CV

Weifan Jiang

12/4/2019

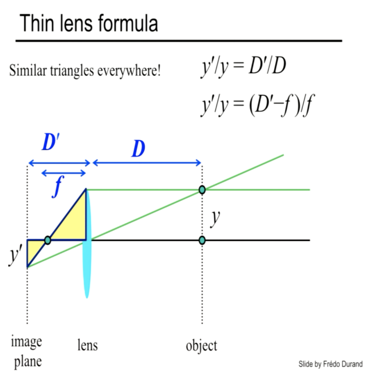
Optics and image formation

Projective Geometry: length and angle is lost, straight lines are preserved.

False Perspective: looks deeper on image

Vanishing point: all parallel lines converging to a vanishing point except lines parallel to image plane.

Thin Lens Formula: $\frac{1}{D'} + \frac{1}{D} = \frac{1}{f}$ (any points satisfying TLF is in focus)



Chromatic aberration: color fringing since lens has different wavelengths for different indices

Spherical aberration: rays farther from optical axis focus closer

Vignetting: outskirt of photo not bright

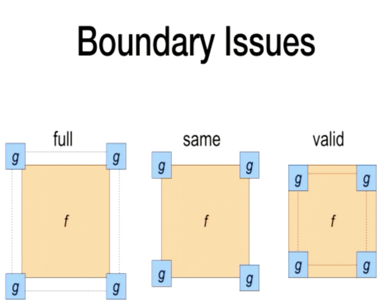
Demosaicing: estimate missing values from neighboring elements

Image Processing

Linear Filtering:

- filter(im, f1 + f2) = filter(im, f1) + filter(im, f2)
- C * filter(im, f1) = filter(im, C * f1)

Convolution: f is an image, g is the kernel, $Conv(f, g)[x, y] = \sum_{i,j} f[x - i, y - j]g[i, j]$



Convolution Properties:

- Commutative: F * H = H * F
- Associative: (F * H) * G = F * (H * G)
- Distributive: (F * G) + (H * G) = (F + H) * G

Complexity of $n \times n$ kernel on $m \times m$ image: $O(n^2m^2)$.

Separable kernel: kernel = product of two subkernels, use associativity ($O(n^2m)$)

Median Filter: select median value in each kernel window, robust against outlier, non-linear

Edge Detection

Image derivative:

- First derivatives by convolution: $d/dx = [-1, 1]$, $d/dy = [-1, 1]^T$

Edge: biggest change, derivative has maximum magnitude, or second derivative is 0.

Laplacian Filter: sum of second order partial derivatives

Gabor Filter: cosine multiplied by Gaussian

Image resizing: sample (alias: causes different signals to become indistinguishable when sampled)

Noise: derivative is high everywhere, must smooth before computing derivative

Canny Edge detector:

1. Filter image with x, y derivatives of gaussian
2. Find magnitude and orientation of gradient
3. Non maximum suppression
4. Hysteresis Thresholding (edge linking): two thresholds to determine weak and strong edges, and link strong edge into weak edges.

Effect of σ (gaussian kernel spread/size): larger σ detects large edges, small σ detects fine features.

Hough Transform:

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

We can transfer samples to parameter space (for lines, transfer them to a point on the slope-intercept space). Not robust to noise - can increase bin size.

Fourier Transform

Function: sum of sine and cosine waves

Convolution in time space = Multiplication in Frequency Space ($g * h = F^{-1}(F(g) \cdot F(h))$)

Hybrid images: take high frequencies of one image, and low frequencies of the other, and combine them.

Geometry

Seam Carving: preserve the most interesting content by removing pixels with low gradient energy. Removing an irregular shaped path from top to bottom (left to right).

1. Computer energy of each pixel of image f: $\sqrt{(\frac{\delta f}{\delta x})^2 + (\frac{\delta f}{\delta y})^2}$
2. Compute each path's cost with dynamic programming: $M(i,j) = Energy(i,j) + \min(M(i-1,j-1), M(i-1,j), M(i-1,j+1))$, then backtrack from the end of path (pixel with least M value in last row).

Greedy approach: Choose the minimim energy option at each step (not optimal)

Failure case of seam carving: when what's interesting does not correspond to high gradient.

Corner detection (idea): moving in all directions produce larger SSD error ("matchable"):

$cornerness(x_0, y_0) = \min E_{x_0,y_0}(u,v), u^2+v^2 = 1$ where $E_{x_0,y_0}(u,v) = \sum_{x,y \in W(x_0,y_0)} [I(x+u,y+v) - I(x,y)]^2$

Implementation:

- $I(x+u, y+v) \approx I + I_x u + I_y v$ where $I_x = \frac{\delta I(x,y)}{\delta x}$, so $E(u,v) \approx [u,v]A[u,v]^T$ where $A = \sum_{(x,y \in W)} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$.
- Cornerness: $C = det(A) - \alpha trace(A)^2$ (trace: min eigenvalue + max eigenvalue)

Harris Corner Detector

- Rotation invariance: cornerness is robust to in-plane rotation (eigenvalue constant, eigenvector not)
- Scale invariance: (eigenvector constant, eigenvalue change) corner location is not co-variant to scaling.

