2017

Image segmentation

COMPUTER VISION – LAB 7/8

JOSEP FAMADAS ALSAMORA / JORDI RIU VICENTE

UNIVERSITAT DE BARCELONA

UB | Facultat de Matemàtiques i Informàtica

Table of contents

Table of Figures

Table of Tables

1. Bottom-up segmentation

1.1. K-Means

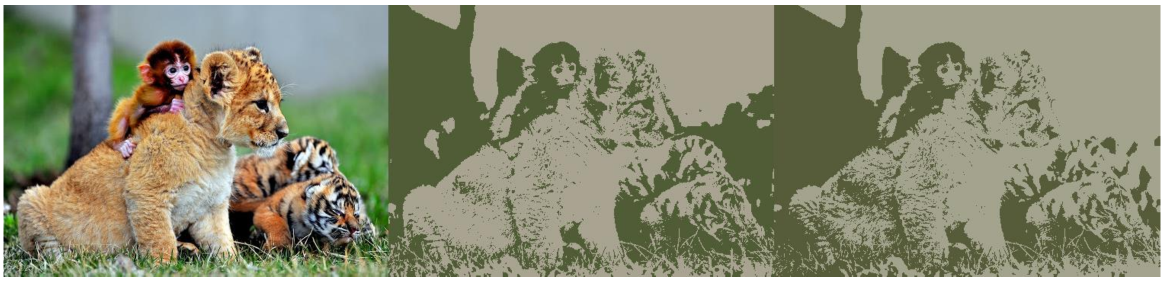


Figure 1: K-Means with K=2 (original, only RGB, RBG + XY)

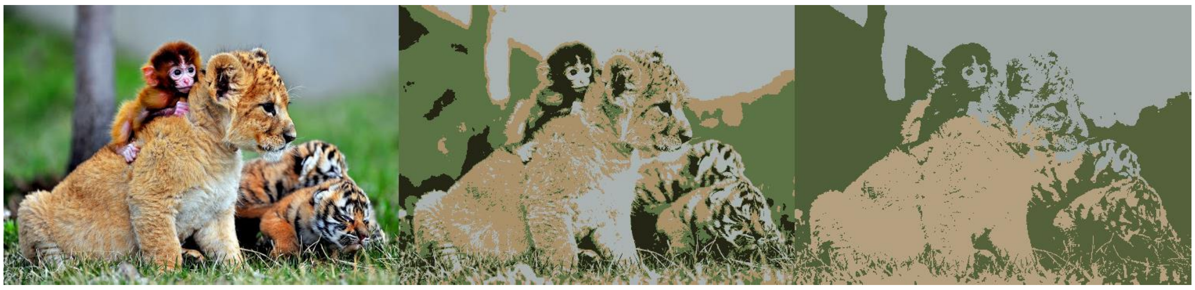


Figure 2: K-Means with K=4 (original, only RGB, RBG + XY)

Ideally, we would like to separate the animals from the background, so the ideal number of clusters would be 2. However, in *Figure 1* we see that with K = 2 the differentiation is not good enough. Therefore, we have increased the number of clusters up to 4 for which we appreciate that the 'only RGB' image correctly separates the animals from the background.

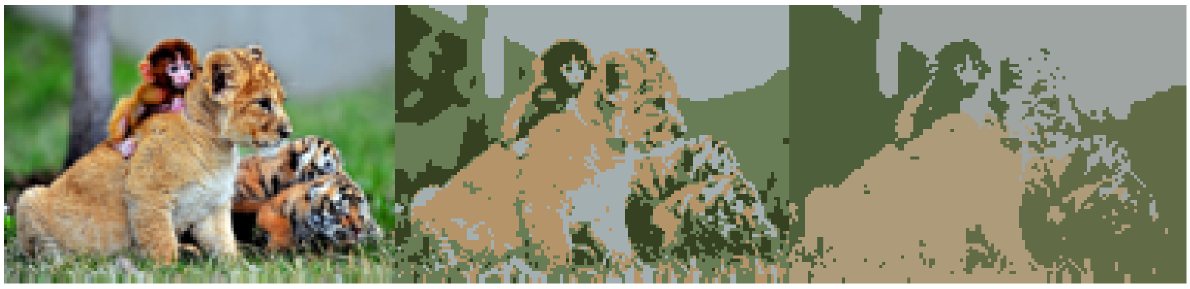


Figure 3: K-Means | K=4 | resize = 0.25 (original, only RGB, RBG + XY)

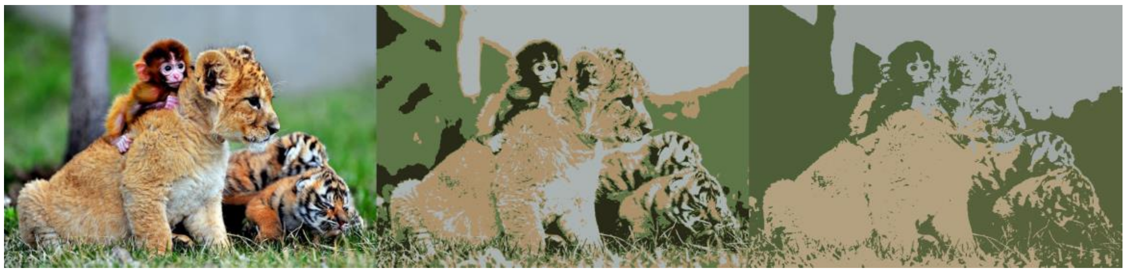


Figure 4: K-Means | K=4 | resize = 4 (original, only RGB, RBG + XY)

We can observe that reducing the size of the image (*Figure 3*) or increasing it (*Figure 4*) does not have a great impact on the final clustering result. However, we can see that reducing the size might be better to avoid the noise when clustering.

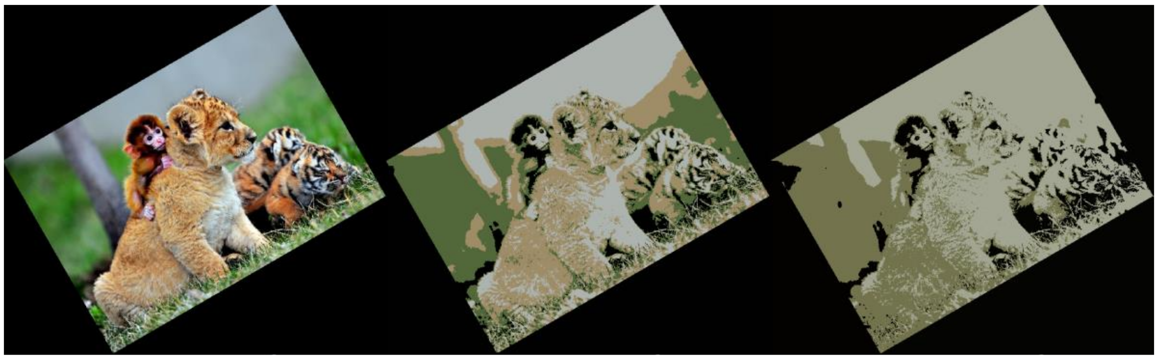


Figure 5: K-Means | K=4 | rotate = 30º (original, only RGB, RBG + XY)

In *Figure 5* it can be appreciated that rotation only affects the clustering when the spatial information is taken into account.

When the K-Means method is used, we cannot expect the output result to be always the same because the initialization of the clusters is random.

