I Capturing Light (retina, HSV, WB, SSD, NCC)

- Two types of light-sensitive receptors: cones (colorful, day) and rods (grayscale, night)
- HSV: consider colors as normal dist. over wavelengths: hue=mean, saturation=variance, value=area
- White balance: colors are adjusted to make a white object (e.g. paper or wall) appear white and not a shade of any other color
- Sum of Squared Differences (SSD):

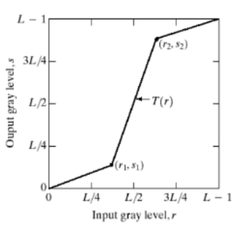
$$ssd(u,v) = \sum_{(x,y)\in N} \left[I(u+x, v+y) - P(x,y)\right]^2$$

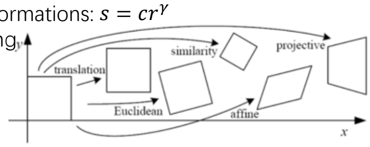
- Normalized Correlation (NCC):

$$ncc(u,v) = \frac{\sum_{(x,y)\in N} \left[I(u+x,v+y) - \bar{I}\right]\left[P(x,y) - \bar{P}\right]}{\sqrt{\sum_{(x,y)\in N} \left[I(u+x,v+y) - \bar{I}\right]^2 \sum_{(x,y)\in N} \left[P(x,y) - \bar{P}\right]^2}}$$

II Image Processing I (Pixel level proc., freq.)

- Image: $f(x,y) = [r(x,y), g(x,y), b(x,y)]$, each: $\mathbb{R}^2 \to \mathbb{R}$
- $f(x,y) = \text{reflectance}(x,y) \cdot \text{illumination}(x,y)$
- Grayscale to grayscale enhancement:
  - Power-law transformations: $s = cr^{\gamma}$
  - Contrast Stretching





Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\left[\, I \mid t \,\right]_{2\times 3}$	2	orientation $+\cdots$	
rigid (Euclidean)	$\left[\, R \mid t \,\right]_{2\times 3}$	3	lengths $+\cdots$	
similarity	$\left[\, sR \mid t \,\right]_{2\times 3}$	4	angles $+\cdots$	
affine	$\left[\, A \,\right]_{2\times 3}$	6	parallelism $+\cdots$	
projective	$\left[\, \tilde{H} \,\right]_{3\times 3}$	8	straight lines	

- Image histogram: #pixels of given grayscale
- Histogram equalization: stretch CDF

Frequencies

- Aliasing: distortion/artifact when signal sampled at too low rate
- Solution: lowpass filters (remove high freqs, leaving only safe low freqs); choose lowest freq in reconstruction (disambiguate)

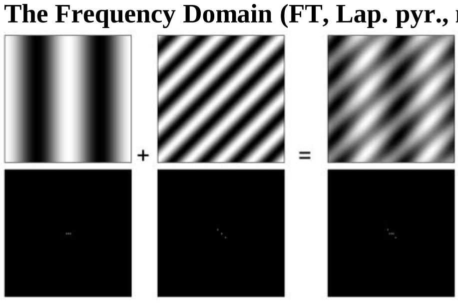
III Image Processing II (Conv, Gaussian, deriva's, DoG)

- Cr's-corre'n: $G = H \otimes F$; $G[i,j] = \sum_{u=-k}^{k}\sum_{v=-k}^{k} H[u,v]F[i+u, j+v]$
- Conv: $G = H * F$; $G[i,j] = \sum_{u=-k}^{k}\sum_{v=-k}^{k} H[u,v]F[i-u, j-v]$
- Conv is **commutative** and **associative**, like multiplication
- Gaussian(σ)$*$Gaussian(σ)=Gaussian($\sqrt{2}\sigma$)
- Subsampling: do Gaussian filtering before subsampling

stack preserves size!