SIFT: scale invariant feature transform

1. Normalize the rotation and scale of a patch (find the dominant gradient and rotate the image to make the dominant gradient in the same place)
2. Closeness of two matchable points: euclidean distance of points in feature space
3. For each patch after normalization: break each patch into grids, compute gradient histograms fro each grid, and concatenate them to a feature scriptor.
4. Rescale the descriptor to have unit norm, then clip high values (0.2) and devide by norm again.

HOG: compute SIFT descriptors on a grid same as a cell (reoptimize hyperparameters).

Feature Matching:

Nearest Neighbor Distance Ratio: compare distance of closest neighbor (NN1) and second closest neighbor (NN2)

- Ratio approach 0: confident match; ratio approach 1: match too close
- Output matches in order of confidence

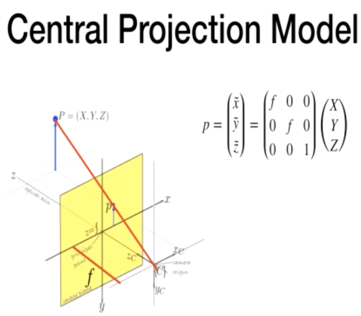
Homography:

Homogenous coordinate: $(x/z, y/z)_{Cartesian} \rightarrow (x,y,z)_H$

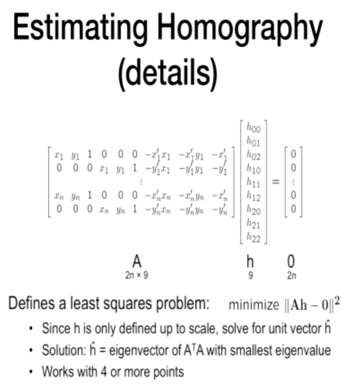
In Homogenous space: Point equation of a line: $l^T p = 0$; cross product of two points = the line goes through both points; cross product of two lines: intersection of the lines

Central Projection Model: model used to illustrate cameras

Scale Invariance



Estimate Homography:



Warping: take a pixel location in the warped image, consider where to grab the point from the pre-transform image.

How to get matches to estimate homography:

RANSAC (Random Sample Consensus):

- Sample numberr of points required to fit the model
- Solve for model parameters
- Score by fraction of inliners within a threshold

Object, Scene, Action Recognition

Neural Networks

In each layer of NeuroNet: $x_{i+1} = f(W_i x_i + b)$.

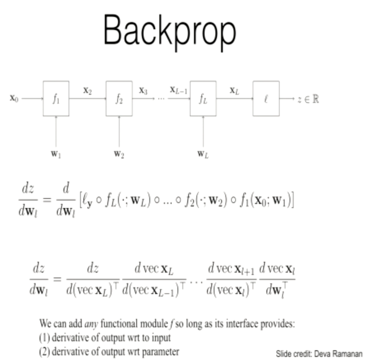
Loss Function: $\min_\theta \sum_i L(f(x_i, \theta), y_i)$.

Gradient Descent: $\theta_{t+1} = \theta_t + \alpha \frac{dL}{d\theta}$; (with momentum) $z_{t+1} = \beta z_t + \frac{dL}{d\theta_t}$, $\theta_{t+1} = \theta_t + \alpha z_{t+1}$. (Regularization to avoid overcomplex models)

BackProp: cache intermediate partial derivatives

Choices for f:

ReLU (Rectified Linear): $ReLU(a) = max(a, 0)$, $ReLU'(a) = \begin{cases} 1, a > 0 \\ 0, a \leq 0 \end{cases}$.



Leaky ReLU: $\alpha min(a, 0)$ when $a < 0$. Solves the problem that ReLU sometimes have 0 gradient, and α can be learned.

Convolutional Networks: stack together convolution and pooling (average and subsample).

Max Pooling: with window size and stride (step size).

Breaking Neural Networks:

- adversal images: find minimal change to input data that maximize the Loss Function ($\max_\Delta L(f(x + \Delta), y) - \lambda||\Delta||_2^2$).
- fix adversal attacks: generate adversal examples while training, and use the new examples for training.

Guided Backprop: zero-out negative gradient while backproping, to ignore stuff that the neuron does not detect.

Training a deeper network: both training and testing error went up, deeper network underfits…overdeep plain nets have higher training error

Deep Residuaal Network: add a skip option, so $H(x) = F(x) + x$, easy to represent Identity change. (Deeper ResNet has low train and test error.)

