Structured Forests for fast edge Detection

Sowmya Dasari
IMT2015507

ABSTRACT

- Patches of edges exihibit well known structures such as straight lines and T-junctions.

- Our novel approach to learning decision trees robustly maps the structured labels to a discrete space which standard information gain measures may be evaluated.

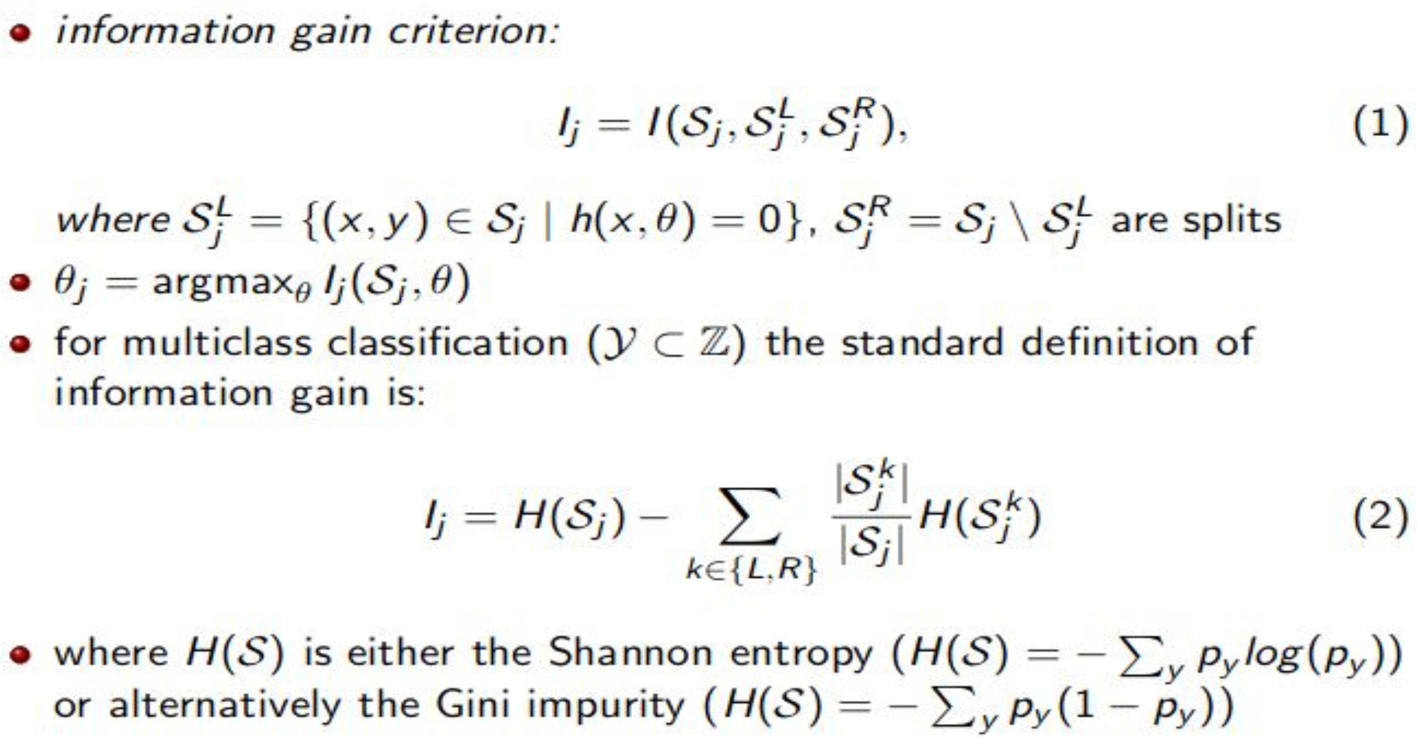
- We formulate the problem of edge detection as predicting local segmentation masks given input image patches.The result is an approach that obtains realtime performance .

- Each forest predicts a patch of edge pixel labels that are aggregated across the image to compute our final edge map.

Older approaches

- Each of these approaches takes an image patch and computes the likelihood that the center pixel contains an edge. The independent edge predictions may then be combined using global reasoning.