- Gaussian pyramid: repr. $N \times N$ img as $N \times N, \frac{N}{2} \times \frac{N}{2}, \frac{N}{4} \times \frac{N}{4}, \ldots$ imgs
- Construction process: filter, subsample, filter, subsample, ⋯; use: improve search over trans/scale
- Partial derivatives: $\frac{\partial f}{\partial x}$ - lots of vertical lines; $\frac{\partial f}{\partial y}$ - horz lines
- Edge strength: $\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$; grad direc'n: $\theta = \tan^{-1}\frac{\frac{\partial f}{\partial x}}{\frac{\partial f}{\partial y}}$
- Effect of noise: derivative very intense, cannot reveal real edges
- Derivative theorem of convolution: $\frac{\partial}{\partial x}(h * f) = \left(\frac{\partial}{\partial x}h\right) * f$
- Derivative of Gaussian filter: Gaussian $*$ $[1, -1]$
- Shape of conv output: 'full', 'same', 'valid' – orig' size + [ksize-1, 0, -ksize+1]. e.g. (33, 33) $*$ (11, 11): 43; 33; 23.
- Near edge: clip (black), wrap (circular), copy edge, reflect

IV The Frequency Domain (FT, Lap. pyr., matching)



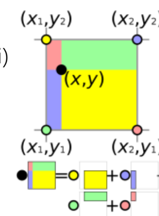
In Fourier magnitude spectrum:

- offset=freq, center=avg val of img, symmetrical about origin
- rot. in img=rot. in FT; horz/vert lines: edge effects, reduced by windowing
- conv theorem: $F[g * h] = F[g]F[h]$; conv → product by Fourier transf'm
- Shapening: img+α ×details. $f' = f * [(1+\alpha)\text{id} - \alpha g]$, g: Gaussian filter
- **Laplacian Pyramid**: iterative Gaussian & subtraction
- Alpha blend'n': window=size of largest prominent feature, <= smallest p.f.
- Lossy Image Compression (JPEG): split into blocks; compute DCT coefficients; coarsely quantize; encode with Huffman encodings
- Smoothing vs. derivative filters: sum to 1/0
- Matching with filters: 1. zero-mean filter (fastest, not a great matcher); 2. SSD: next fastest, sensitive to overall intensity; NCC: slowest, best results
- Denoising (salt-and-pepper noise ↓): median filter > Gaussian filter

V Image Transformations ($\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$)

- Scaling: $\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$; 2D rotation: $\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$($1\,\frac{\text{deg o}'}{\text{fr}'\text{d}'\text{m}}$); 2D shear: $\begin{bmatrix} 1 & s_x \\ s_y & 1 \end{bmatrix}$
- Homogeneous coords: $(x,y,w) \equiv \left(\frac{x}{w}, \frac{y}{w}\right)$; $(x,y,0) \equiv \text{inf}$; $(0,0,0)$ invalid
- Affine transformation $\begin{bmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{bmatrix}$: parallel lines→parallel; ratio preserved.

  6 d.o.f.: translation x, y; scaling x, y; rotation θ; shearing x. need 2 corrsp.
- Projective transformation: 8 d.o.f, 4 corrsp. (right-bottom of matrix is 1)
- All transformations preserve straight lines
- Bilinear interpolation (see the figure on the right)
- Delaunay triangulation: maximize smallest angle (Voronoi)



VI Data-driven Methods: Faces

Computing means. Two requirements:

- 1. Alignment of objects; 2. objects must span a subspace

Deviation from the mean.

Extrapolating faces

VII The Camera

- Camera captures pencils of rays; Aperture: Center of Projection (COP); img formed on img plane; effective focal len'th d=dist. COP to img plane
- Put img plane in front of COP → projection plane; camera look down –z
- $(x,y,z) \xrightarrow{\text{projection}} \left(-d\frac{x}{z}, -d\frac{y}{z}, -d\right)$ (d is f in terms of focal length)
- Aperture: ×too small (less light get through; diffraction)
- Depth of field: max-min depth clear; small ap't're → larger depth of f'd
- F-number: focal length / aperture diameter
- Field of view (FOV): $\phi \uparrow = \tan^{-1}\frac{d}{2f\downarrow}$, where d is radius of camera retina
- Exposure: small aperture needs longer shutter speed (longer time)