On the other hand, we should also take into account that a concrete initialization will always lead to the same result, so we could say that the method is partially deterministic.

The clusters are the number of sets in which we divide all the pixels of the image.

1.2. Mean-shift segmentation

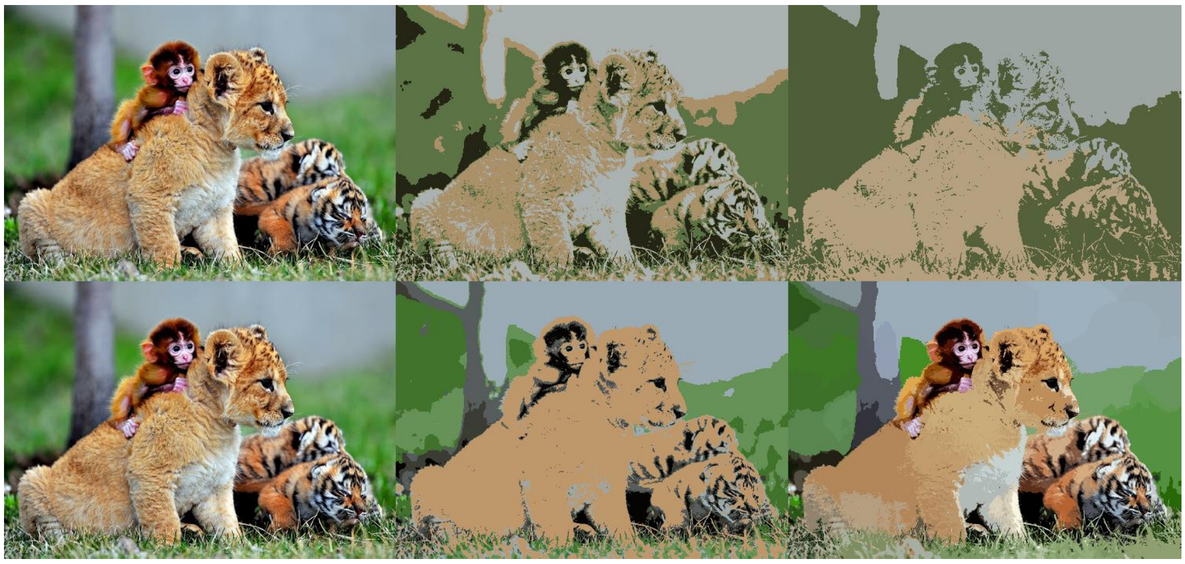


Figure 6: K-Means (Up) vs Mean-shift (Down) || Original (Left), RGB (Middle), RGB+XY (Right)

In *Figure 6* we can see that the mean-shift algorithm performs far better than the K-Means when we want to separate the animals from the background. The main limitation we have observed in the mean-shift algorithm is its computational time.

In this algorithm the number of clusters does not need to be determined a priori and is the own algorithm that does it.

The parameter we have to fix here is the bandwidth which delimits the shape of the peaks in the histogram we are going to accept and this will directly affect the number of clusters.

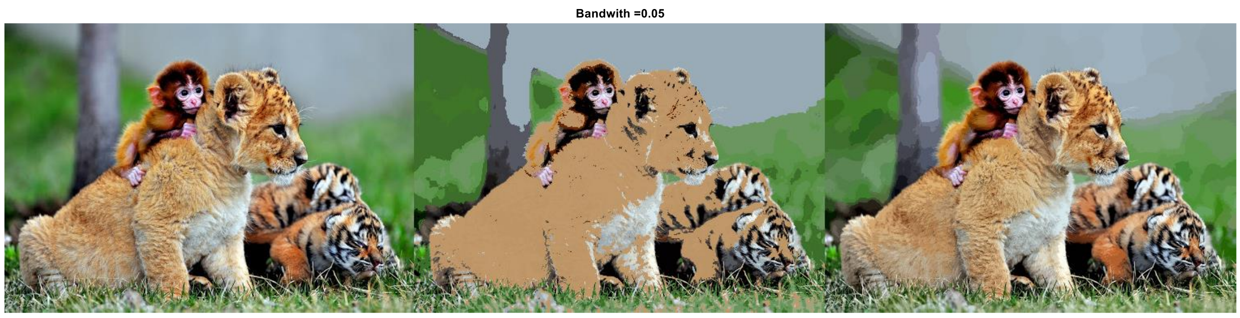


Figure 7: Mean-shift (Down) | BW = 0.05 || Original (Left), RGB (Middle), RGB+XY (Right)

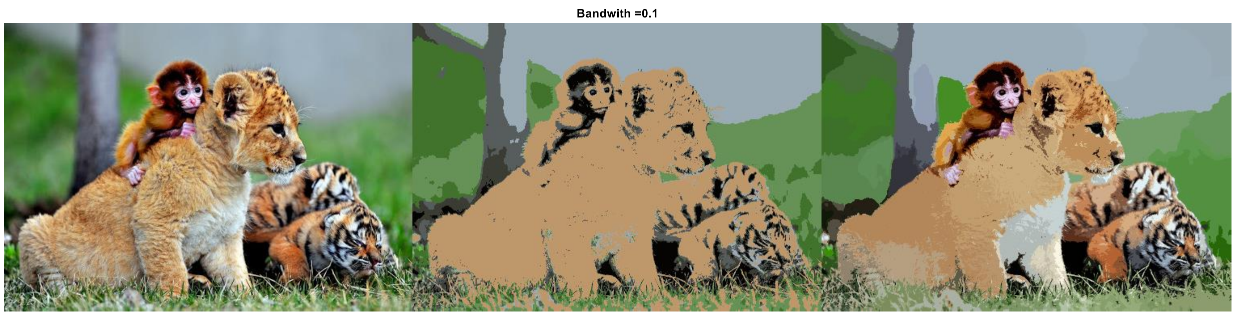


Figure 8: Mean-shift (Down) | BW = 0.1 || Original (Left), RGB (Middle), RGB+XY (Right)

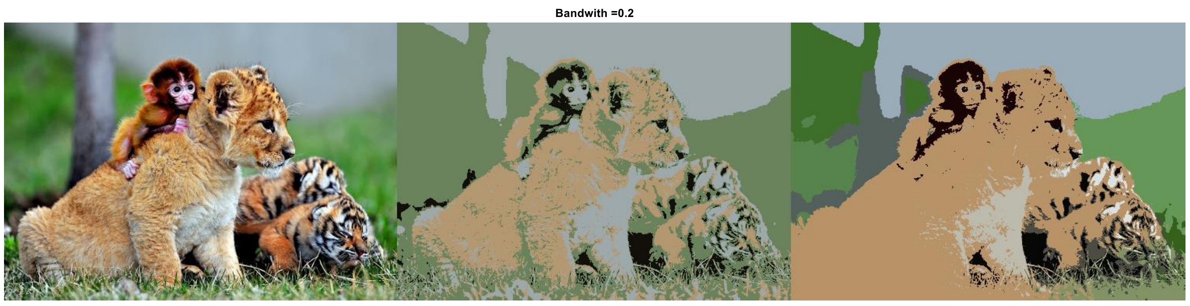
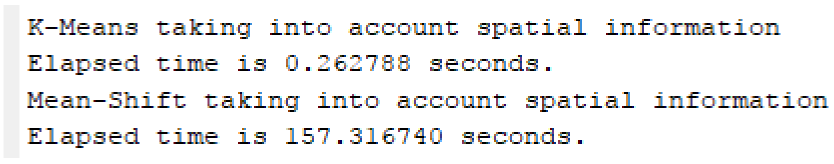


Figure 9: Mean-shift (Down) | BW = 0.2 || Original (Left), RGB (Middle), RGB+XY (Right)

From the different tries performed in figures from 7 to 9 we have decided that the optimal bandwidth is 0.1, because despite the fact that with a bandwidth of 0.05 the image is better defined, the number of clusters is too high and so is the computational time.

