easy to detect computationally, Used to form higher level features

- What families of methods exist for edge detection?

Families of methods for edge detection include gradient-based methods, morphological methods, Laplacian-based methods, Marr-Hildreth operator, Sobel operator, and Canny edge detector.

- What information do image gradients provide?

Image gradients provide information about the rate of change in an image. They can be used to detect edges, determine the orientation of a feature, and measure the amount of contrast in an image.

- How does the Canny edge detector work?

1. Convolve image by smoothing kernel.
2. Estimate edge strength and edge normal direction.
3. Threshold on edge strength and keep only edge points that are local extrema in the edge normal direction (non-maximum suppression implemented by local search).

- What is differential edge detection?

Differential edge detection (Lindeberg 1993)

- Non-maximum suppression: An edge point is a point where the gradient magnitude assumes a maximum in the gradient direction.

- This can be expressed in terms of:

– Gradient $\nabla L = (L_x, L_y)^T$ and gradient magnitude $|\nabla L| = \sqrt{L_x^2 + L_y^2}$

– The normalized gradient direction: $e_v = \frac{\nabla L}{|\nabla L|} = \frac{(L_x, L_y)^T}{\sqrt{L_x^2 + L_y^2}}$

– Directional derivative in any direction α :
 $\delta_\alpha = \cos \alpha \delta_x + \sin \alpha \delta_y$

– Directional derivative in gradient direction:
 $\delta_v = \frac{L_x}{\sqrt{L_x^2 + L_y^2}} \delta_x + \frac{L_y}{\sqrt{L_x^2 + L_y^2}} \delta_y$ with $L_v = \delta_v L = \sqrt{L_x^2 + L_y^2} = |\nabla L|$

Requirements for gradient magnitude to be maximal in gradient direction:

$$\begin{cases} \delta_v(L_v) = 0 \\ \delta_{vv}(L_v) < 0 \end{cases} \quad \text{or} \quad \begin{cases} L_{vv} = 0 \\ L_{vvv} < 0 \end{cases}$$

In terms of coordinates:

$$\begin{aligned} L_{vv} &= (\cos \alpha \delta_x + \sin \alpha \delta_y)^2 L = \cos^2 \alpha L_{xx} + 2 \cos \alpha \sin \alpha L_{xy} + \sin^2 \alpha L_{yy} = \\ &= \frac{L_x^2}{L_x^2 + L_y^2} L_{xx} + \frac{2L_x L_y}{L_x^2 + L_y^2} L_{xy} + \frac{L_y^2}{L_x^2 + L_y^2} L_{yy} = \frac{L_x^2 L_{xx} + 2L_x L_y L_{xy} + L_y^2 L_{yy}}{L_x^2 + L_y^2} = 0 \end{aligned}$$

Since the denominator is irrelevant, the edges are given by:

$$\begin{aligned} \widetilde{L}_{vv} &= L_x^2 L_{xx} + 2L_x L_y L_{xy} + L_y^2 L_{yy} = 0 \\ \widetilde{L}_{vvv} &= L_x^3 L_{xxx} + 3L_x^2 L_y L_{xxy} + 3L_x L_y^2 L_{xyy} + L_y^3 L_{yyy} < 0 \end{aligned}$$

Differential edge detection in practice:

1. Convolve image by a Gaussian kernel.
2. Compute partial derivatives up to order three and combine these into the differential invariants $\widetilde{L_{vv}}$ and $\widetilde{L_{vvv}}$ at every image point.
3. Search for the zero-crossings of $\widetilde{L_{vv}}$ that satisfy $\widetilde{L_{vvv}} < 0$.
 - Gives subpixel accuracy and connected edge segments automatically.
 - Avoids issues of orientation estimation and handling as well as edge tracking in discrete non-maximum suppression.
4. Can be combined with either a single low threshold on the gradient magnitude or hysteresis thresholding using two thresholds.

- What should the image derivatives be equal to on edge points?

Image derivatives should be equal to zero on edge points. This is because at an edge, the intensity of the pixel is changing abruptly. Therefore, the derivative of the intensity with respect to position will be zero, since the intensity is not changing with respect to position.

Image features II

- What are the motivations for computing interest points? What are they typically used for?

We want to compute interest points to find matching patches between images. We want to find correlations, common features across images.