PROCESS



- *information gain criterion:*

$$I_j = I(\mathcal{S}_j, \mathcal{S}_j^L, \mathcal{S}_j^R), \tag{1}$$

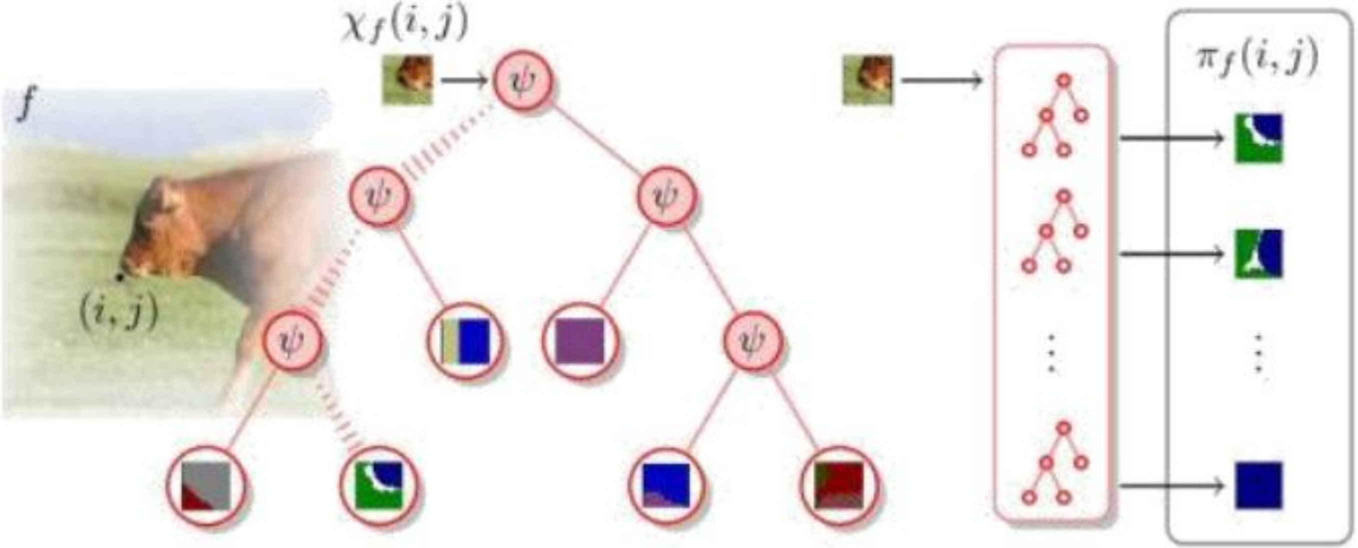
  where $\mathcal{S}_j^L = \{(x, y) \in \mathcal{S}_j \mid h(x, \theta) = 0\}$, $\mathcal{S}_j^R = \mathcal{S}_j \setminus \mathcal{S}_j^L$ are splits

- $\theta_j = \text{argmax}_\theta \, I_j(\mathcal{S}_j, \theta)$
- for multiclass classification ($\mathcal{Y} \subset \mathbb{Z}$) the standard definition of information gain is:

$$I_j = H(\mathcal{S}_j) - \sum_{k \in \{L,R\}} \frac{|\mathcal{S}_j^k|}{|\mathcal{S}_j|} H(\mathcal{S}_j^k) \tag{2}$$

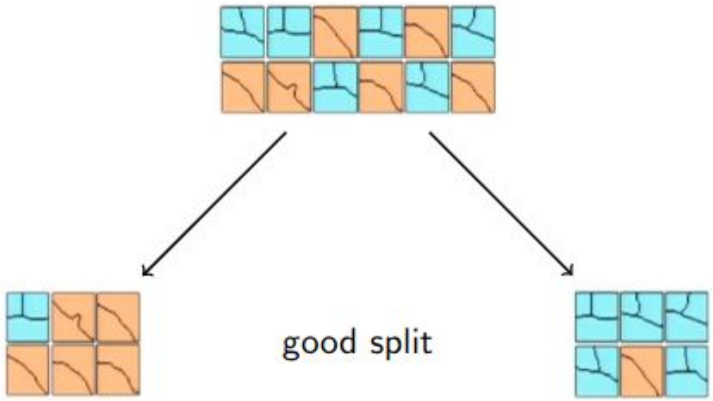
- where $H(\mathcal{S})$ is either the Shannon entropy ($H(\mathcal{S}) = -\sum_y p_y log(p_y)$) or alternatively the Gini impurity ($H(\mathcal{S}) = -\sum_y p_y(1 - p_y)$)