On the other side, if the bandwidth is too large (0.2 for example) the number of clusters is too low and the resulting image is far from being good.

The algorithm is deterministic because for a single value of bandwidth it will always converge to the same segmentation.



```
K-Means taking into account spatial information
Elapsed time is 0.262788 seconds.
Mean-Shift taking into account spatial information
Elapsed time is 157.316740 seconds.
```

Figure 10: Execution time for K-Means and Mean-Shift algorithms

In *Figure 10* we can check that, as it has been commented before, the execution time of the mean shift algorithm is more than 5000 times longer than the K-Means one.

2. Top-down segmentation

2.1. Active contour models

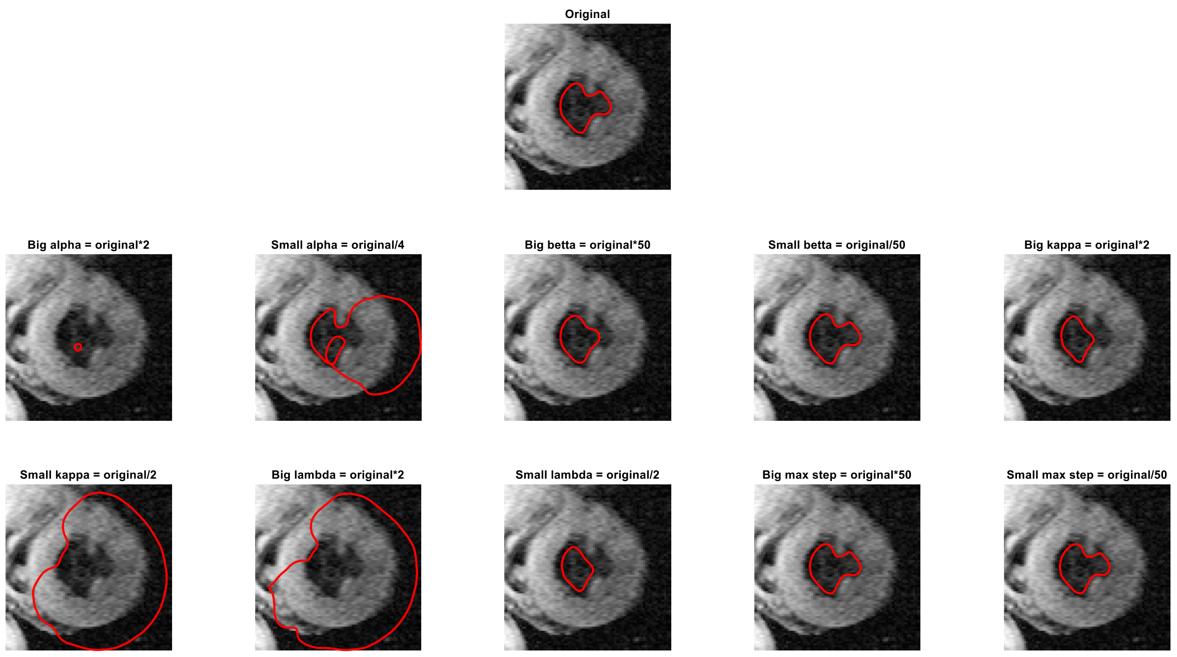


Figure 11: Energy minimization point model (Snakes) critical parameters (heart image)

	Optimal	*Lower critical (approx.)*	*Upper critical (approx.)*
Alpha	0.1	Optimal * 2	Optimal / 4
Betta	0.01	Optimal * 50	Optimal / 50
Lambda	0.05	Optimal * 2	Optimal / 2
Kappa	0.2	Optimal * 2	Optimal / 2
Max step	0.4	Optimal * 50	Optimal / 50

Table 1: Snake parameters (heart image)

To get *Figure 11* we performed the segmentation by active contour models fixing as optimal the provided default parameters and modifying each of them separately up and down until we observed that the algorithm behaved poorly. We considered a range of values from 50 times lower the optimal to 50 times higher the optimal.

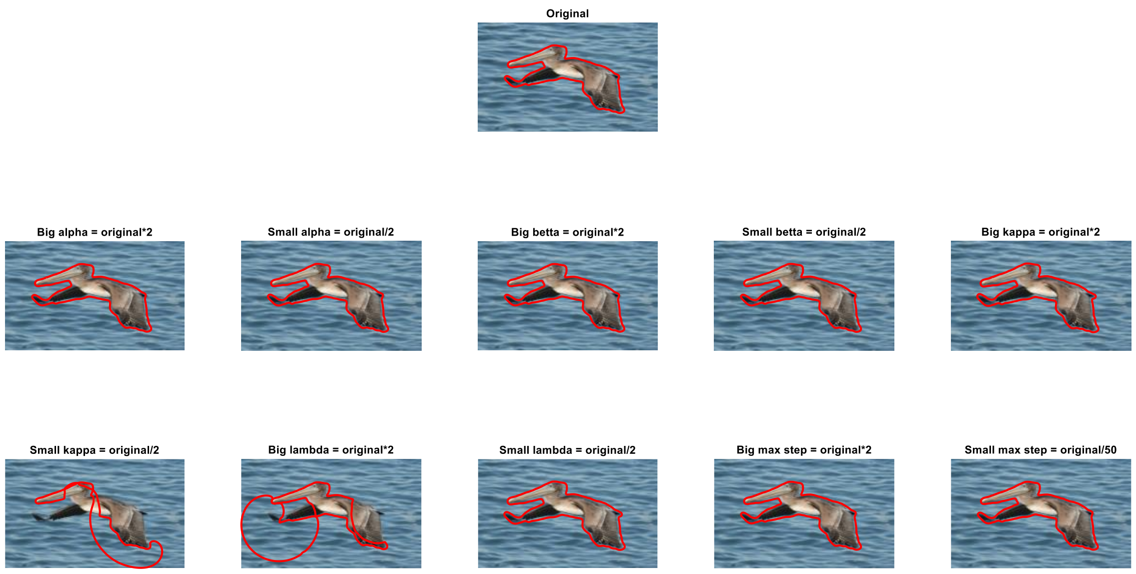


Figure 12: Energy minimization point model (Snakes) critical parameters (duck image)

In this case we have seen that the only 2 parameters that have a hard influence when multiplied or divided by 2 from the optimal are the kappa and the lambda, the same as could be expected from the previous figure. That does not mean that the other parameters do not have an influence on the performance but that this two are the most relevant ones.