The main motivation for computing interest points is to make a robust and efficient comparison between two images. Interest points can be used to determine the similarities and differences between two images and to identify objects within an image. They are typically used for object recognition, image registration, image stitching, and other computer vision applications.

- Describe three common interest point detectors including their mathematical definitions.

Common approach to image-based matching and recognition:

1. Detect interest points.
2. Compute image descriptors around the interest points.
3. Match the image descriptors.

SIFT (Lowe 2004), SURF (Bay et al 2008), Harris corner Detection, Laplacian Blob detection.

Harris Corner Detection: (Compute scale-space representation)

- Compute partial derivative L_x, L_y
- Compute products $L_x^2, L_x * L_y, L_y^2$ at every image point
- Compute weighted sums of products using window function at a scale s
- At every image point compute Harris operator H
- Detect local extrema of H s.t. $H \geq H_0$

Laplacian Blob Detection: (Compute scale-space representation)

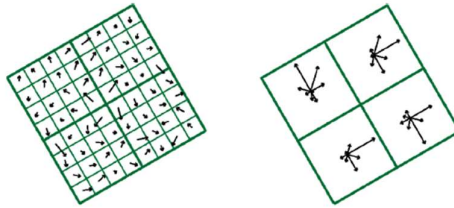
- Compute Laplacian $\nabla^2 L = L_{xx} + L_{yy}$
- Find spatially local extrema $\nabla^2 L$ and regard these as blob responses with
$$\nabla^2 L < 0$$
$$\nabla^2 L > 0$$

SIFT Descriptor:

- Compute gradient ∇L around (x_0, y_0) at scale t
- Create 36-bin histogram (gradient directions) (Peaks where we have max value. Orientation estimates)
- Define 4x4 grid and compute gradients again
 - Take 8-bin histograms
- Trilinear interpolation for distributing weighted bins

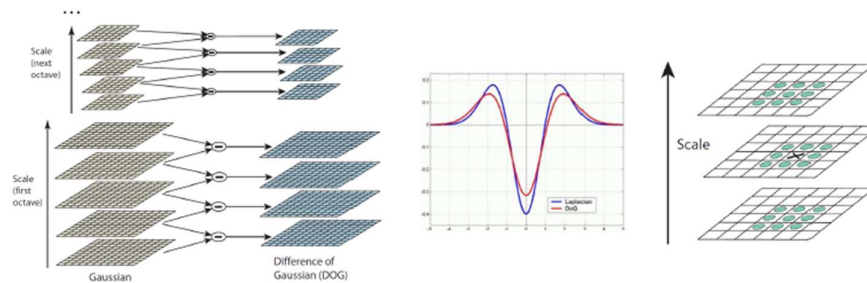
The SIFT descriptor

- Place a window around the interest point based on the predicted orientation.
- Collect gradient directions in a grid of histograms.
- Here, schematically illustrated for a 2x2 instead of a 4x4 grid:



4x4 grid of histograms with 8 bins for gradient directions \Rightarrow 128D descriptor

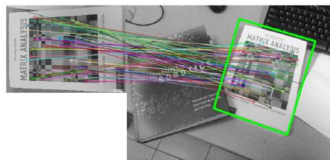
- Previously, we talked about the Laplacian being used for blob detection.
- SIFT uses a faster approximation based on Differences of Gaussians (DoG).
Interpretation: “whatever was removed when going from one scale to the next”.



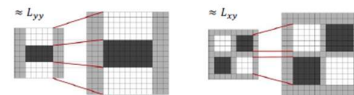
The SURF descriptor

Faster alternative to SIFT, suitable for low-end devices.

- Compute derivatives L_x and L_y around (x_0, y_0) at scale t of the interest point.
- Orientation normalization similar to SIFT.
- Sums of derivatives $\sum L_x$, $\sum |L_x|$, $\sum L_y$, $\sum |L_y|$ over 4x4 over sub-windows around the interest point.



- Interest point detection by determinant of Hessian
$$\det \mathcal{H}L = L_{xx}L_{yy} - L_{xy}^2$$
- Derivatives approximated by Haar wavelets (box filters) to allow for fast computations by integral images. (Bay et al. 2008)



Based on fast computations of pixels over rectangles.

- Fast, but not as accurate as SIFT:
 - Less invariant to rotations
 - Ringing phenomena due to box filters
 - Worse scale-space properties over scale

- Why is scale selection an important operation?

