ICBV HW #3

Dor Litvak Gal Elgavish

Question 1 - Color

a) Compute $R_i[L]$: the response of the cones (S_i for i = 1,2,3) to the received spectrum of light L.

Receptor response to general light: $R_i[L] = \int_0^\infty S_i(\lambda)L(\lambda)\,d\lambda$

In the discrete case: $R_i[L] = \sum_\lambda S_i(\lambda)L(\lambda)$

We can write the received spectrum of light L as a vector of the wavelengths:

$L = (0, 0, 6, 1, 3, 2, 5, 3, 3, 0, 0)$

Same for the sensitivity of the cones S_1, S_2, S_3:

$S_1 = (0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0)$

$S_2 = (0, 0, 0, 0, 1, 2, 1, 0, 0, 0, 0)$

$S_3 = (0, 0, 1, 6, 1, 0, 0, 0, 0, 0, 0)$

We will use the discrete equation above to calculate the response of the cones to the received spectrum of light L:

$R_1[L] = \sum_\lambda S_1(\lambda)L(\lambda) = (0\cdot0)+(0\cdot0)+(0\cdot6)+(0\cdot1)+(3\cdot3)+(0\cdot2)+(0\cdot5)+(0\cdot3)+(0\cdot3)+(0\cdot0)+(0\cdot0) = 9$

$R_2[L] = \sum_\lambda S_2(\lambda)L(\lambda) = (0\cdot0)+(0\cdot0)+(0\cdot6)+(0\cdot1)+(1\cdot3)+(2\cdot2)+(1\cdot5)+(0\cdot3)+(0\cdot3)+(0\cdot0)+(0\cdot0) = 12$

$R_3[L] = \sum_\lambda S_3(\lambda)L(\lambda) = (0\cdot0)+(0\cdot0)+(1\cdot6)+(6\cdot1)+(1\cdot3)+(0\cdot2)+(0\cdot5)+(0\cdot3)+(0\cdot3)+(0\cdot0)+(0\cdot0) = 15$

b) Compute $\gamma_{i,j}$: the response of the cones (S_i for i = 1,2,3) to the base colors (P_j for j = 1,2,3).

Receptor response to base colors: $\gamma_{i,j} = \int_0^\infty S_i(\lambda)P_i(\lambda)\,d\lambda$

In the discrete case: $\gamma_{i,j} = \sum_\lambda S_i(\lambda)P_i(\lambda)$

We can write the sensitivity of the cones S_1, S_2, S_3 to different wavelengths as vectors:

$S_1 = (0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0)$

$S_2 = (0, 0, 0, 0, 1, 2, 1, 0, 0, 0, 0)$

$S_3 = (0, 0, 1, 6, 1, 0, 0, 0, 0, 0, 0)$

Same for the spectra of the 3 base colors P_1, P_2, P_3:

$P_1 = (0, 0, 0, 3, 1, 3, 4, 5, 4, 0, 0)$

$P_2 = (0, 0, 0, 5, 3, 2, 1, 1, 1, 0, 0)$

$P_3 = (0, 5, 3, 2, 1, 1, 1, 0, 0, 0, 0)$

We will use the discrete equation above to calculate the response of the cones to the base colors:

$\gamma_{1,1} = \sum_\lambda S_1(\lambda)P_1(\lambda) = (0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0) \cdot (0, 0, 0, 3, 1, 3, 4, 5, 4, 0, 0)^T = (3 \cdot 1) = 3$

$\gamma_{1,2} = \sum_\lambda S_1(\lambda)P_2(\lambda) = (0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0) \cdot (0, 0, 0, 5, 3, 2, 1, 1, 1, 0, 0)^T = (3 \cdot 3) = 9$

$\gamma_{1,3} = \sum_\lambda S_1(\lambda)P_3(\lambda) = (0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0) \cdot (0, 5, 3, 2, 1, 1, 1, 0, 0, 0, 0)^T = (3 \cdot 1) = 3$

$\gamma_{2,1} = \sum_\lambda S_2(\lambda)P_1(\lambda) = (0, 0, 0, 0, 1, 2, 1, 0, 0, 0, 0) \cdot (0, 0, 0, 3, 1, 3, 4, 5, 4, 0, 0)^T = (1 \cdot 1) + (2 \cdot 3) + (1 \cdot 4) = 11$

