

Segmentation = Extract something from image
 Divides an image into meaningful regions or objects (finding group of pixels)
 Accurate segmentation allows us to:
 • counting : number of each type of object
 • measuring : geometric properties, area, perimeter of objects in the images
 • property study : intensity, texture

Types:

- Semantic segmentation:** classification, don't care single object just the category.
 - three cars are colored Blue, but don't know how many.
 - The sky is colored Red.
- Instance segmentation:** focuses only on distinct objects it can't ignore the background
 - Car1 is Blue. Car2 is Green. Car3 is Yellow. The sky is ignored.
- Panoptic segmentation:** combine both, backgrounds get categories, distinct objects get unique IDs
 - Car1 is Blue, Car2 is Green, Car3 is Yellow. The sky is Red.

2 Main properties of intensity values :

Intensity properties

- Discontinuity (Separate):** separate based on abrupt intensity change (edges)
 - Edge-based : Detecting edges that separate region from each other
- Similarity (Group):** group similar pixels (intensity, color, texture)
 - Thresholding : Using pixel intensity
 - Region-based: Grouping similar pixels -> region growing/merge&split

Point Detection

Line detection original image after apply 45 degree mask thresholded image

First Derivative (Gradient): Robert/Sobel/Prewitt

- Apply these mask make normal picture -> Gradient Picture
- Prewitt is simpler, but Sobel have noise suppression because more weight on center
- Each Pixel apply Gx get Rx and Gy get Ry, then $R = \sqrt{Rx^2 + Ry^2}$

Canny Method (1986): using 1st derivative, can detect thin line

- Gaussian smoothing : noise reduction
 - Apply 5x5 Gaussian Filter
- Compute gradient : Sobel
 - Each Pixel apply Gx get Rx and Gy get Ry, then $R = \sqrt{Rx^2 + Ry^2}$
- Non-maximum suppression : make edge thinner
 - edge edge pixel : pixel size 1 pixel -> 1px
- Hysteresis thresholding : connect not edges, 2 steps:
 1. Morphological Thinning, pixel-wise
 - Strong Edge: $> \text{MaxVal} \rightarrow$ เน้นเส้นที่แข็งแกร่ง
 - Weak Edge: $\text{MinVal} < \text{ค่าความเข้ม} < \text{MaxVal} \rightarrow$ ไม่แน่ใจ
 - Non-Edge: ค่าความเข้ม $< \text{MinVal} \rightarrow$ ไม่ใช่เส้นภาพแน่นอน
 2. Edge tracking : only Weak edge that connect with strong edge will remain
 - Weak edge ที่เชื่อมกับ Strong edge ยังคงอยู่

Canny Method

① Gaussian smoothing : noise reduction
 ② Compute gradient : Sobel
 ③ Non-maximum suppression : make edge thinner
 ④ Hysteresis thresholding : connect not edges, 2 steps:
 ⑤ Morphological Thinning, pixel-wise
 ⑥ Edge tracking : only Weak edge that connect with strong edge will remain

Thresholding

- Global threshold: one Threshold for all pixels
- Local threshold: divide image into non-overlapping sections, each section got fix threshold
- Adaptive threshold: For each pixel, the threshold is computed from its local neighborhood (sliding window).

Otsu's Method

Too low -> reduce obj size
 Too high -> include background

Morphological Operations

Morphological

Union: $A \cup B$
 Intersection: $A \cap B$
 Complement: $\neg A$
 Difference: $A - B = A \cap \neg B$
 Reflection: $\bar{A} = \{w | w = -a, a \in A\}$
 Translation: $A_z = \{c | c = a + z, z \in Z\}$
 Dilation: Bridge small gaps and connect objects
 Erosion: Remove small noise or irrelevant details.

Operate pixel-wise

(A) OR (B)
 (A) AND (B)
 (NOT A) AND (B)
 (A XOR B)

Region Filling

$X_0 = (X_1 \oplus B) \cap A$: เน้นสีในรูป พื้นดินเป็น 0 น้ำเป็น 1
 ด้วยเปลี่ยนแล้ว B ให้เป็น +

Boundary Extraction

$B(A) = A - (A \ominus B)$

- $(A \ominus B)$ หรือ A ที่เล็กลง เมื่อลบก็จะได้ข้อบก.
- $(A \ominus B) - A$: ได้ขอบเขต

Connected Components

$X_k = (X_{k-1} \oplus B) \cap A$: พื้นดินเป็น 0 น้ำเป็น 1 แม่น้ำมองเห็นชัดเจนกว่าที่ชื่อ กัน

Point Pattern Recognition

Using Hit-or-Miss transformation to find pattern Bk in A.
 1. Convert A to binary image.
 2. To match the shape of target, dilate A by (W-X).
 3. Hit-or-Miss: $(A \ominus X) \cap (A \oplus (W-X)) = 1$ pixel
 4. Pruning: removes small spurs left after thinning/skeletonization.
 5. Thinning: 1-pixel wide line
 6. Connected components: finds connected components in the skeleton.