Scale selection is important in matching features because we want to find a mapping between features invariant from the scale.

- Describe how scale selection can be performed in practice.

In practice we should compute normalized derivatives, to make image features scale invariant.

- Solution: Use scale-normalized derivatives

$$\delta_{\xi} = t^{1/2} \delta_x, \delta_{\eta} = t^{1/2} \delta_y$$

and the detectors with their scale-normalized counterparts, e.g.

$$\nabla_{norm}^2 L = t (L_{xx} + L_{yy})$$

$$\det \mathcal{H}_{norm} L = t^2 (L_{xx} L_{yy} - L_{xy}^2)$$

- Detect scale-space extrema, i.e. not only over image space, but also over scale.
- This allows for the inherent size of local image structures to be determined and a scale-invariant reference frame to be created.

- What is the motivation for using image pyramids in computer vision?

Image pyramids are used in computer vision to enable the efficient and accurate analysis of image data at multiple scales. By creating multiple versions of an image at different resolutions, image pyramids allow us to analyze the same image contents at different scales and levels of detail. This can be useful in applications such as object detection, image registration, and image segmentation, where a given object may appear differently at different scales.

- How are image pyramids computed from image data?

Image pyramids are created by applying a smoothing filter (e.g., Gaussian or Laplacian) to an image and then down-sampling the resulting image to create a lower-resolution version of the image. This process is repeated until the desired scale of the image pyramid is reached. The resulting set of images is then stacked up, forming the image pyramid.

- Describe a basic trade-off issue that arises in hybrid pyramids.

One of the basic trade-offs in hybrid pyramids is between accuracy and efficiency. On the one hand, using a more accurate algorithm such as Gaussian or Laplacian may result in higher quality pyramids; however, this can come at the cost of increased computation time. On the other hand, using a simpler algorithm such as box filters may result in lower quality pyramids; however, the computation time will be reduced. Therefore, it is important to consider the trade-off between accuracy and computation time when choosing a hybrid pyramid algorithm.

- What is the purpose of computing image descriptors at interest points?

The purpose of computing image descriptors at interest points is to characterize the appearance of an image region in a way that is invariant to changes in viewpoint, illumination, and other factors. This allows for accurate identification of objects in an image or matching of two images.

- How is the SIFT descriptor defined from image data?

The SIFT descriptor is defined by finding the local gradients of an image at the interest points, and then assigning a histogram of these gradients to each point. The descriptor is composed of 8×8 bins that contain a histogram of directions of the gradients at that point.

- How is the SURF descriptor defined from image data?

The SURF descriptor is defined by first calculating the Haar wavelet responses of an image at the interest points. Then, a vector containing the responses of the wavelet is assigned to each point. The descriptor is composed of 64 elements containing the responses of the wavelet at the point.

- Outline the basic steps in an algorithm that matches interest points with associated image descriptors between two images of the same scene.

1. Detect interest points in both images.
2. Compute image descriptors for each interest point in both images.
3. For each interest point in the first image, search the descriptor of the second image to find the closest descriptor.
4. Store the matching pairs of interest points.
5. Filter out false matches by using a distance ratio test.
6. Compute the homography matrix between the two images.
7. Use the homography matrix to warp the second image on to the first.
8. Evaluate the quality of the warping.
9. Use the warped image to further refine the matched pairs of interest points.

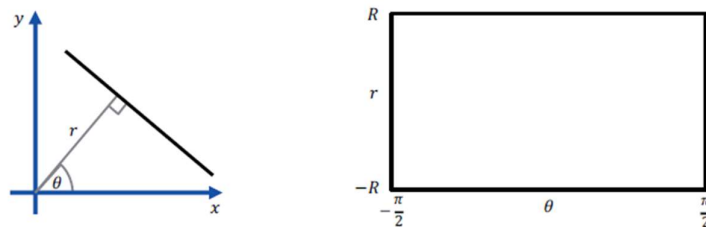
Model fitting and representation

- How does a Hough transform work for lines?

During edge detection instead of segment we can have fragment of them, or sometimes many points. We compute Hough transform to find lines relying on them; lines which strength depends on how many points rely on them.