Object, Scene, Action Recognition

Supervised object recognition

Nearest Neighbot

Deformable part models: model encodes local appearance + geometry.

Semantic segmentation using convolutional networks: Classification problem after downsampling convolved picture.

- Image Pyramid: shrink image to different dimensions, run through CNN
- Skip Connections (ResNet)
- Dilation: subsampling to allow convolution layers capture more informations. (Kernal has gaps)

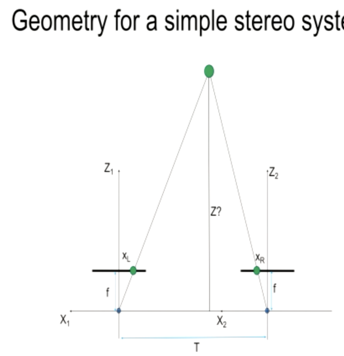
Classical Categorizaation: group objects by common properties

Prototype Theory: an object will be classified as instance of category if similar enough to a prototype

The perception of function: affordances (direct perception) that we think about actions instead of assigning name to objects. (Gibson)

- Limitations: similar structured objects can have different functionalities (mailbox/trashcan)

Stereo



$Z = f\frac{T}{X_R - X_L}$

Epipolar Geometry: which window matches best along horizontal axis

Stereo Matching: match same points in two images when camera shifted horizontally

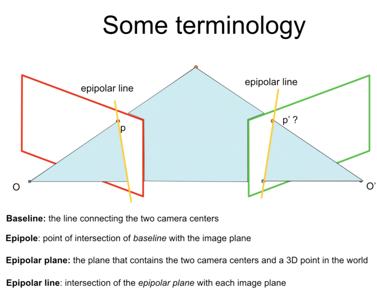
Good Stereo Correspondence: similar intensities, neighboring pixels move about the same amount

- Greedy: window search, do not consider smoothness
- Stereo as energy minimization (better), $E_d(d) = \sum_{x,y \in I}$ SSD between $I(x,y)$ and $I(x+d,y)$ (match cost).
  $E_s = \sum_{p,q \in \epsilon} V(d_p, d_q)$, choice of V: potts model (1 if depth different)
  $E(d) = E_d(d) + E_s(d)$.

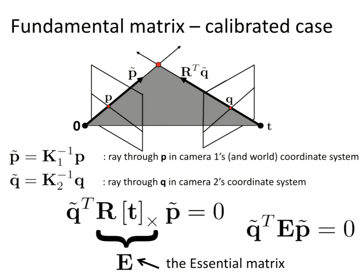
Implementation: DP, $D(x,y,i)$ is min cost solution such that $d(x,y) = i$.

$D(x,y,i) = C(x,y,i) + min_{j=0,\ldots,L} D(x-1,y,j) + \lambda|i-j|$.

Stereo with translation and rotation



Can code all geometries as $p^T E p' = 0$, E: essential matrix, p and p' in homegenous space.



$F = K_2^{-T}[t]_\times K_1^{-1}$: Fundemental matrix

Estimate Fundemental Matrix: $x'^T F x = 0$, one matched pair gives one constraint, F has 8 variables so need 8 points.

8-point algorithm

$$\begin{bmatrix} u_1u_1' & v_1u_1' & u_1' & u_1v_1' & v_1v_1' & v_1' & u_1 & v_1 & 1 \\ u_2u_2' & v_2u_2' & u_2' & u_2v_2' & v_2v_2' & v_2' & u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_nu_n' & v_nu_n' & u_n' & u_nv_n' & v_nv_n' & v_n' & u_n & v_n & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0$$

We want to solve the linear system: $Af = 0$

The solution f is the eigenvector corresponding to the smallest eigenvalue of $A^T A$

Problem with 8-point algorithm: orders of magnitude difference between columns in A to make Least Squares yield poor results.

Motion and Video

Optical Flow: estimate pixel movement from image H to I

brightness consistency: $H(x, y) = I(x + u, y + v)$

small movement: $I(x + u, y + v) = I(x, y) + (dI/dx)u + (dI/dy)v$

brightness consistency constraint equation: $I_x u + I_y v + I_t = 0$, $I_t = I(x, y) - H(x, y)$.

Spartial Coherent Constraint: pretend pixel's neighbor has same u, v (use least squares for best fit: $A^T Ax = A^T b$)

Problem (aliasing): estimated flow less than actual flow, solution: downsample

Learning from video:

$$\hat{c}_j = \sum_i A_{ij} c_i \quad \text{where } A_{ij} = \frac{\exp\left(f_i^T f_j\right)}{\sum_k \exp\left(f_k^T f_j\right)}$$

Material Properties

- Ideal Diffusion: Lambertian (scatter to all directions)
- Ideal Specular: mirror
- Directional diffuse: partial