PROCESS

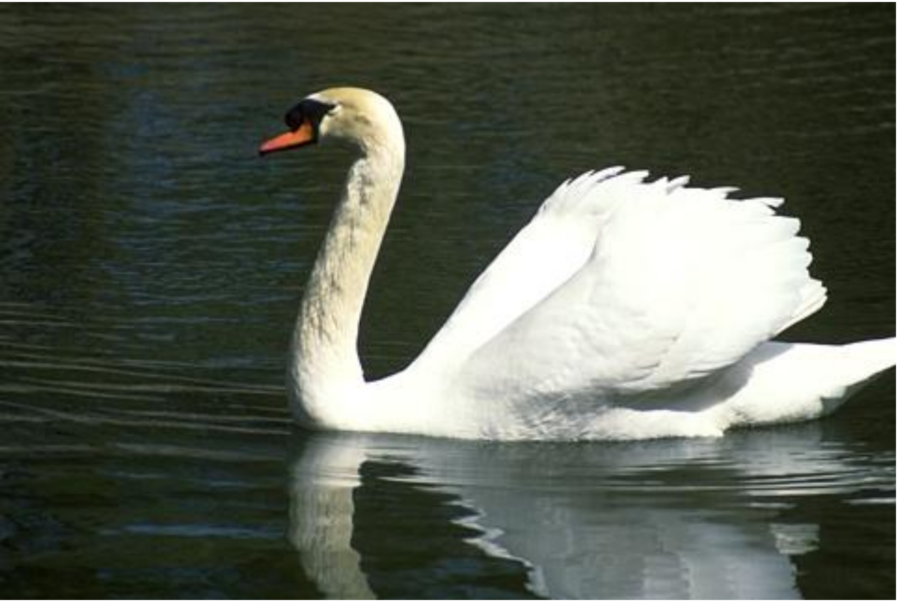


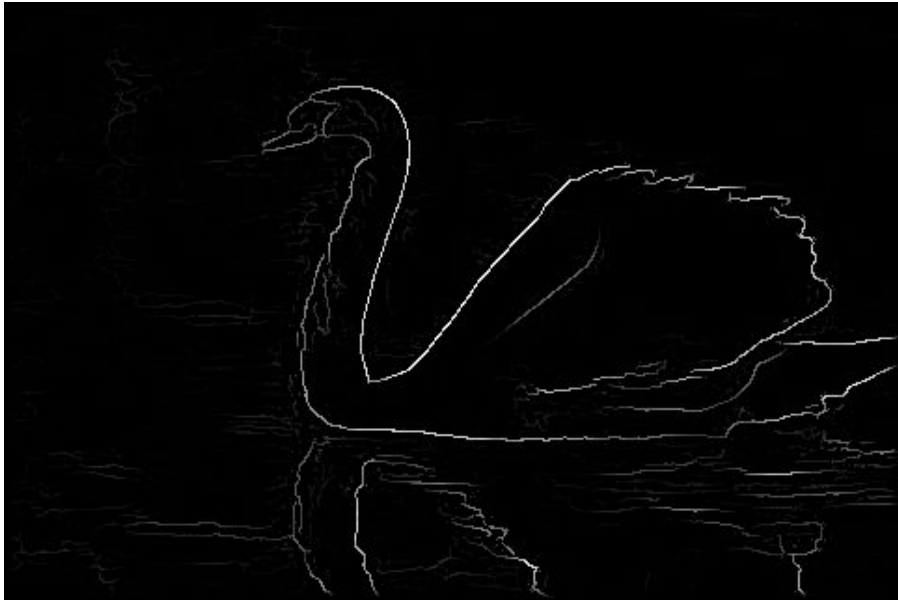
PROCESS

- First, structured output spaces are often high dimensional and complex.
- We use Intermediate mapping to which encodes whether every pair of pixels in the y belongs to the same or different segment.
- Then it is converted into a fixed number of class labels.