Problem: parameterization in duality space

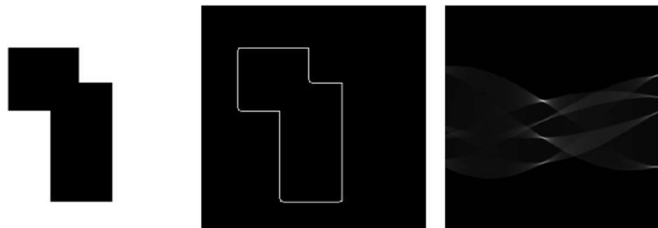
- Observation: vertical lines correspond to $k \rightarrow \pm\infty$
- Better method: use parameterization $x \cos \theta + y \sin \theta = r$
 θ = orientation of line, r = perpendicular distance to origin



- Point (x, y) in the image \Rightarrow curve $r(\theta) = x \cos \theta + y \sin \theta$ in the Hough (accumulator) space.

- How many accumulators should you use?

Which line corresponds to which point in Hough space?

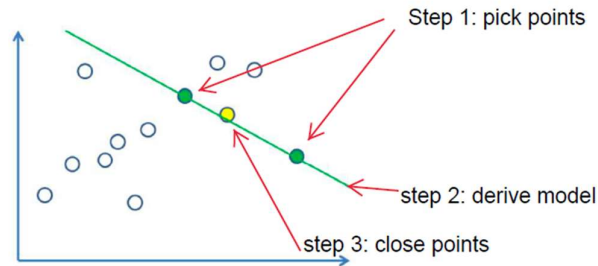


Question: How fine should θ and r be quantized?

- Too coarse: poor resolution in line directions.
- Too fine: not enough samples in accumulators.

- How does RANSAC work?

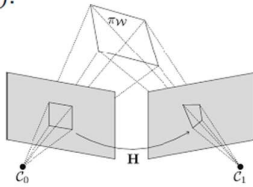
1. Pick a minimum number of points
2. From those points derive the parameters => Model
3. Go through all the other points and see how many of them are close enough to the model
 - a. Count how many edge points are within – say 2 pixels – from this hypothesized line
4. Repeat 1-3 for S iterations
5. Pick the solution with highest number of matching points in 3



Three inliers, nine outliers.

- What is a homography?

$\pi_1 \pi_2$



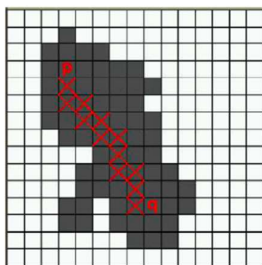
Assume to have a plane in 3D space viewed by two cameras.

An homography H is a transformation from the coordinates of 1 camera to the other. A point (x_i, y_i) in the plane on one camera corresponds to a point (x'_i, y'_i) in the other camera.

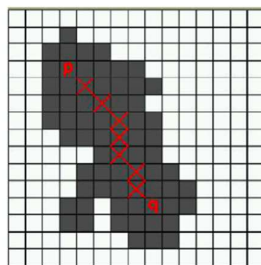
$$(x'_i, y'_i, 1)^T = H (x_i, y_i, 1)^T$$

Homography can be used to match features found with SIFT/SURF..

- How do you create a connected component? Two components are connected if you can find a path between them using the neighbourhood system. Path:consecutive pairs of component



4-neighbourhood



8-neighbourhood

4-neighbourhood: longer path

8-neighbourhood: shorter path

Finding connected components (recursively scan the image):

1. Label all the foreground pixels
2. Recursively assign label L to all neighbours
3. Stop if there is no unlabelled foreground pixels

- In what ways can two shape descriptors be different?

Shape descriptors can differ in the number of parameters they use to describe the shape, the types of parameters they use, the algorithms used to calculate them, and the range of shapes they can represent. For example, one shape descriptor may measure the perimeter of a shape while another may measure the area.

- Can you give two examples of two shape descriptors?
- How do you compute moment descriptors?

Moment descriptors are computed by taking the integral of an image over its pixels and applying a weighting function to each pixel. The result of the integral is a set of moments that can be used to describe the shape, size, orientation, and other features of the image. For example, the area of an image can be computed by summing up all the pixels in the image, and the centroid of the image can be computed by taking the average of the x- and y-coordinates of the pixels. The Hu moments are a set of seven moment descriptors that are commonly used to describe the shape of an image. These moments are computed by taking the second order moments of the image and normalizing them so that they are invariant to scale and rotation.