	Original	*Salt & Pepper noise*	*Gaussian noise*
Alpha	0.1	0.1	0.2
Betta	0.1	0.1	0.1
Lambda	-0.05	-0.026	-0.03
Kappa	0.3	0.32	0.26
Max step	0.4	0.4	0.4

Table 2: Optimal snake parameters (duck image)

2.1.1. Salt and Pepper noise

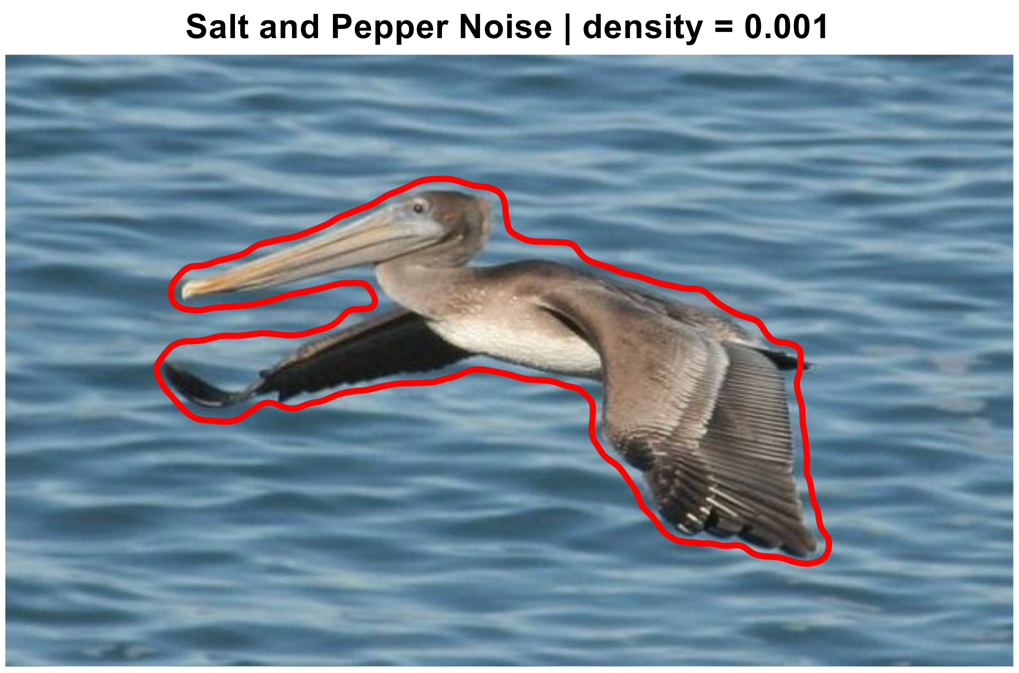


Figure 13: Good segmentation (using snake) with noise || Salt & Pepper (density = 0.001)

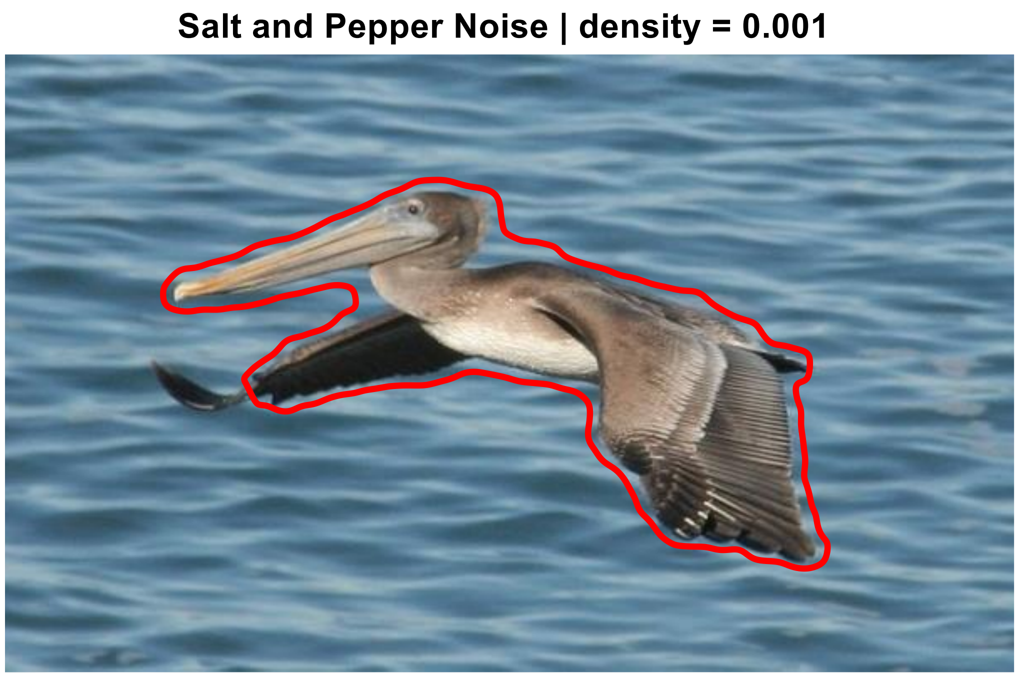


Figure 14: Bad segmentation (using snake) with noise || Salt & Pepper (density = 0.001)

For the salt and pepper noise we have tried different density values and realized that over 0.001 the algorithm was not reliable (see *Figure 15* below). For this exact density we observe that with the parameters defined in *Table 2* we usually reached the image in *Figure 13*. However, depending on the noise configuration sometimes the algorithm was not able to detect the last part of the right wing of the duck (*Figure 14*).

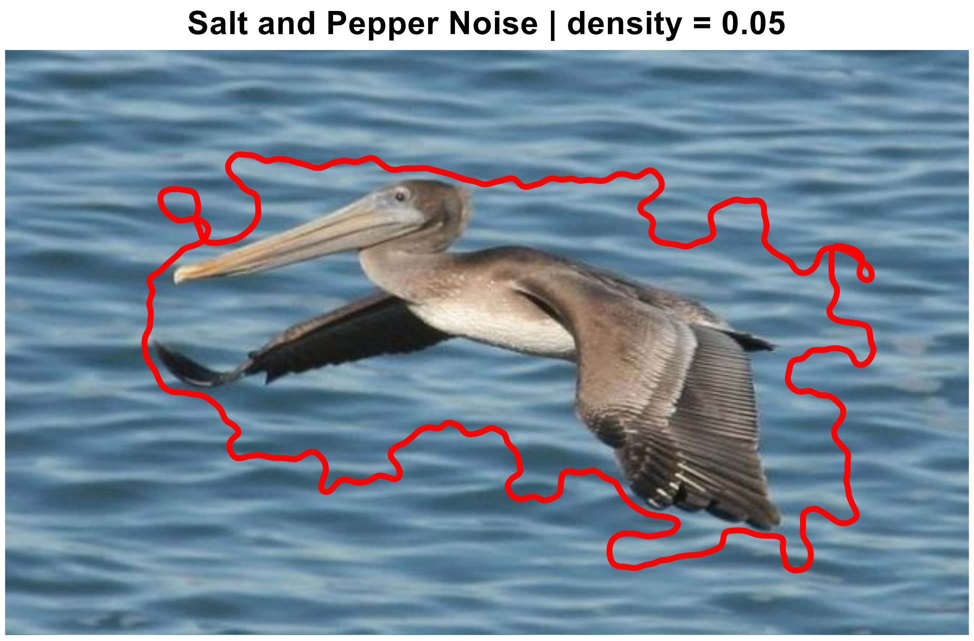


Figure 15: Divergence of the algorithm with noise || Salt & Pepper (density = 0.05)