VIII Homographies and Panoramas

- Foreshortening: line not // img plane, appears shorter
- The Plenoptic Function: $P(\theta, \phi, \lambda, t, V_x, V_y, V_z)$ θ, ϕ: angles; λ: wavelength; t: time; V_x, V_y, V_z: view point; returns: intensity of light
- Spherical panorama: all light rays through a point is a panorama $P(\phi, \theta)$
- Homography: projective mapping between any to projection planes with the same center of projection. $\boldsymbol{p}' = H\boldsymbol{p}$. Solved by least squares.
- Mosaics: stitching images together. imgs are reproj' onto common plane

IX More Mosaics

- Changing camera center: planar scene → still homography
- Alpha blending: center seam (portion: dist to border)
- 3D → 2D Perspective Projection: $(x,y,z) \to \left(-f\frac{x}{z}, -f\frac{y}{z}\right)$; $\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} f & 0 & u_c \\ 0 & f & v_c \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$, where X_c, Y_c, Z_c are 3D coordinates
- 3D Rotation Model: Project from img to 3D ray; Rotate ray by camera motion (R_{01} matrix); Project back into new img. $H = K_0 R_{01} K_1^{-1}$
- Cylindrical projection: $(\sin\theta, h, \cos\theta) = (\hat{x}, \hat{y}, \hat{z}) = \frac{1}{\sqrt{X^2+Z^2}}(X, Y, Z)$
- Spherical p.: $(\sin\theta\cos\phi, \sin\phi, \cos\theta\cos\phi) = (\hat{x}, \hat{y}, \hat{z}) = \frac{1}{\sqrt{X^2+Y^2+Z^2}}(X \ldots)$

X Automatic Image Alignment

- Feature-based alignment: 1. feature detection (2 images respectively); 2. feature matching; 3. compute image transformation

Harris Corner detector: 1. compute Gaussian derivatives at each pixel; 2. compute 2nd moment matrix M in a Gaussian window around each pixel; 3. compute corner response function $R = \frac{\det M}{\operatorname{trace} M}$; 4. threshold R; 5. Non-Maximum Suppression
Properties: rot'n invariance; partial inv' to affine intensity change; variant to image scaling (need to consider regions of diff sizes around a pt)
Adaptive NMS: sort pts by NMS radius (preserve pts w/ response>thre')
Feature Descriptor – Multi-Scale Oriented Patches: Orientation (θ)=blurred grad; rot. inv' frame=scale-space position (x, y, s)+orien' θ
MOPS descriptor vector: 8×8 oriented patch (sampled at 5×scale);
Bias/gain normalization: $I' = (I - \mu)/\sigma$
Outlier rejection – Lowe's trick: SSD of closest match/SSD of 2nd closest
– Symmetry: x's nearest neighbor is y, y's NN is x
RAndom SAmple Consensus: 1. select 4 feature pairs (at random); 2. compute homography H; 3. keep the most inliers where $\operatorname{dist}(p', Hp) < \varepsilon$; 4. recompute homography H (by least squares). loop.

XI 3D Vision

- To get depth, we need at least two camera centers
- Multi-view geometry problems: 1. 3D structure – 3D point coordinates? 2. Correspondence: given a pt in one img, where are pts in other imgs? 3. Motion: relative camera params between imgs?

Coordinate frames + transforms:
world coords–[Extrinsics($R, \boldsymbol{t}$)]→camera coords–[Intrinsics(K)]→img c'ds

Intrinsics: $\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$(camera coords)→img plane coords$\begin{bmatrix} x_i \\ y_i \end{bmatrix}$→img coords$\begin{bmatrix} x_c \\ y_c \end{bmatrix}$

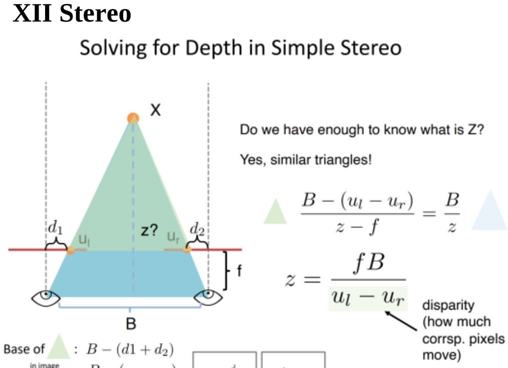
$(x, y)_i = f\frac{(x,y)_c}{z_c}$; $(u, v)_i = m_{(x,y)}(x, y)_i + o_{(x,y)}$; Let $f_{(x,y)} = f \cdot m_{(x,y)}$