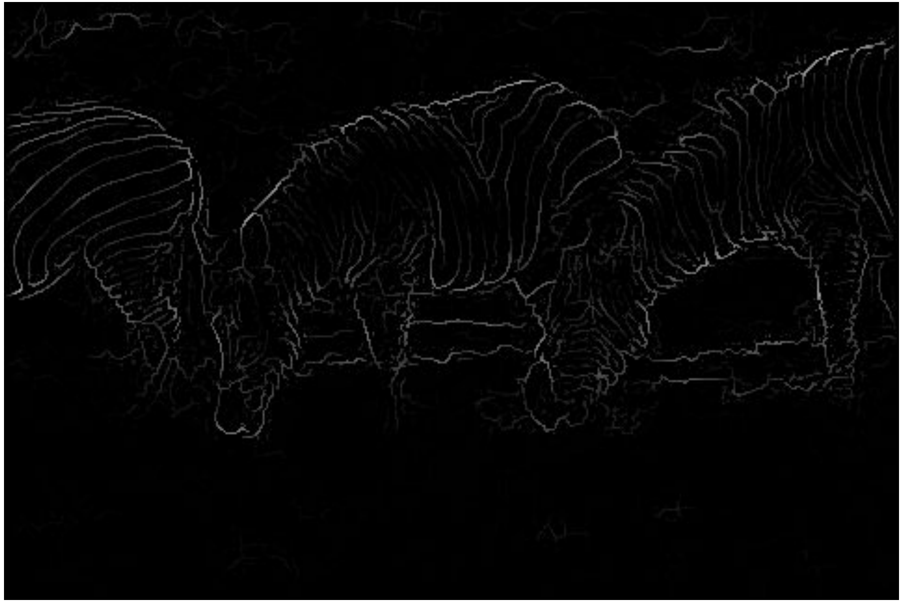


Results

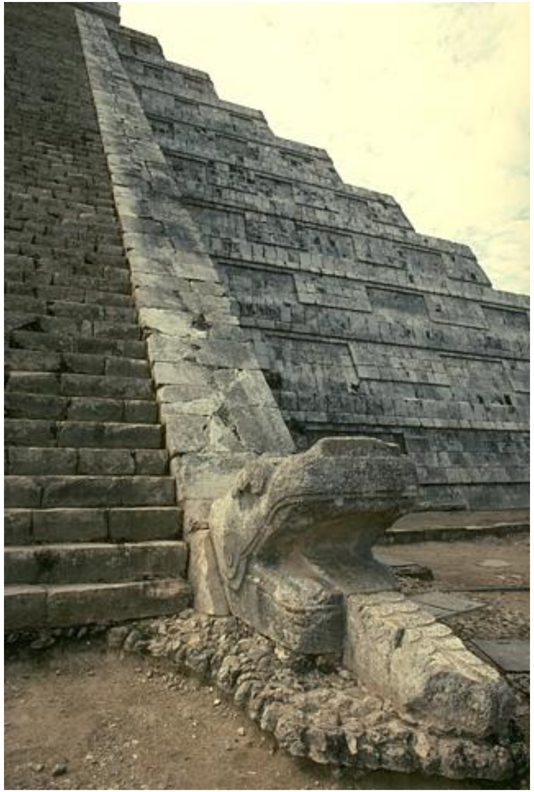


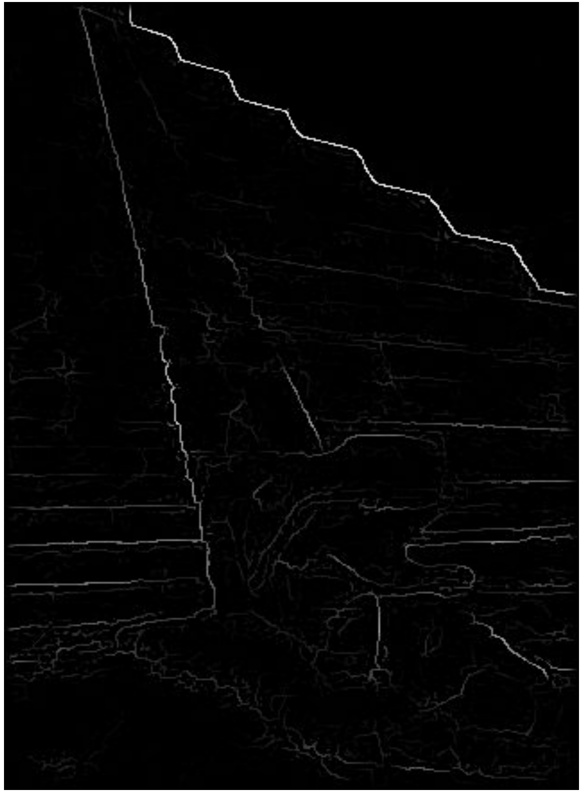


Results

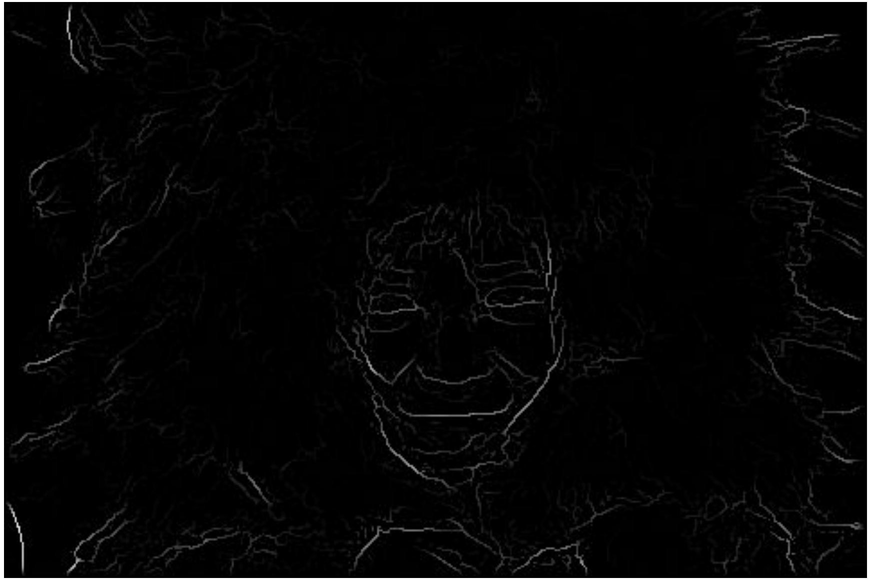


Results

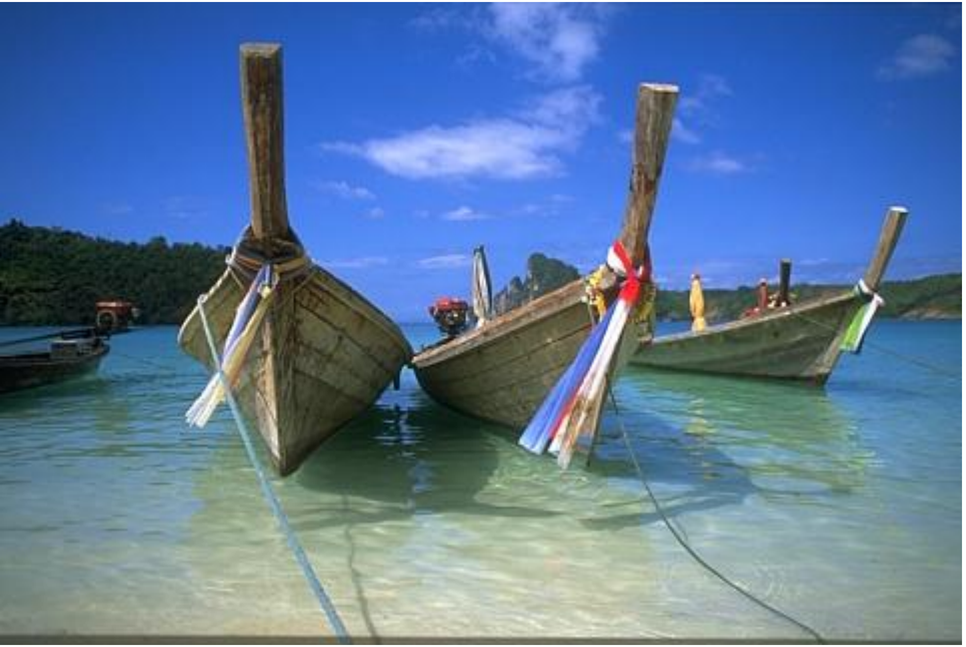