2.1.2. Gaussian noise

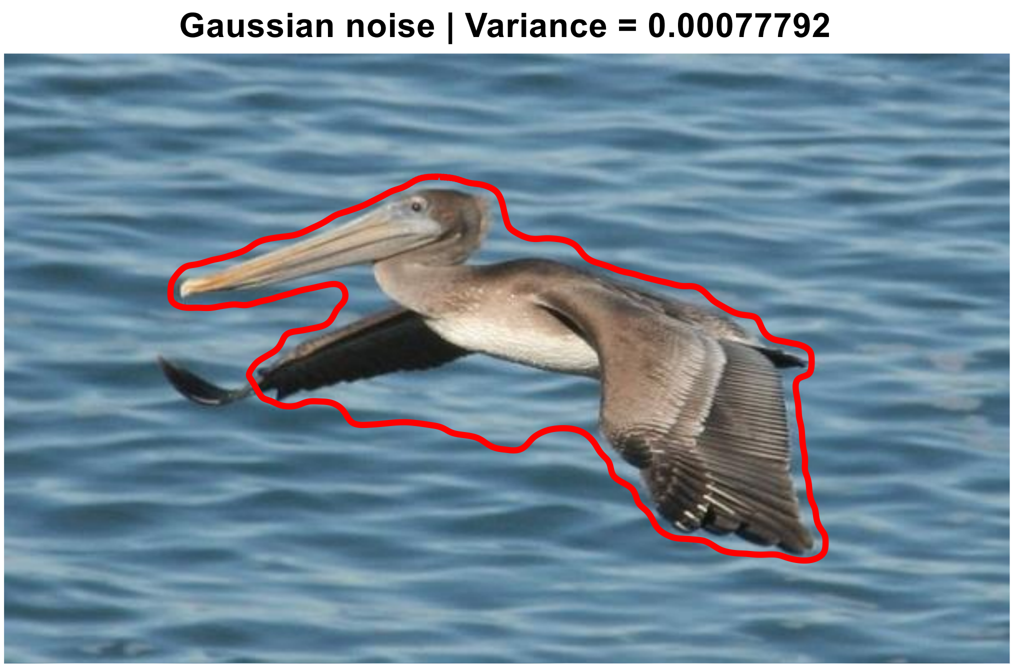


Figure 16: Bad segmentation (using snake) with noise || Gaussian (var = var(image)/20)

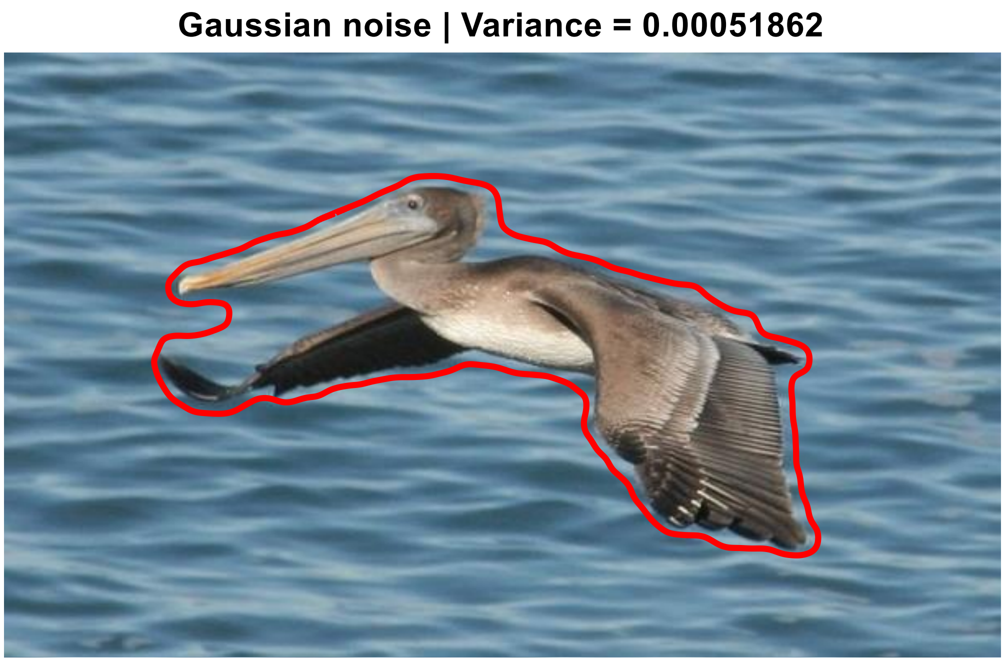


Figure 17: Good segmentation (using snake) with noise || Gaussian (var = var(image)/30)

We have also tried the gaussian noise. For a noise variance 20 times smaller than the image one we could never detect the last part of the right wing of the duck (see *Figure 16*). If we reduced the variance of the noise to 30 times smaller, using the optimal parameters seen in *Table 2* we reached the result of *Figure 17*. Note that now we have increased alpha in order to prevent the contour from performing loops.

2.2. Level sets

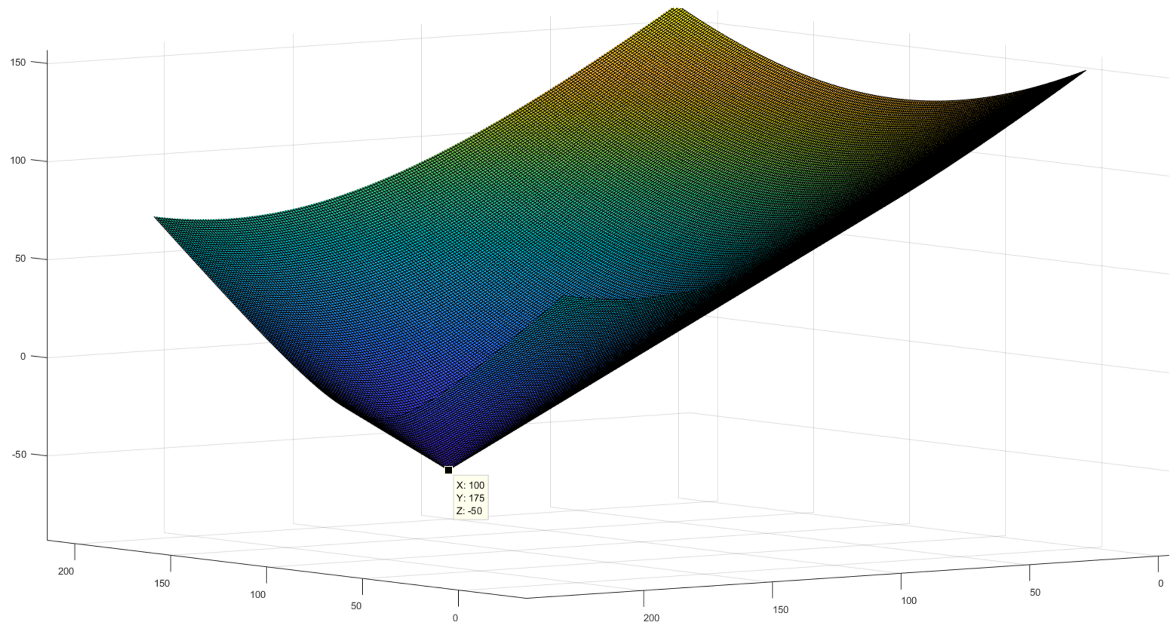


Figure 18: Initial level set model

The initial level set model is a cone with its vertex in (100, 175, -50), see *Figure 18*. Its zero-level set is a circumference of radius 50 centered at the same X and Y as the cone vertex.