Intrinsic matrix: $\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_x & s & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$ m: pixel per unit length, i.e. density s: skew, usually 0

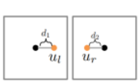
Extrinsic matrix: $\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{3\times3} & \boldsymbol{t}_{3\times1} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$ R: rotation; $\boldsymbol{t}$: translation

Intrinsic: 4+1=5 d.o.f.; Extrinsic: 3+3=6 d.o.f.
F'nd'ment'l Scale Ambig'ty: scaling world & camera doesn't change img
Camera to world coords: 1-to-1 corresp.; img to camera: pixel to ray!
Camera calibration: Take a picture of an object with known 3D geometry (coords) → identify corresp. Solve for world to img coords matrix (plug depth in!!); QR decomposition to get K & R

XII Stereo



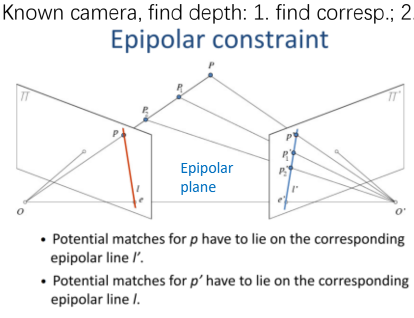
Depth is inversely proportional to disparity. Parallax: Change in position from different view point



Epipolar line: point in another image must be on a line. Use a window to search on that line. win size: large enough to have e'gh intensity variat'n; small e'gh to contain only px w/ similar disparity

XIII Epipolar

General case: non-parallel axes; know camera intrinsics
Known camera, find depth: 1. find corresp.; 2. triangulate.



Baseline: OO'
Epipoles: e, e'
Let $\overline{OP} = \boldsymbol{x}$, $\overline{O'P'} = \boldsymbol{x}'$, then $\boldsymbol{x}'^T E\boldsymbol{x} = 0$; $E = [\boldsymbol{t}_\times]R$;
In O' coord sys, $\boldsymbol{x}' \cdot (\boldsymbol{t} \times (R\boldsymbol{x} + \boldsymbol{t})) = 0$, where $R\boldsymbol{x} + \boldsymbol{t}$ is $\boldsymbol{x}$ in O' coord sys.
E is *essential matrix*, rank 2, 5 d.o.f. coeffs of $l' = E\boldsymbol{x}$; $l = E^T\boldsymbol{x}'$; $E\boldsymbol{e} = E^T\boldsymbol{e}' = 0$.

Uncalibrated case: Recall we set $\boldsymbol{x} = K^{-1}\hat{\boldsymbol{x}}, \boldsymbol{x}' = K^{-1}\hat{\boldsymbol{x}}'$, where $\hat{\boldsymbol{x}} = [u, v, 1]^T$; now we don't know K, K'. So we plug it in, then we get fundamental matrix $F = \left(K'^{-1}\right)^T EK^{-1}$, with rank 2, 5 d.o.f, $l' = F\hat{\boldsymbol{x}}$ (other properties like E).
Eight-point algorithm: use 8 or more matches to solve for F (least squares); take SVD of F and throw out the smallest singular value to make F singular
Triangulation: know corresp. $\boldsymbol{x} \leftrightarrow \boldsymbol{x}'$, K, K', R, $\boldsymbol{t}$; let 3D point be X, then $\boldsymbol{x} = KX$, $\boldsymbol{x}' = K'(RX + T)$, solved by least squares.