- What do you like to preserve with dimensionality reduction?

The goal is to reduce the dimensionality of data, while retaining as much as possible of the variation present in the dataset

- How does a PCA work? See slides.

Image segmentation

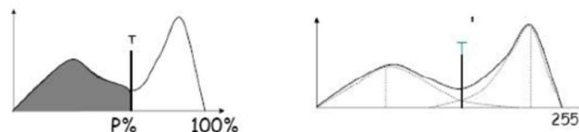
- What is the purpose of image segmentation?

The purpose of image segmentation is to divide image into fragments/segments. To cut off semantic objects,

- What is Gestalt Theory? We perceive the whole scene, not just the individual parts. Sometimes it is impossible to say anything about the scene, by only looking at the parts.
- How to select a threshold for histogram based segmentation?

Thresholding – two simple methods

- If these methods work in a practical application, why not use them?
- P-tile method
 - Assume the object is either lighter or darker than the background and has a size that covers P% of the whole image.
 - Choose a threshold such that P% of the pixels are above or below the threshold.
- Mode method
 - Find the peaks and valleys of the histogram.
 - Set the threshold at the point of the deepest value with respect to the nearby peaks.



- How does K-means work?

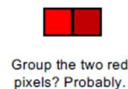
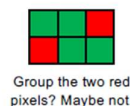
K-means clustering

- Group pixels based on similarity in colours (or any other measure).
- K-means algorithm:
 1. Choose K initial mean values (possibly randomly).
 2. Assign each pixel to the mean that is closest.
 3. Update means as the average of pixels assigned to each mean.
 4. Iterate until there is no change in mean values.
- Problem: segments can be split up into pieces.

- Why is spatial coherence important for segmentation?

It's important because determines the way in which we consider neighbour pixels. This might prevent having segments splitted into pieces.

- Methods mentioned so far neglect dependency between neighbouring pixels.
- This may cause segments to be split up into different pieces.
- If spatial coherence between pixels is taken into account, this can be avoided.
- The dependency between neighbouring pixels or regions can be modelled in many different ways.



- What does a mean-shift algorithm try to do and how?

A mean-shift algorithm is a clustering technique that attempts to discover clusters in a given dataset. It does this by iteratively shifting the data points closer to the mean of their local neighborhood until the points have converged to a local maximum. This local maximum is then taken to be the center of the cluster. The algorithm can be applied to any dataset, however it is most commonly used in image processing.

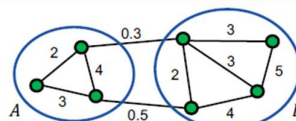
- What is an affinity matrix?

Affinity matrix represents the link between nodes as similarities, which can be expressed with different measures.

- What does Normalized Cuts try to optimize?

Normalized cuts

- Goal: Maximize sum of within cluster similarities, while minimizing sum of across cluster similarities.
- Minimize Normalized Cut:
$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(B, A)}{assoc(B, V)}$$
 where
 - A and B are two disjoint sets of vertices and $V = A \cup B$.
 - $cut(A, B)$ – sum of links between vertices in A and B .
 - $assoc(A, V)$ – sum of links connected to any vertex in A .
- Segmentation found by solving a generalized eigenvalue problem.



$$\begin{aligned} cut(A, B) &= 0.8 \\ assoc(A, V) &= 9.8 \\ assoc(B, V) &= 17.8 \\ Ncut(A, B) &= 0.1265 \end{aligned}$$

- What cost functions does an energy formulation for segmentation often include?

Semantic segmentation with CRF

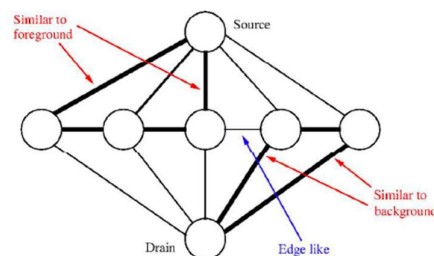
- Idea: Set up the problem as a energy (cost) minimization problem.
- Find the combination of labels that minimizes the cost

$$E = \sum_x \psi_x(l_x) + \sum_{x,y \in N_x} \psi_{x,y}(l_x, l_y)$$

- Cost per node $\psi_x(l_x)$
 - Low cost if pixel has a colour similar to model l_x , high otherwise.
- Cost per link $\psi_{x,y}(l_x, l_y)$
 - Normally $\psi_{x,y}(l_x, l_y) = 0$, if $x = y$.
 - Low cost if pixels have different colours (edge)
 - High cost if pixels have similar colours (smooth surface)
- Result of energy minimization: assign pixels to most similar models, while aligning borders of segments to edges.