Level set parameters:

- mu: Coefficient that regulates the contribution of the curve length to the energy that we want to minimize. Larger mu forces the system to a configuration with lower length.

- Nu: It has a similar effect as mu but, instead of regulate the curve length, regulates the interior region area.
- C1: Expected mean gray level for the inner region. The system will tend to reach a configuration with a gray level similar to this value.
- C2: Expected mean gray level for the outer region. The system will tend to reach a configuration with a gray level similar to this value.
- Lambda1: Coefficient that regulates the contribution of the inner energy to the total energy that we want to minimize. Larger lambda1 forces the system to a configuration with lower inner energy.
- Lambda2: Coefficient that regulates the contribution of the outer energy to the total energy that we want to minimize. Larger lambda2 forces the system to a configuration with lower outer energy.
- Kappa: Coefficient that penalizes the difference between the Laplacian of phi and the divergence of its normalized gradients.
- G: Distance weights, by default is 1.
- Tau: Time step, it's similar to the learning rate in a gradient descent, larger tau makes the system converge faster but can lead to a divergent or unstable system.

In order to try the level sets algorithm, we have executed two different demos, with the following differences:

	Levelset_demo	*Levelset_demo2*
Initial model vertex	(128, 112, -5)	(128, 85.3, -5)
C1 (Expected inside gray level)	30	100
C2 (Expected outside gray level)	100	30

Table 3: Differences between level set demos

The differences in the results can be seen in figures from 19 to 23.