XIV Structure-from-Motion (SfM) & Multi-View Stereo (MVS)

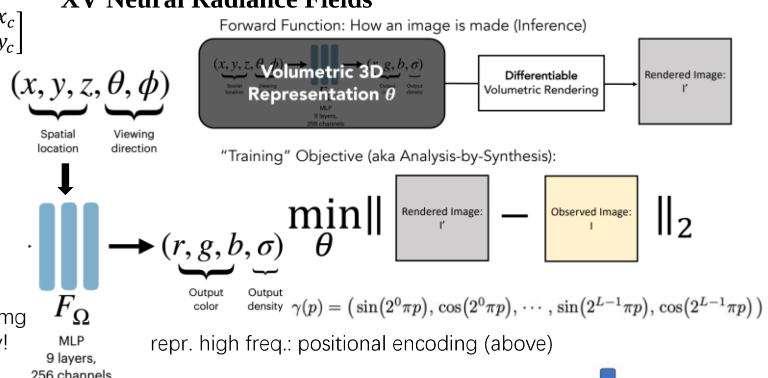
Structure from Motion: many image → where taken & 3D model
Input: images with pts in correspondence; Output: 3D coords & motion(R, t)
Objective function: minimize reprojection error
Correspondence: Feature detection (SIFT) & matching (between each pair, refine with RANSAC to estimate fundamental matrix, link up to form whole)

- Minimize sum of squared reprojection errors:

$$g(\mathbf{X}, \mathbf{R}, \mathbf{T}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \underbrace{w_{ij}}_{\substack{\text{indicator variable:} \\ \text{is point } i \text{ visible in image } j\text{?}}} \cdot \left\| \underbrace{\mathbf{P}(\overbrace{\mathbf{x}_i}^{\boldsymbol{x}\text{: 3D coords}}, \mathbf{R}_j, \mathbf{t}_j)}_{\substack{\text{predicted} \\ \text{image location}}} - \underbrace{\begin{bmatrix} u_{i,j} \\ v_{i,j} \end{bmatrix}}_{\substack{\text{observed} \\ \text{image location}}} \right\|^2$$

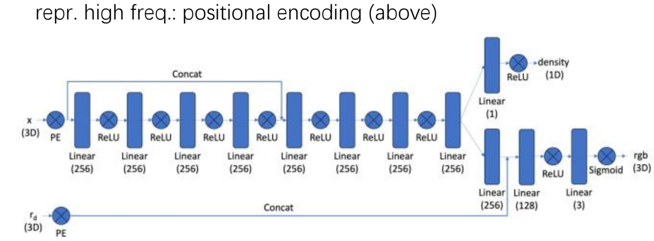
Start with a small seed reconstruction, and grow
Incremental SfM: pick strong init pairs, init model with 2-frame SfM, while there are connected images: pick img with most known 3D pts, estimate pose of camera, triangulate any new points run bundle adjustment
Multi-view Stereo: know lots of calibrated images, give 3D model. match patches, give depth map. combine many depth maps.

XV Neural Radiance Fields



repr. high freq.: positional encoding (above)

Network Structure:



Alpha blending: $I = \sum_{i=1}^{D} C_i\alpha_i \prod_{j=1}^{i-1}(1 - \alpha_j)$ layer closest to camera is ly. 1
Camera, pixel to ray: Suppose $\boldsymbol{x}_c = \begin{bmatrix} R & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix} \boldsymbol{x}_w$, then camera center in world coords: $\boldsymbol{r}_o = -R^{-1}\boldsymbol{t}$. Convert pixel coords to camera coords with depth=1, then convert to world coords $\boldsymbol{x}_w$; finally $\boldsymbol{r}_d = \frac{\boldsymbol{x}_w - \boldsymbol{r}_o}{\|\boldsymbol{x}_w - \boldsymbol{r}_o\|_2}$. Expected color=$\sum_{i=1}^{n} \exp\left(-\sum_{j=1}^{i-1} \sigma_j\delta_j\right) \boldsymbol{c}_i(1 - \exp(-\sigma_i\delta_i))$, $\alpha_i = (1 - \exp(-\sigma_i\delta_i))$
We sample points along the rays.

XVI Texture: Statistical Models (Histograms)

Texture: spatially repeating patterns. Deal with filter responses.
1. simple statistics (mean, std); 2. marginal histograms; 3. hist. of joint resps.

XVII Sequence Models for words and pixels

Image = seq. of visual words → generated by LLMs

XVIII Generating Images from Noise

CNN for artistic style transfer; diffusion: keep adding noise

Complementation

1. Derivative filters: $D_x = [1, -1], D_y = [1, -1]^T$.
2. Outer product between two 1D Gaussians = 2D Gaussian.