- What is the purpose of graph cuts for segmentation?

Graph cuts with 5 pixels



- Graph cuts can be used to minimize the energy E for $|L| = 2$.
- Add links to Source (foreground) and Drain (background) with costs given by $\psi_x(l_x)$, and links with costs $\psi_{x,y}(l_x, l_y)$ between neighbours.
- Goal: Find the lowest cost split of the graph into two pieces.

- How does a graph cut work?

Graph cut works by dividing a graph into two sets of vertices, called clusters, such that all edges between the two sets have a minimum cost. This is done by assigning weights to the edges and then using an algorithm to find the minimum cut. The algorithm works by finding a minimum cut that reduces the weight of the edges connecting the two clusters to a minimum. The resulting clusters are then used to determine the graph's clusters and the associated weights.

- What does a morphological opening and closing operation do?

Morphological opening and closing operations are image processing techniques used to remove noise and isolate objects in an image. A morphological opening operation consists of an erosion followed by a dilation, which removes small objects from an image and preserves large objects. A morphological closing operation consists of a dilation followed by an erosion, which removes small holes from an image and preserves large objects. Both operations are useful for image segmentation and can be used to identify objects in an image.

Deep networks for computer vision

- How does a perceptron work?

A perceptron works by taking in inputs, multiplying them by weights, and then passing them through an activation function to produce an output.

- Why do you need an activation function?

An activation function is needed to introduce non-linearity in the network, allowing it to model complex relationships between inputs and outputs.

- What is a loss function?

A loss function provides a measure of how well the model is performing and is used to adjust the weights of the network in order to reduce errors.

- How does backpropagation work in principle?

Backpropagation works by propagating the error of the model backward through the network in order to adjust the weights of the network to reduce the error.

- Why does a CNN have so many fewer parameters than a FCN has?

CNNs have fewer parameters than FCNs because they use convolutional layers to reduce the number of connections between nodes.

- What are the typical layers of a CNN?

Typical layers of a CNN include convolutional layers, pooling layers, and fully-connected layers.

- How do learned image descriptors perform compared to handcrafted ones?

Learned image descriptors generally perform better than handcrafted ones because they are able to learn the relationships between the various features in an image.

- What is the structure of an autoencoder?

The structure of an autoencoder consists of an encoder which compresses the input data into a latent representation and a decoder which reconstructs the data from the latent representation.

- What is the difference between a VAE and a GAN?

A VAE is a generative model which learns a probability distribution of the data and is able to generate new samples from this distribution, while a GAN is a discriminative model which learns to differentiate between real and generated samples.

- What can GANs be used for?

GANs can be used for various tasks such as image generation, image translation, and image super-resolution.

- Mention an innovation that has affected modern networks for segmentation.

An innovation that has affected modern networks for segmentation is the use of fully convolutional networks, which allow for the segmentation of entire images in a single forward pass.

Object recognition

- What is the difference between recognition and classification?

Recognition is the process of identifying an object in an image or video, while classification is the process of assigning a label to an object.

- What makes a good feature space for recognition?

A good feature space for recognition should have distinct features that are invariant to transformations. Invariances such as scale, rotation, view-point, and illumination should be taken into account.

- What kind of invariances do you often want in recognition?

Common invariances that are often desired in recognition are scale, rotation, view-point, and illumination.

- What classes of recognition methods exist and what are their differences?

The main classes of recognition methods are feature-based methods, template-based methods, and learning-based methods. Feature-based methods use a set of features to describe an object and then use a comparison algorithm to match the features to a known object. Template-based methods use a template to represent an object and then search for the template in an image. Learning-based methods use a training set of labeled data to create a model for the object.

- What does a typical feature based method consist of?

A typical feature based method consists of extracting features from an image, computing descriptors from the extracted features, and then comparing the descriptors to known objects.

- What steps does a Bag of Words approach include?

The Bag of Words approach includes extracting local features from an image, constructing a histogram of the features, and then comparing the histogram to known objects.