$\gamma_{2,2} = \sum_\lambda S_2(\lambda)P_2(\lambda) = (0, 0, 0, 0, 1, 2, 1, 0, 0, 0, 0) \cdot (0, 0, 0, 5, 3, 2, 1, 1, 1, 0, 0)^T = (1 \cdot 3) + (2 \cdot 2) + (1 \cdot 1) = 8$

$\gamma_{2,3} = \sum_\lambda S_2(\lambda)P_3(\lambda) = (0, 0, 0, 0, 1, 2, 1, 0, 0, 0, 0) \cdot (0, 5, 3, 2, 1, 1, 1, 0, 0, 0, 0)^T = (1 \cdot 1) + (2 \cdot 1) + (1 \cdot 1) = 4$

$\gamma_{3,1} = \sum_\lambda S_3(\lambda)P_1(\lambda) = (0, 0, 1, 6, 1, 0, 0, 0, 0, 0, 0) \cdot (0, 0, 0, 3, 1, 3, 4, 5, 4, 0, 0)^T = (6 \cdot 3) + (1 \cdot 1) = 19$

$\gamma_{3,2} = \sum_\lambda S_3(\lambda)P_2(\lambda) = (0, 0, 1, 6, 1, 0, 0, 0, 0, 0, 0) \cdot (0, 0, 0, 5, 3, 2, 1, 1, 1, 0, 0)^T = (6 \cdot 5) + (1 \cdot 3) = 33$

$\gamma_{3,3} = \sum_\lambda S_3(\lambda)P_3(\lambda) = (0, 0, 1, 6, 1, 0, 0, 0, 0, 0, 0) \cdot (0, 5, 3, 2, 1, 1, 1, 0, 0, 0, 0)^T = (1 \cdot 3) + (6 \cdot 2) + (1 \cdot 1) = 16$

c) Is there a spectrum of light $\widetilde{L} = \beta_1 P_1 + \beta_2 P_2 + \beta_3 P_3$ that generates the same response from the cones as L

We saw in class that:

$R_i[\widetilde{L}] = \int_0^\infty S_i(\lambda)\widetilde{L}(\lambda)\, d\lambda = \sum_{j=1}^{3} \beta_j \gamma_{i,j}$

We want to find the values of β_j j = 1,2,3 such that $R_i[L] = R_i[\widetilde{L}]$

In another words: $R_i[L] = R_i[\widetilde{L}] = \sum_{j=1}^{3} \beta_j \gamma_{i,j}$

From part (a) we know: $R_1[L] = 9, R_2[L] = 12, R_3[L] = 15$

$R_1[L] = 9 = \sum_{j=1}^{3} \beta_j \gamma_{1,j} = \beta_1\gamma_{1,1} + \beta_2\gamma_{1,2} + \beta_3\gamma_{1,3} = 3\beta_1 + 9\beta_2 + 3\beta_3$

$R_2[L] = 12 = \sum_{j=1}^{3} \beta_j \gamma_{2,j} = \beta_1\gamma_{2,1} + \beta_2\gamma_{2,2} + \beta_3\gamma_{2,3} = 11\beta_1 + 8\beta_2 + 4\beta_3$

$R_3[L] = 15 = \sum_{j=1}^{3} \beta_j \gamma_{3,j} = \beta_1\gamma_{3,1} + \beta_2\gamma_{3,2} + \beta_3\gamma_{3,3} = 19\beta_1 + 33\beta_2 + 16\beta_3$

We will solve system of 3 linear equations:

$9 = 3\beta_1 + 9\beta_2 + 3\beta_3$

$12 = 11\beta_1 + 8\beta_2 + 4\beta_3$

$15 = 19\beta_1 + 33\beta_2 + 16\beta_3$

The solution is:

$\beta_1 = \frac{44}{31} = 1.41935$

$\beta_2 = \frac{77}{31} = 2.48387$

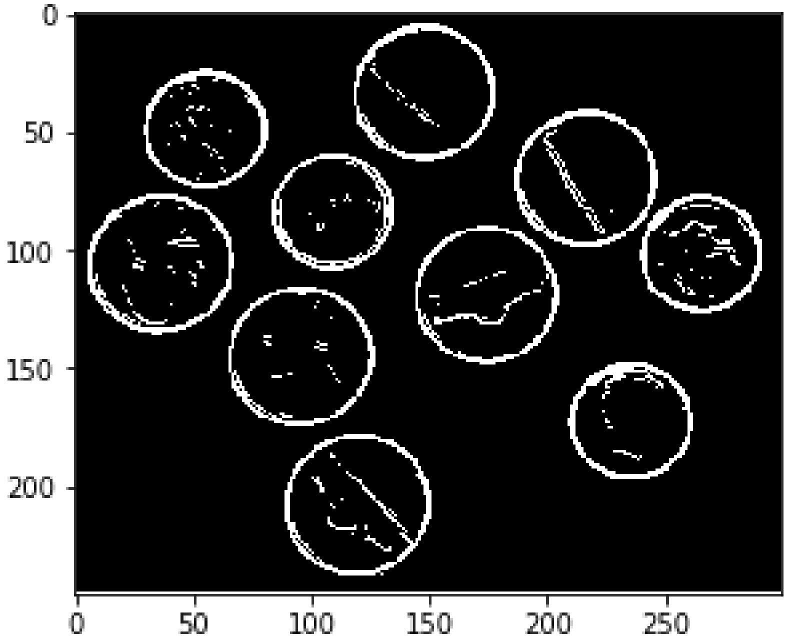
$\beta_3 = -\frac{182}{31} = -5.87096$

Question 2 - Hough Transform

1) Produce an edge map from the image using an edge detector of your choice

Since we needed the gradients' directions for the Hough transform to be efficient, we chose a 1^{st} order operator.

So, we used Sobel's 3×3 kernels (with smoothing). And used G_x, G_y to calculate the gradient's magnitude and direction in each pixel.
We normalized the magnitude to be in $[0, 255]$, and, returned 1 when the normalized magnitude is above the threshold (a hyperparameter), and 0 otherwise.
We also saved the orientation of the gradient in each pixel for the Hough transform (where the angles are in $[-\pi, \pi]$).



2) Detect all the circles in the image using Hough transform

Our HoughCircles function gets the following parameters:

- edge_map - from the edge detector
- direction_map - from the edge detector (the gradients' orientations)
- min_r - minimum radius to scan
- max_r - maximum radius to scan
- n_r - # of buckets in radius
- n_x - # of buckets in height
- n_y - # of buckets in width
- maxima_kernel_size - for the adaptive threshold
- maxima_constant - for the adaptive threshold

We chose min_r=15, max_r=40 since it covered all of the circles with spare.
We chose 100 buckets for each parameter to be off by roughly two pixels and make the radius more correct.

After we got all the (x_i, y_i, θ_i) coordinates and angles of the edges, our pseudo-code for collecting the votes is:

$\vec{Rs} \leftarrow$ numpy.linspace(min_r, max_r, n_r)
for (x_i, y_i, θ_i) in edges **do**
    for r_j in $\vec{Rs}$ **do**
        $C_{x(i,j)} \leftarrow x_i + r_j \cdot \sin\theta_i$
        $C_{y(i,j)} \leftarrow y_i - r_j \cdot \cos\theta_i$
    end for
end for

This is equivalent to solving the following system for each $r_j, (x_i, y_i, \theta_i)$:

$$\begin{bmatrix} 1 & 0 & -r_j \cdot \sin\theta_i \\ 0 & 1 & r_j \cdot \cos\theta_i \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} C_{x(i,j)} \\ C_{y(i,j)} \\ 1 \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

In order to be more efficient we built and solved the following sparse system:

$$Ax = b$$

A is a diagonal $(3 \cdot n_{edges} \cdot n_r \times 3 \cdot n_{edges} \cdot n_r)$ sparse matrix where:

- main diagonal is: $\vec{\mathbf{1}}$
- 1^{st} upper off-diagnoal is: $\cos\vec{\theta} \otimes \vec{Rs} \otimes \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$
- 2^{nd} upper off-diagnoal is: $-\sin\vec{\theta} \otimes \vec{Rs} \otimes \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$