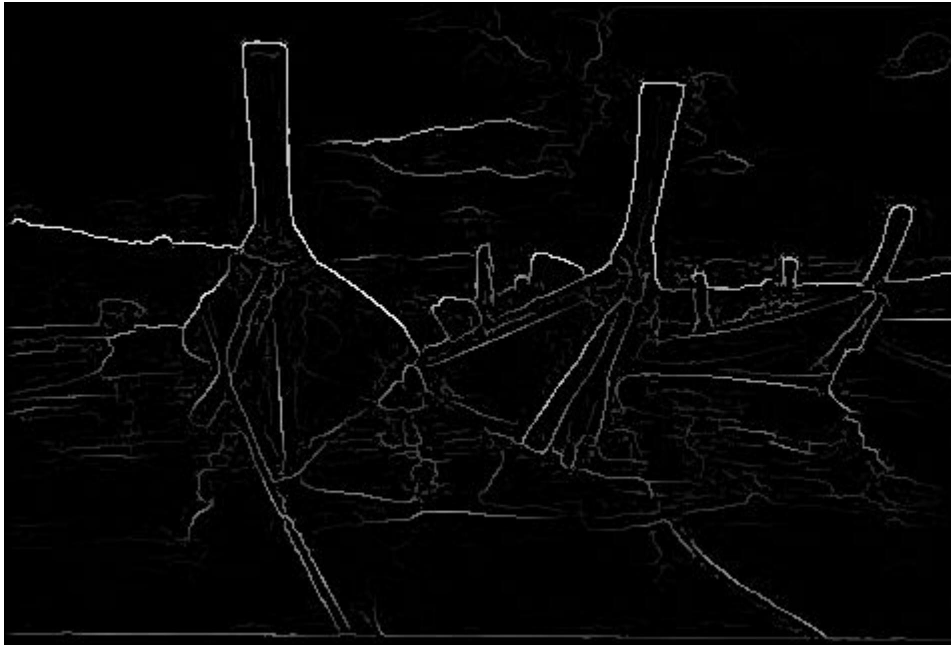


Results

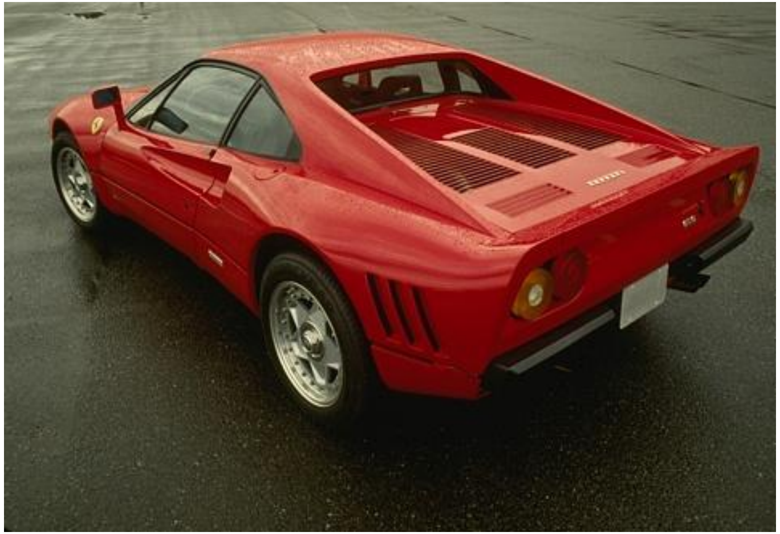


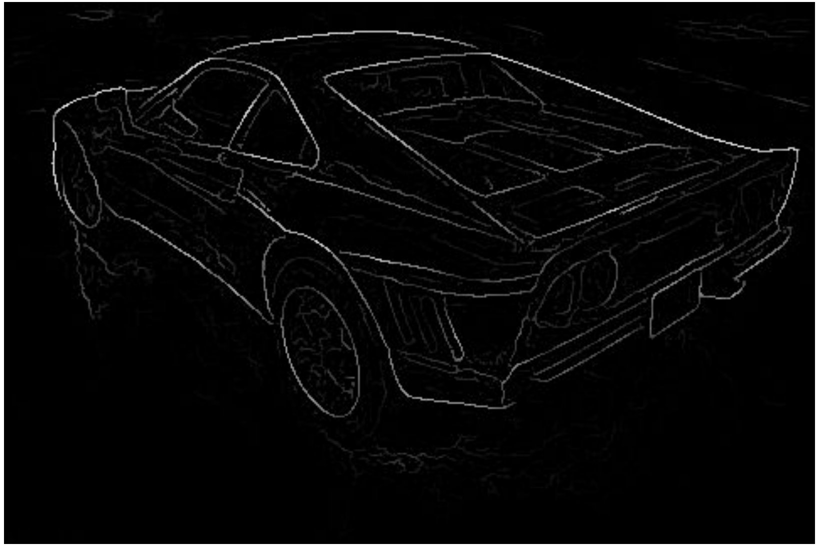
Results





Results





Why is it better than canny edge detector ?

	ODS	OIS	AP	F1-score	FPS
Canny	0.60	0.64	0.58	0.49	15
SE	0.739	0.759	0.796	0.71	40

Novelty

Succesful approach

- I included a sharpening component to sharpen the edges. Due to this speed decreased a little.

Failed approach

- I tried to join some of the broken edges to make it continous but wasn’t succesfull.