- What are the dominating methods based on deep learning?

The dominating methods based on deep learning are Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).

- Why is object detection harder than object classification?

Object detection is harder than object classification because it requires the detection of multiple objects in an image and their respective locations.

- What characteristics does a nearest neighbour classifier have?

A nearest neighbour classifier has the characteristic that it classifies an object based on its closest neighbour in the feature space.

- How do you find a decision boundary for Bayesian classification?

To find a decision boundary for Bayesian classification, one must compute the posterior probability of each class given the features of a particular object. The decision boundary is then formed by the class with the highest posterior probability.

Stereo geometry

- How does stereo work in general?

Ability to perceive depth and 3D relationships between objects by using two eyes to gain visual information about an object's size, shape and position in space.

- Why can you get double vision?

You get a double vision when you have points not matching between both cameras(both eyes)

- What is triangulation?

Triangulation with parallel cameras is a technique used to calculate the exact position of a point in 3D space by measuring the angular distance between the point and two cameras. It works by placing two cameras at known positions and orientations in relation to each other, and then measuring the angle between the two cameras and the point. The point's exact position in 3D space can then be calculated from the angle measurements. This technique is used in many applications, such as surveying, robot navigation, and 3D printing.

Assume we have 2 parallel cameras with baseline B and two matching points (x_l, y) and (x_r, y) in respective camera.

Left camera: $x_l = f \cdot X/Z$, $y_l = f \cdot Y/Z$

Right camera: $x_r = f \cdot (X-B)/Z$, $y_r = f \cdot Y/Z$

Difference in x-coordinates is the horizontal disparity $d = x_l - x_r = f \cdot B/Z$

Assuming B and f known, the depth $Z = B \cdot f/d$

- What is the relationship between disparities and depths? SEE ABOVE for formula

The relationship between disparities and depths is that disparities are the differences between the points of view of two cameras, and depths are the actual distances between the cameras and the object in 3D space. Disparities are the measurable differences between the points of view, and depths are the distances that can be calculated from these disparities. The relationship between disparities and depths is an important concept in triangulation with parallel cameras.

- Why does the error in depth increase for larger distances?

$$d = x_l - x_r = f \cdot B/Z$$

$$Z = B \cdot f/d$$

$$dZ/dd \Rightarrow Z \text{ depends on } B \Rightarrow \text{as we increase } B \text{ the error increase}$$

The error in depth increases for larger distances because the disparity between the cameras decreases as the distance increases. Since the disparity is used to calculate the depth of the object, a decrease in disparity results in an increased error in the depth measurement. This is an important consideration when using parallel cameras for triangulation, as it is important to choose a setup that minimizes the distance between the cameras and the object in order to get an accurate depth measurement.

- What are the key concepts of epipolar geometry?

Epipolar geometry is the study of the geometric relations between two views of a three-dimensional scene or object. It is the mathematical basis of stereo vision and is essential for applications such

as 3D reconstruction, motion estimation, and image registration. The key concepts of epipolar geometry include:

1. Epipolar lines: Epipolar lines are the lines that connect a point in one image to the corresponding point in the other.
2. Epipolar planes: Epipolar planes are planes of points in 3D space that project onto the same epipolar line in two different images.
3. Fundamental matrix: The fundamental matrix is a matrix of 9 elements that describes the epipolar geometry between two images. It can be used to calculate the epipolar lines and epipolar planes.
4. Epipolar constraint: The epipolar constraint is a mathematical equation that states that two points in two images that correspond to the same 3D point must lie on the same epipolar line. This constraint is used for stereo matching algorithms.
5. Essential matrix: The essential matrix is a matrix derived from the fundamental matrix that describes the epipolar geometry between two images without taking into account the intrinsic parameters of the cameras.

- What is an essential matrix and how is it used?

An essential matrix is a 3×3 matrix that describes the epipolar geometry between two images of a scene, taken from two different cameras. It encodes the rotation and translation of the cameras relative to each other. It is used in computer vision algorithms, such as structure from motion and visual odometry, to estimate the motion of the camera, and can also be used to calculate the 3D coordinates of points in a scene from two images.

- What might complicate stereo matching?

Some of the things that might complicate stereo matching include:

- Poorly lit environments
- Moving objects in the scene
- Occlusion of one of the cameras
- Varying camera positions
- Reflections or transparent objects
- Objects with similar texture or color
- Objects with non-Lambertian surfaces

- How does a simple stereo matcher work?

