CS543: COMPUTER VISION

INDEX

1. Camera Lens
2. Interpolation [upsample / downsample]
3. Image Transformations
   - Affine
   - Projective
   - Inverse Warping
4. Image Filtering
   - filters – Pattern detectors
   - convolution
   - gaussian filter (low Pass Filter)
   - Edge filters (high Pass Filter)
5. Fourier Analysis

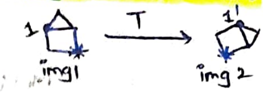
   space $\rightleftharpoons$ sinusoids

6. Edge Detection | Corner Detection
   - Partial derivatives of images via convolution.
   - Canny Edge Detector
   - Harris Corner Detector
7. Blobs / interest Points
   - well defined, rich in scene invariant to scale, rotate, light
   - n_σ : gaussian filter → smoothens
   - $\nabla(n_\sigma)$: der. of n_σ → get edges
   - $\nabla^2(n_\sigma)$: 2nd der of n_σ → get edges.
   - SIFT Detector.
8. Fitting
   - represent features [edge, blob ..] w/ a parametric model.
   - Algos:
     - → Robust Least Squares
     - → RANSAC.
9. Hough Transform

   img space $\rightleftharpoons$ Param space $\begin{bmatrix} m, c \\ \theta, \rho \end{bmatrix}$ (Polar rep.)

   point $\rightleftharpoons$ line

10. Image Alignment



Solve for T, given img1, img2

Logic
1. Extract features → SIFT
2. Putative matches → (1,1'), (2,2')
3. Solve for T w/ point pairs (②) → RANSAC.

11. Camera, Light & Shading
   - Pinhole camera
   - lamberts law, albedo effect
   - Horizons, Vanishing Point (VP)
   - Camera Matrix

image coord ← world coord

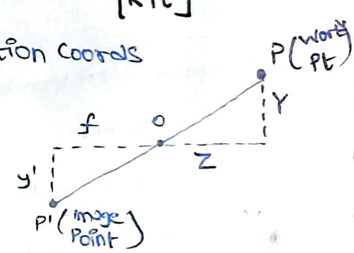
$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} as & k & c_x \\ 0 & s & c_y \\ 0 & 0 & 1/f \end{bmatrix} \underbrace{\begin{bmatrix} C_p \\ [I\,|\,0] \end{bmatrix}}_{3\times4} \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} \times \begin{bmatrix} X \\ Y \\ Z \\ T \end{bmatrix}$$

image coordinates; camera Intrinsics (K); canonical Projection; camera Extrinsics [R|t]; 3D world Pt.

   - Derived Projection Coords

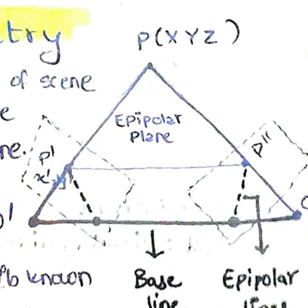
$P'_{x',y'} \left(\frac{fX}{Z}, \frac{fY}{Z} \right)$

$y' \propto \frac{1}{Z}$



12. Single View Modelling
   - camera calibration w/ VP
     - solve for (K), [R|t]
       - vars: (f, c_x, c_y) (R, t)
     - 3 orthogonal VPs (≥ 2) VP
     - $v_i \cong K[R|t]\begin{pmatrix} e_i \\ 0 \end{pmatrix}$
       - K: $v_i^T K^{-T} K^{-1} v_j = 0$
       - R: $r_i = K^{-1} v_i$

13. Epipolar Geometry
   - A pt (P') in one img of scene will line on Epipolar line on another img of scene.
   - Solve:
     - M1) Essential Matrix
       $x'^T . E . x = 0$ — calib known
     - M2) Fundamental Matrix — calib unknown.



14. Structure For Motion [SFM]

Given, multiple imgs of scene, w/ known/ unknown camera solve:

1. camera locations → Motion (M)
2. world geometry →
3. triangulation → (Find world coords.) Structure (S) (sparse-geometry)

Modern Incremental SFM Algo:

1. Detect features in imgs. → SIFT
2. Feature Matching → SuperGlue
3. Generate 2D Tracks from matches →
4. SFM model from tracks → S matrix, M matrix
5. SFM model refinement w/ new views, bundle adjust.

15. Multi View Stereo

- Generate dense 3D scene repN. using multiple images.

Algo:

1. Compute correspondences (SFM ① ②)
2. Plane sweep stereo
   - Est. depth of each Pixel.
   - Cost fn: similarity measures
     - NCC
3. Fuse depth maps → single [3D repN] [Mesh, Pointcloud]

16. NeRF

Use MLP to parameterize Radiance Field Function (f_θ) → quantify radiance (intensity, color) of light from a World Pt. coming to camera.

$$f_\theta : \mathbb{R}^{Lx} \times \mathbb{R}^{Ld} \rightarrow [0,1]^3 \times [0, \infty]$$

$$\gamma(x), \gamma(d) \rightarrow (c, \sigma)$$

(x : 3D pt, d : viewing direction) → (color RGB, volume density)

[θ : Model wts.
γ : positional encoding]

Volume Rendering [3D repN]

- Differentiably render out novel views using radiance function.
- $r(t) = o + td$
  - r: ray
  - o: cam center
  - d: direction
- Pixel color $\hat{c}_\theta(r)$ computed using alpha compositing.

$$\hat{c}_\theta(r) = \int_{t_n}^{t_f} T(t)\,\sigma_\theta(r(t))\, c_\theta(r(t), d)$$

- $T(t)$ = transmittance - Prob. of ray travelling $t_n \rightarrow t_f$ w/o hitting a particle.