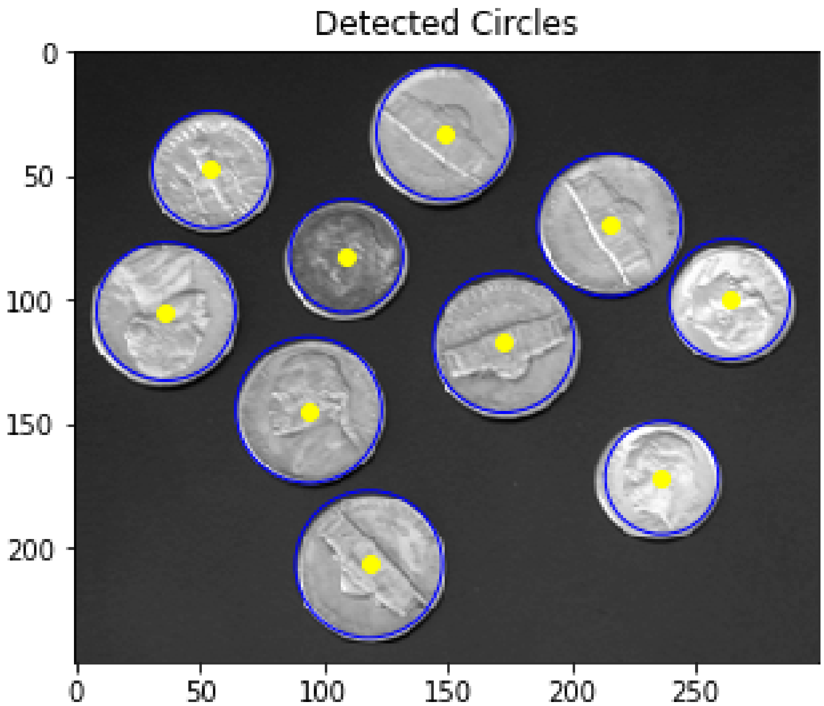
Where $\otimes$ is the Kronecker product, and $\vec{\theta}$ is the flattened gradients' orientations array.

Vector b is a $(3 \cdot n_{edges} \cdot n_r \times 1)$ vector, constructed by tiling (numpy.tile) the $(n_{edges} \times 3)$ homogeneous-coordinates of the edges array for n_r times, and flatten the result.

The resulted vector x is then reshaped, concatenated with $\vec{Rs}$ and reshaped again to result in an $(n_{edges} \cdot n_r \times 3)$ all_votes array, i.e. every row is a single vote from a single edge point containing $(C_{x(i,j)}, C_{y(i,j)}, r_j)$.

Then we used numpy.histogramdd to collect the votes into the different bins: hist_votes array is a tensor of size $(n_x \times n_y \times n_r)$.
In order to find the local maximums we subtracted from each point in the histogram the average neighborhood of it (by convolving with a 3-D cube of size maxima_kernel_size=3).
We then sorted the convolved array and iterated from high picked circle to low (only for values above maxima_constant=30), checking that each new circle's center is outside of previously found circles.

3) Plot the detected circles on top of the original coins image



Question 3 - Curvature

Write an expression for the Frenet frame of the following regular curve: $a(t) = \frac{1}{2}(cos(2t), sin(2t))$ .That is, write an expression for its tangent vector $T(t)$, its normal vector $N(t)$, and its curvature $\kappa(t)$, for every value of the parameter t.

We learned in class the Frenet frame and equation:

$T(t) = a'(t)$

$\kappa(t) = ||T'(t)|| = ||a''(t)||$

$N(t) = \frac{T'(t)}{||T'(t)||} = \frac{T'(t)}{\kappa(t)}$

For the given equation $a(t) = \frac{1}{2}(cos(2t), sin(2t))$:

$T(t) = a'(t) = (-sin(2t), cos(2t))$

$T'(t) = a''(t) = (-2cos(2t), -2sin(2t))$

$\kappa(t) = ||(-2cos(2t), -2sin(2t))|| = \sqrt{(-2cos(2t))^2 + (-2sin(2t))^2} = \sqrt{4(cos(2t)^2 + sin(2t)^2)} = \sqrt{4} = 2$

$N(t) = \frac{T'(t)}{\kappa(t)} = \frac{(-2cos(2t), -2sin(2t))}{2} = (-cos(2t), -sin(2t))$

Question 4 - Segmentation and Relaxation Labeling

Our relaxation labeling algorithm gets the following inputs:

- img - input image
- n_L - the maximum number of labels in the image. We choose to round the angels
- k_size - the size of the compatibility kernel
- epsilon - for the convergence test
- max_iter - maximum number of iterations

Since it is known that in our "world" visually coherent regions are constant gradient patches, we chose the labels as the possible directions of the gradients, and the initial confidence is computed by the similarity of each pixel direction to each of the possible directions.