A simple stereo matcher works by comparing two images taken from slightly different perspectives and calculating disparity values which indicate the depth of objects in the scene. This is done by looking at corresponding points in the two images and calculating the difference between them. The disparity values can then be used to infer the depth of the objects in the scene.

- What constraints are often used to improve stereo matchers?

Constraints are often used to improve the accuracy of stereo matchers. These constraints can include enforcing smoothness, enforcing left-right consistency, and enforcing the epipolar geometry constraint. These constraints help reduce errors in the disparity values that may be caused by outliers or image noise.

- What parts do a deep network based stereo matcher often contain?

A deep network based stereo matcher often contains several components, such as a feature extraction module, a cost computation module, a disparity estimation module, and a post-processing module. The feature extraction module extracts features from the two input images, the cost computation module compares the features from the two images and calculates cost values, the disparity estimation module estimates the disparity values, and the post-processing module refines the disparity values and applies any constraints.

Motion and optical flow

- Why is motion more complex than stereo?

Motion is similar to stereo matching but more complex because it's in any direction (we cannot use equipolar lines).

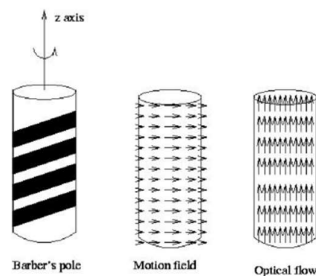
Motion is more complex than stereo because it involves capturing changes in the environment over time. Stereo only captures the current state of the environment, while motion involves tracking changes in the environment over time, such as detecting and tracking objects and movements. Motion also requires more complex algorithms to process the data and extract meaningful information from the video.

- What is a Focus of Expansion?

Focus of Expansion (FOE) is an important concept in motion analysis and optical flow. It is the point in an image where all image features appear to be moving away from. It is often used to determine the direction of motion in a scene, as well as to detect any changes in the scene. FOE is also used to measure the speed of motion in a scene.

- What is Motion field and what is Optical flow?

- Optical flow is the apparent motion of brightness patterns.
- Usually, optical flow corresponds to the motion field, but not always.
- For example, the motion field and optical flow of a rotating barber's pole are different, as illustrated in the figure.



- How do you derive the optical flow constraint?

Consider the intensity of a single scene of a point in time: $I(x(t), y(t), t)$

Compute the derivative to see how change in time: $dl/dt = I_x * dx/dt + I_y * dy/dt + I_t$

Hp: brightness constancy assumption: $dl/dt = 0$

$$\Rightarrow I_x * u + I_y * v + I_t = 0 \quad (\text{Optical flow constraint equation})$$

- What is a Second moment matrix?

A second moment matrix for the Lucas & Kanade algorithm is a matrix that stores information about the second derivatives of the image. This matrix is used to calculate the components of the optical flow vector at each pixel in the image. This vector can be used to identify areas of motion in the image and measure the magnitude of the motion.

- How does Lukas & Kanade cope with large optical flow?

The Lucas & Kanade algorithm is able to cope with large optical flow by using a pyramid approach. This approach involves breaking the image down into smaller and smaller images and then computing the optical flow vector at each level. This approach allows the algorithm to better handle the large amounts of movement in the image and reduces the amount of computation needed to calculate the optical flow vector.

- What are the characteristics of Horn & Schunck?

The Horn & Schunck algorithm is an iterative method used to solve optical flow equations. It was developed by Brian Horn and Barbara Schunck in 1981 and is widely used in computer vision applications. The algorithm is based on an energy minimization technique that uses a combination of spatial and temporal derivatives of the image to compute optical flow. It is an efficient and robust method that is able to handle a variety of motion types, including translation, rotation, expansion, and contraction. The algorithm is also able to handle occlusions, and it is relatively insensitive to brightness changes.

- How do modern methods avoid problems around the borders of objects?

Modern methods use techniques such as image segmentation and object recognition to identify and separate objects in an image. Image segmentation algorithms split an image into a set of non-overlapping regions, while object recognition algorithms use features such as shape and color to identify and classify objects in an image. These techniques help to eliminate problems around the borders of objects by accurately detecting the boundaries of objects and separating them from their surroundings.