CS4243
AY23/24 Sem 2
github.com/jasonqiu212

05. Segmentation

- Goal: Separate image into coherent regions
- Idea: **Clustering** - Group similar data points together
- Challenges: What makes 2 points same/different? Choice of features (e.g. Color, Intensity, Position), Which clustering algorithm?
- **k-Means Clustering** - Iteratively re-assign points to nearest cluster center
  1. Given K, randomly initialize the cluster centers $c_1, \dots, c_K$
  2. For each point p_i, find the closest c_j and put p_i into cluster j
  3. Given points in each cluster, set c_j to be mean of points in cluster j
  4. Repeat, if c_j have changed up to some threshold
  - Pros: Simple, Converges to local min.
  - Cons: Setting K, Sensitive to initial centers (Since k-means converges to local min.), Sensitive to outliers (Can add more clusters), Assumes spherical clusters (Fix with mean-shift)
- **Simple Linear Iterative Clustering (SLIC) Superpixels**
  - **Superpixel** - Group of pixels that share common traits
    * Application: Inputs to other CV algo. since more compact representation with perceptual meaning
  - Num. of pixels: n_{tp}; Target num. of superpixels: n_{sp}
  - Initial width of each superpixel: $s = \sqrt{\frac{n_{tp}}{n_{sp}}}$
  - Features: $z = [r, g, b, x, y]$
  - Color distance: $d_c = ||\langle r_j, g_j, b_j\rangle - \langle r_i, g_i, b_i\rangle||$
  - Spatial distance: $d_s = ||\langle x_j, y_j\rangle - \langle x_i, y_i\rangle||$
  - Scaling factors: d_{cm} and $d_{sm} = s$ set as max. expected values of d_c and d_s respectively
  - $D = \sqrt{(\frac{d_c}{d_{cm}})^2 + (\frac{d_s}{d_{sm}})^2} = \sqrt{d_c^2 + (\frac{d_s}{s})^2 c^2}$
  1. Split img. into grid of size $s \times s$. Set cluster centers as lowest gradient position in 3×3 neighborhood from superpixel center to speed up convergence since initialize on value common to surrounding.
  2. For each cluster center, check distance to all pixels within $2s \times 2s$ neighborhood. Assign pixels to closest checked center.
  3. Update cluster centers using mean and repeat if not converged (Same as k-Means)
  4. Optional: Replace superpixel region by average value to create stained glass effect
  - Modification of k-Means: Not random initialization, Compute pixel's distance only to closest set of cluster centers
  - Can enforce connectivity and use other features too
- **Mean-Shift Clustering** - Find local density maxima in feature space
  - **Attraction basin** - Region in feature space for which all trajectories of centroids lead to same mode
  - **Cluster** - All data points in attraction basin of a mode
  1. For each data point:
     (a) Define window around and get centroid
     (b) Shift window to centroid
     (c) Repeat until window centroid stops moving
  - Segmentation with Mean Shift: Do mean shift and merge pixels in same attraction basin
  - Choosing window size: Trial and error, Sample points and use avg. dist. to knn. (Num. of neighbors needs to be large enough to ensure increase in density)
    * Larger window size $\rightarrow$ Fewer clusters
  - Pros: No assumptions on cluster shape, 1 parameter, Finds variable num. of modes (vs. specified k in k-Means), Robust to outliers
  - Cons: Choosing h, Slow, Scales poorly with feature space dimension
  - Optimizations:
    * After each run of mean shift, assign all points within radius r of end point to same cluster
    * Assign all points within radius $c < r$ of search path to mode $\rightarrow$ More aggressive, less confident

06. Texture

07. Keypoints

08. Descriptors

09. Homography

10. Optical Flow

11. Tracking

12. Deep Learning