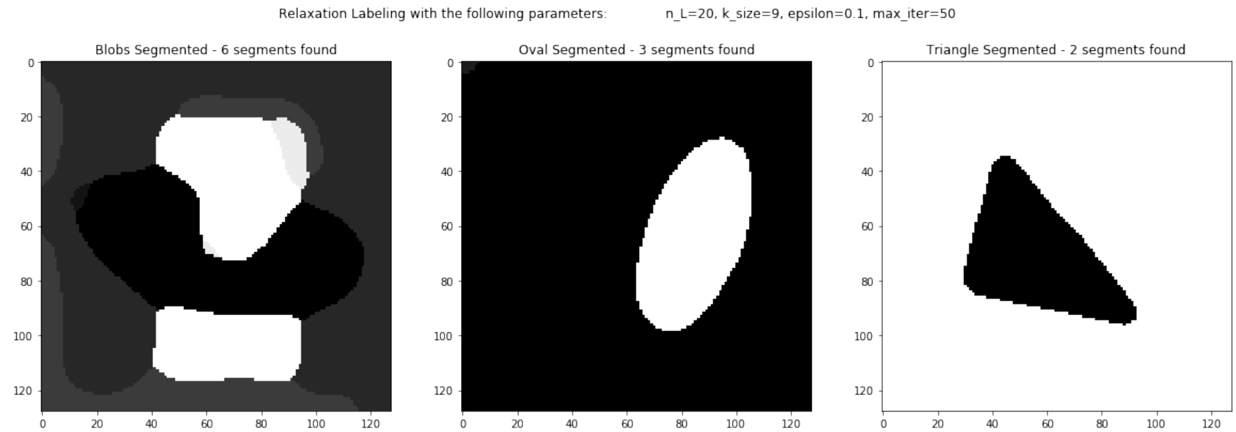
These are the Relaxation Labeling parameters:

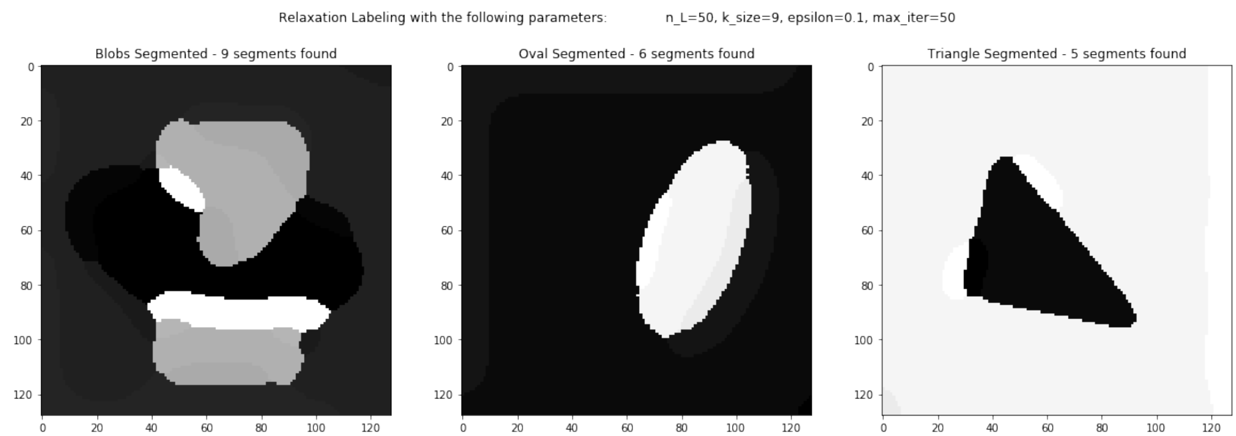
- $B = \{\text{pixels in the image}\}$
- $\Lambda = \{n \cdot \frac{360°}{n_L - 1}\} \forall n \in [0, n_L)$
- $r_{i,j}(\Lambda_i, \Lambda_j) = \begin{cases} \frac{1}{k\_size^2} & \text{if } \Lambda_i = \Lambda_j \text{ and } j \text{ is in the k\_size neighborhood of } i \\ 0 & \text{otherwise} \end{cases}$
- $p_i^0(\Lambda_j) = \frac{1 + \cos \theta_i - \Lambda_j}{\sum_{k=1}^{n\_L} 1 + \cos \theta_i - \Lambda_k}$ where θ_i is the gradients' orientation of pixel i.