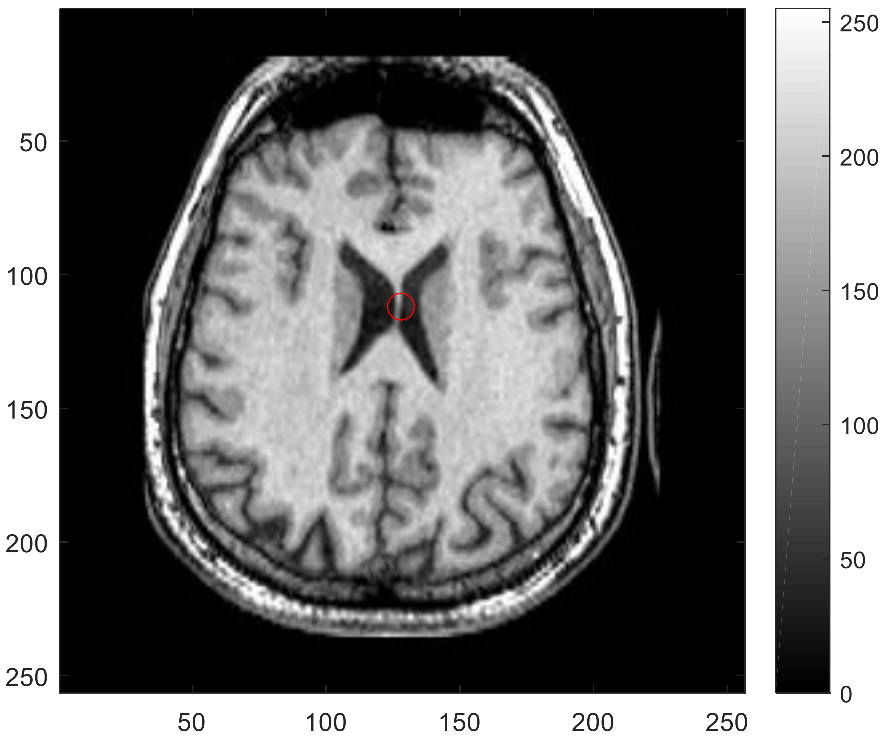


Figure 19: Level set demo 1: Initial zero level set

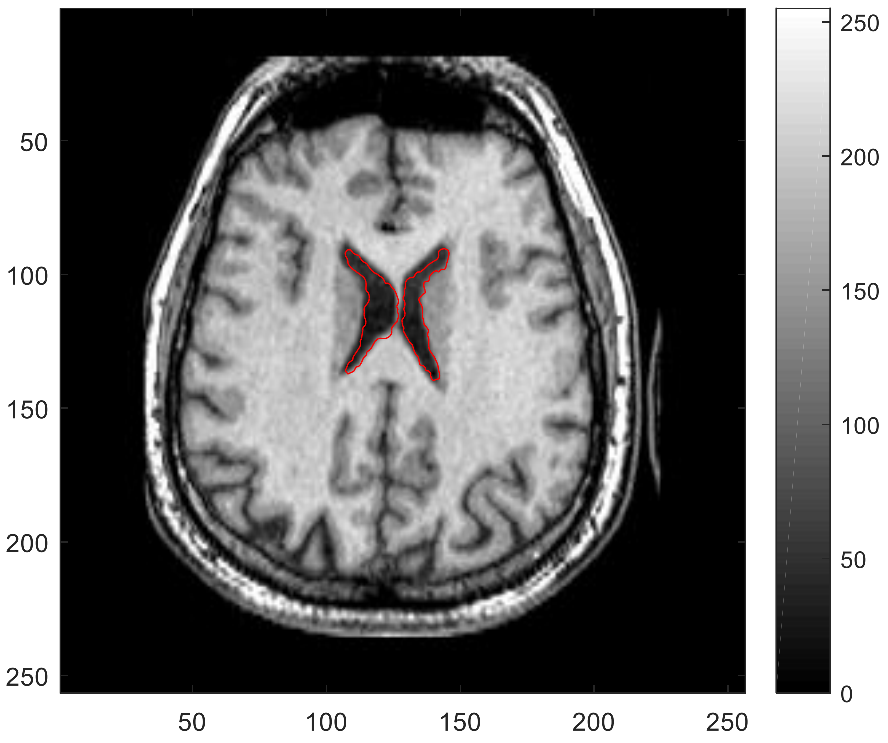


Figure 20: Level set demo 1: Result

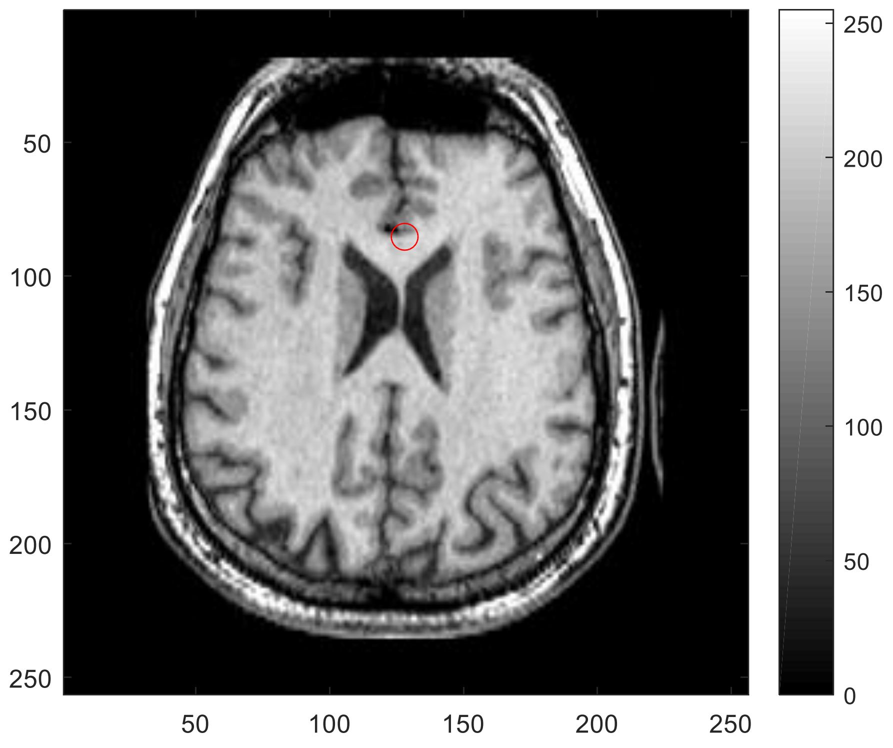


Figure 21: Level set demo 2: Initial zero level set

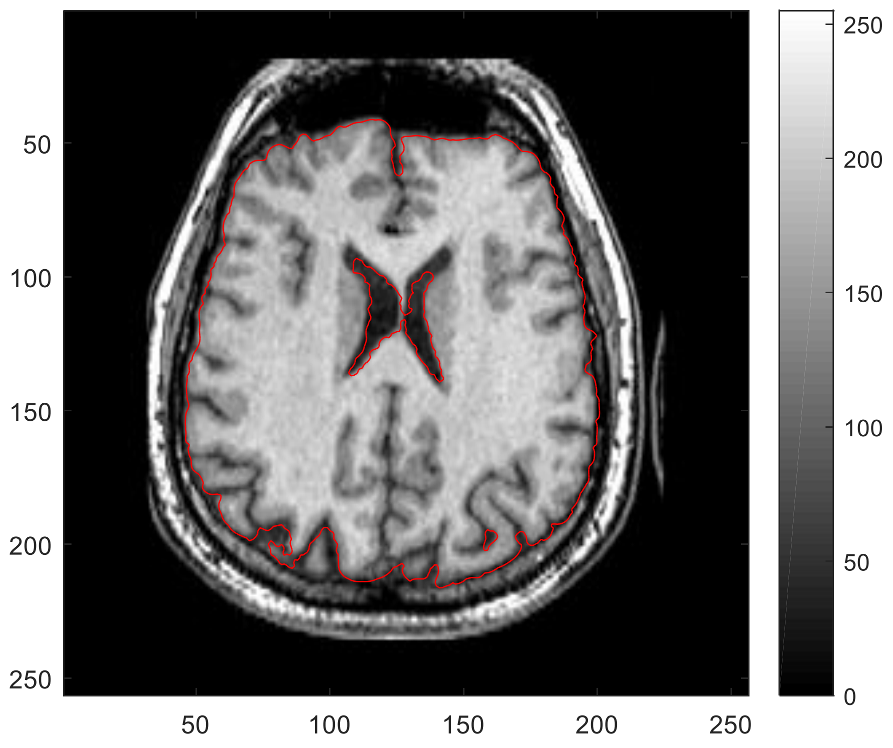


Figure 22: Level set demo 2: Result

The main difference between both demos is that in the first one we start from a dark center and we set an expected gray level of the area inside the zero-level contour much darker than the gray level outside so it only captures the butterfly-shape figure seen in *Figure 20*.

On the other hand, in the second demo we do it the other way around, we start from a lighter initial zero-level set and seek an inner region lighter than the outer, so we reach the result in *Figure 22* where all the brain is captured.

2.3. Techniques comparison

In order to compare both top-down segmentation techniques we took an image of spaghetti with tomato sauce on it and applied them.

2.3.1. Snake

For the snake technique we tried to different approaches:

	Approach 1	*Approach 2*
Alpha	0.1	0.08
Betta	0.01	0.5
Lambda	-0.08	0.08
Max step	0.4	0.4
Kappa	-0.25	0.3

Table 4: Snake parameters for both approaches

The first approach consisted on starting from a big balloon and compressing it.

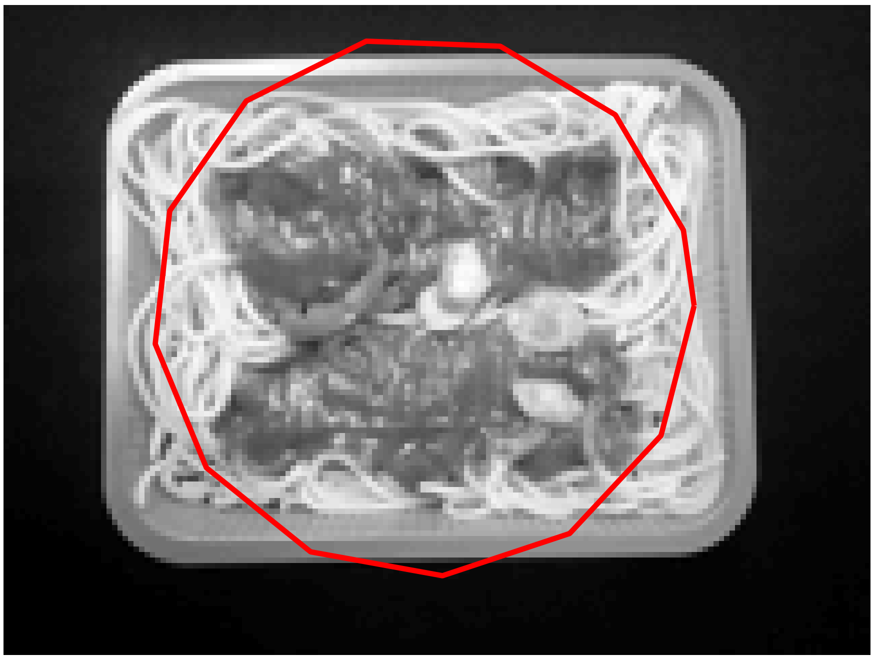


Figure 23: Snake technique (approach 1) | Initial balloon

With this approach and the parameters from *Table 4* we reached the result in *Figure 24*. (Notice that the kappa value is negative because we want our balloon to stop in dark regions for which the image energy will be higher for this kappa and keep compressing through light ones.

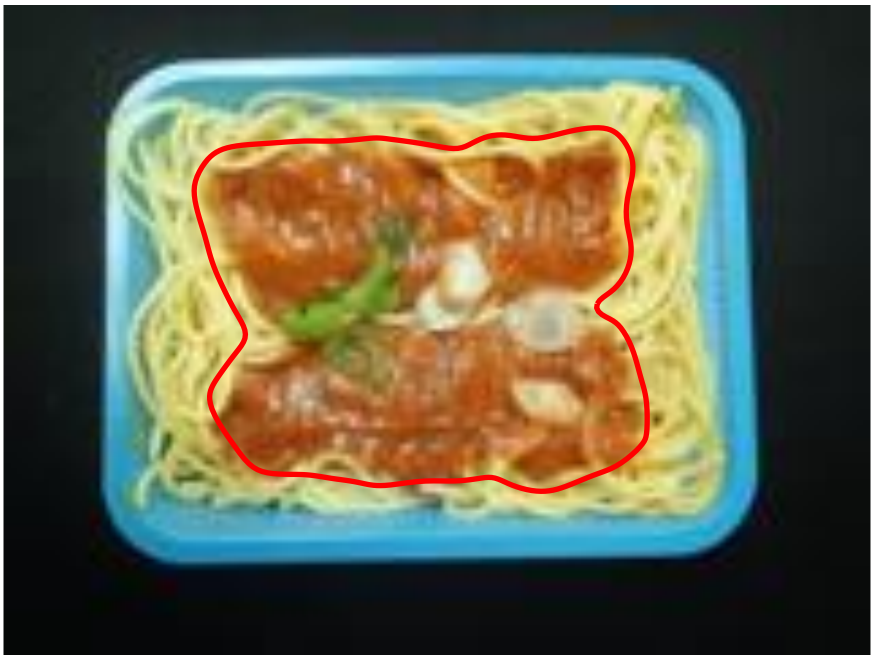


Figure 24: Snake technique (approach 1) | Result

For the second approach we tried to start from a balloon inside and expand it (see *Figure 25*)