Applications

- Morphological Smoothing: Remove bright/dark noise (opening + closing)
- Morphological Grayscale: $f = f \oplus b - (f \ominus b)$: ขาว - 黑
- Textural Segmentation: Distinguish texture regions by structure size.
- Granulometry: Estimate particle size distributions.

FCN – Fully Convolutional Networks

- เปลี่ยนจาก CNN แบบ classification (ลงหัวเขียว FC layer)
 → เป็น network ที่มีแต่ conv layer
- Idea:
 ใช้ CNN encoder ทำให้ feature map เล็กลง
 ใช้ transposed convolution (ConvTranspose2d) หรือ upsampling
 เพื่อขยายกลับไปที่ resolution เดิม. มี skip connection มากขึ้น
 - FCN-32s : ลาก 1x1 (A) -> 32x32 เลย TransConv รอบด้าน
 - FCN-16s : ลาก 2×2 ก่อน เก็บมาบานกับ 2×2 ของ conv encoder (skip connection) ได้ 2×2 (B) -> 32x32
 - FCN-8s : ลาก B -> 4×4 ก่อน เก็บมาบานกับ 4×4 ของ conv encoder (skip connection) ได้ 4×4 -> 32x32

Loss Functions for Segmentation

- ใช้สอง loss รวมกันเพื่อให้ training stable:
 - DiceLoss (1 - Dice-Score) : focus pixel**
 - เน้นให้ overlap ระหว่าง prediction mask กับ GT สรุป
 - ตีม้าวคลาส positive น้อย (object เจ้า ๆ)
 - BCEWithLogitsLoss : focus class**
 - รวม Sigmoid + Binary Cross Entropy ในตัวเดียว เสียบกันไว้ใช้
 - สามารถตั้ง class weight (เพื่อปรับ balance precision/recall)

DeepLab v1

- ใช้ DCNN + Atrous (Dilated) Convolution
 - Dense Coord-Convolution Network (DCNN):
 • Atrous conv: $\text{Labeled coordinate } (x,y) \text{ as input}$
 - ใช้ "ข้อมูล" ใน kernel → receptive field ใหญ่ขึ้น
 - ไม่ต้องเพิ่ม kernel size และไม่ต้อง down-sampling $3 \times 3 \text{ kernel}, \text{dilatation rate} = 2$
 - Output feature map upscale กลับไปตามจุดตัดด้วย bilinear interpolation -> ไม่จำเป็นต้อง Train ใหม่ TransposeConv
 - ใช้ fully connected CRF (Conditional Random) ขับ refine ขอบ object ให้คมชัด
 - เป็น discriminative model → สนใจเรื่อง ๆ เวลา เข้า input ต้องเปลี่ยน label ออกมากเป็นอย่างไร

DeepLab v2 – ASPP (Atrous Spatial Pyramid Pooling)

- ใช้ backbone ที่ atrous conv ขนาดต่างกัน แต่ dilation rate ต่างกัน ทำงานบน scale
 - เพิ่ม Global Average Pooling (context ใหญ่ๆ) และ upsample
 - Concatenate ทุก branch ตาม channel dimension → ให้ feature ร่วมผลลัพธ์

DeepLab v3 / v3+

- Encoder-Decoder Model ต่อจาก U-Net – able to sharp object boundaries
- ใช้ depth-wise separable convolution เพื่อให้คำนวณไวขึ้น
 - depth-wise separable conv ตัวเดียว 1x1 pointwise conv (1x1 ConV0) to increase computational efficiency
- v3+:
 - ใช้ Aligned Xception เมนู ResNet101 เป็น encoder (ลึกและประวัติการพัฒนา)
 - แทน max pooling แห่งเดียวกัน depth-wise separable conv
- EfficientNet-NAS-FPN**
 - Backbone = EfficientNet (compound scaling → balance depth/width/resolution)
 - FPN (Feature Pyramid Network) รวม feature multi-scale
 - ใช้ NAS ทำ architecture FPN ที่ลงตัวกับ backbone → ได้ precision + efficiency