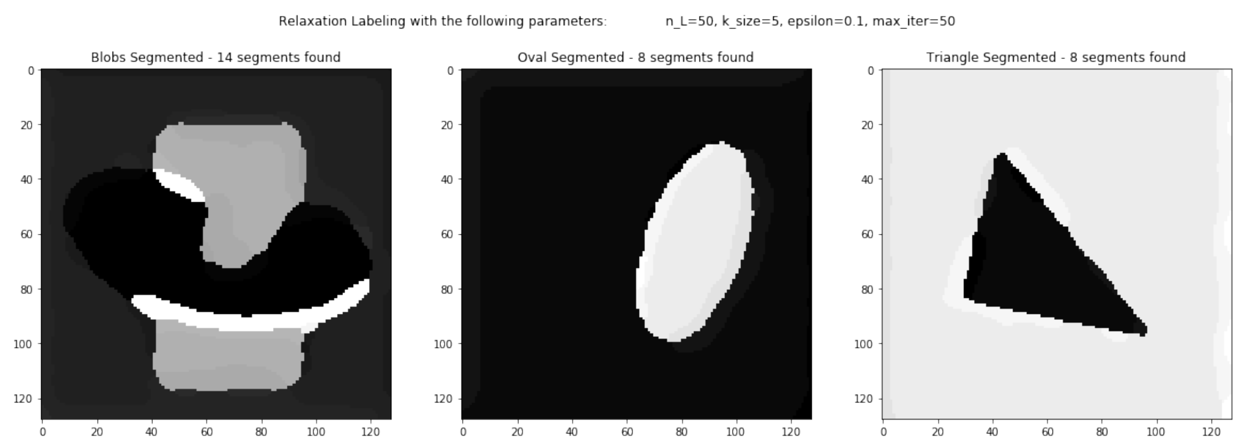
Then our algorithm performs the following computational steps:

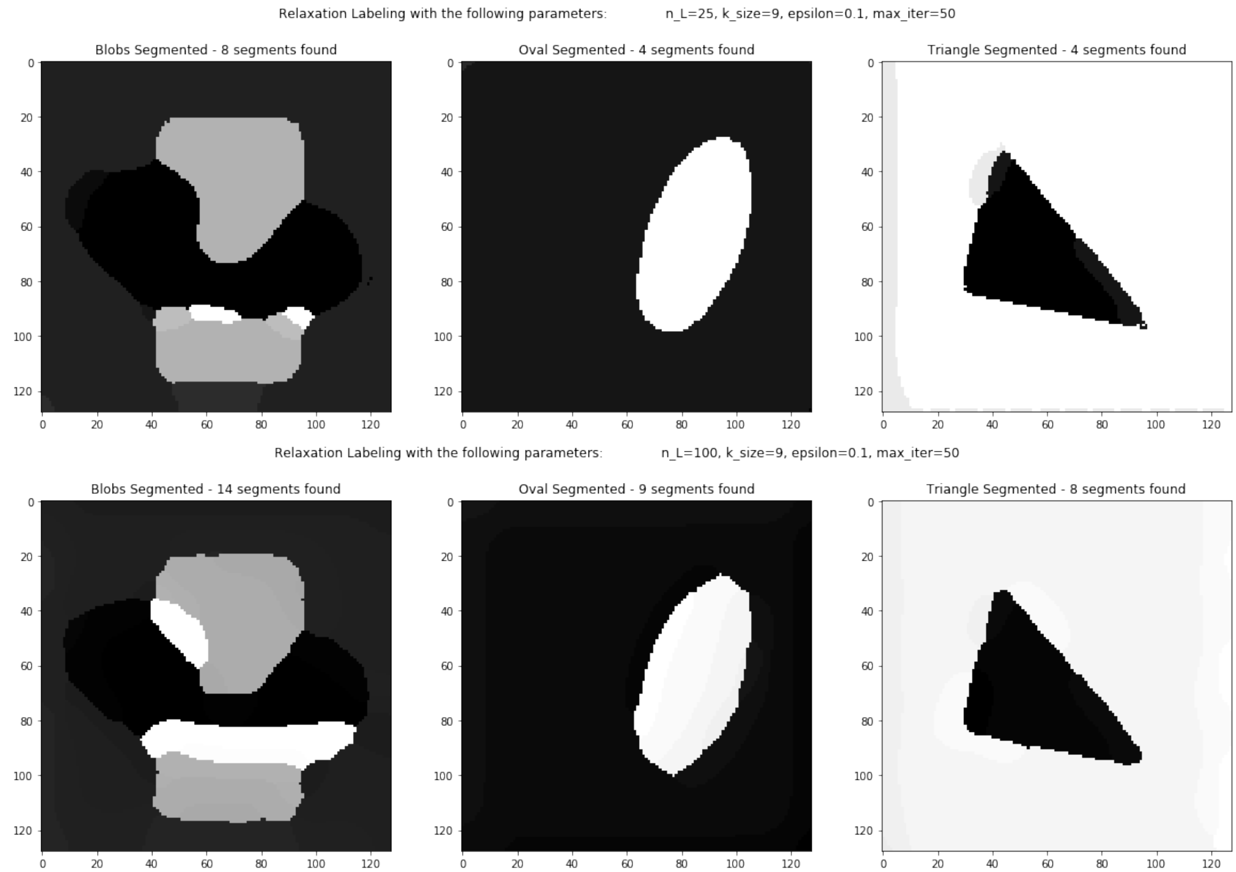
1. We are computing the initial confidence array P of size $(n_L \times n_{rows} \times n_{cols})$ by first finding the array θ (gradients' orientation) of the same size of the input image, by using first order operator $[-1, 0, 1]$ in each direction and computing the angle with numpy.arctan2.
   Then P^0 is computed like explained above (in the Relaxation Labeling parameters).
2. We then build the K_compatibility cubic array which is of size $(k_{size} \times k_{size} \times k_{size})$, where: $K[0, :, :] = K[2, :, :] = 0$ and $K[1, :, :] = \frac{1}{k\_size^2}$
3. We initialize the ALCs (average local consistencies) list to be [inf] in order for the first iteration to not converge.
4. We are performing up to max_iter iterations, and in each one:
   (a) The support array S of the same size as P is computed by convolving P with K_compatibility array.
   (b) We update P by the update rule learned in class.
   (c) We compute and archive the current ALC
   (d) If $\|ALC_k - ALC_{k-1}\| < \epsilon$ we stop the iterations.
5. We return the numpy.argmax of P on the labels axis as the chosen label.

Here are results for different parameters:









Question 5 - Reading Material

a) In class, the concept of common fate was mentioned. What is the additional name Wertheimer suggested for this concept?

Additional name Wertheimer suggested to the concept of *common fate* is name of *Uniform Destiny*.

Uniform Destiny - Humans tend to perceive elements moving in the same direction as being more related than elements that are stationary or that move in different directions.

b) List all the lows of perceptual organization mentioned in the article, and explain each of them in one sentence.

The Factor of Proximity

Objects with more proximity are perceived in the same group.

The Factor of Similarity

Objects with more similarity are perceived as grouped together.

The Factor of Uniform Destiny (Common fate)

Objects that move together are perceived in the same group.

The Factor of Objective Set

The same group in different scenes would be perceived as different objects.

The Factor of Closure

Objects that create a closed shape are perceived in the same group.

The Factor of the "Good Curve"

Objects tend to be perceived as they create a complete shape.

The Factor of Direction

Objects tend to be perceived as creating successive parts that follow one another.

The Factor of the Good Gestalt

Objects that create continuity, in lines or in shape, tend to perceived together.

The Factor of past experience

Pre-experience and knowledge about the world influence the objects perceived together. The human brain knows that the sun is always up. We tend to perceive objects we do not know as that light on them is from the top. (Example from class).