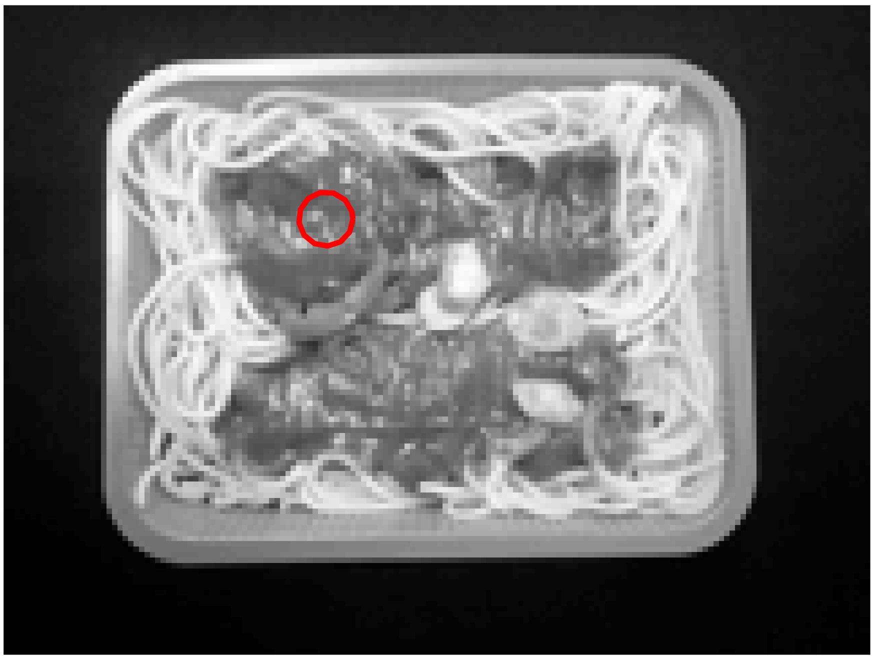


Figure 25: Snake technique (approach 2) | Initial balloon

As we can see in *Figure 26* we could not reach a good segmentation starting from an inner balloon.

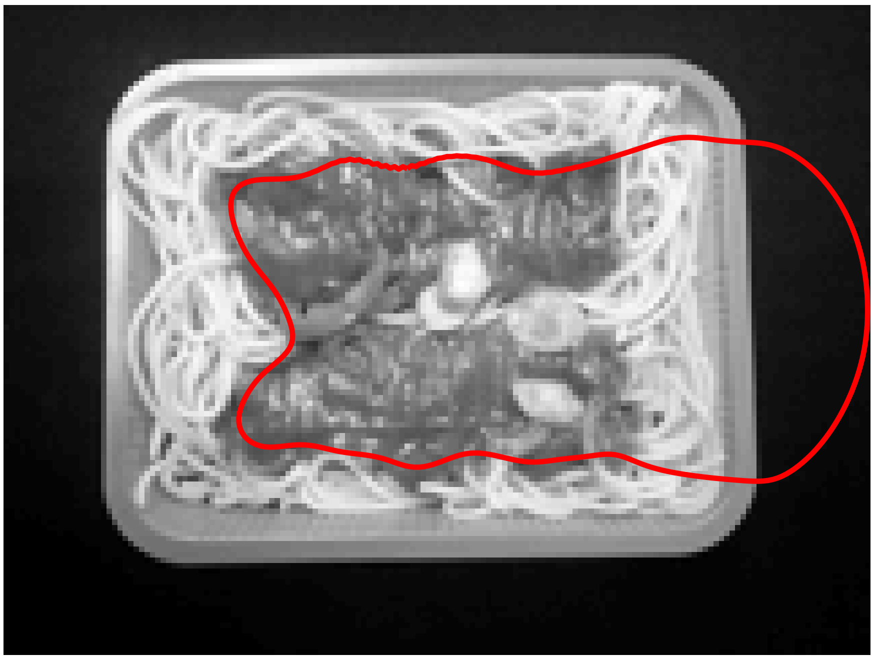


Figure 26: Snake technique (approach 2) | Result

2.3.2. Level sets

For the level sets we tried one unique approach which we considered that would be the best one, which has the initial zero level set seen in *Figure 27*:

	Values
Mu	500
Nu	0
C1	100
C2	200
Lambda1	0.5
Lambda2	0.5
Kappa	0.05
G	Default
Tau	1

Table 5: Level sets parameters

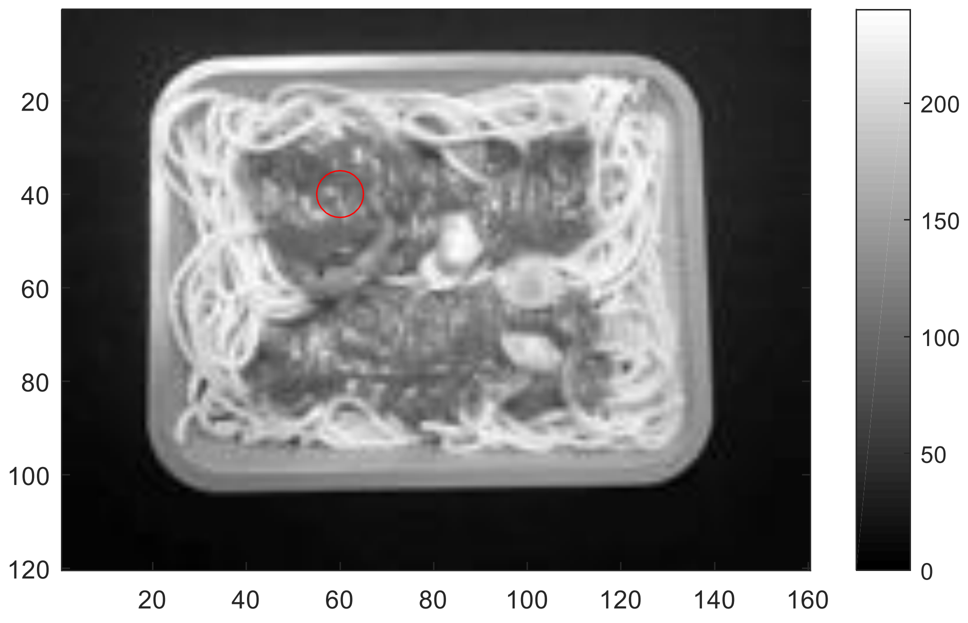


Figure 27: Level sets technique | Initial zero-level set

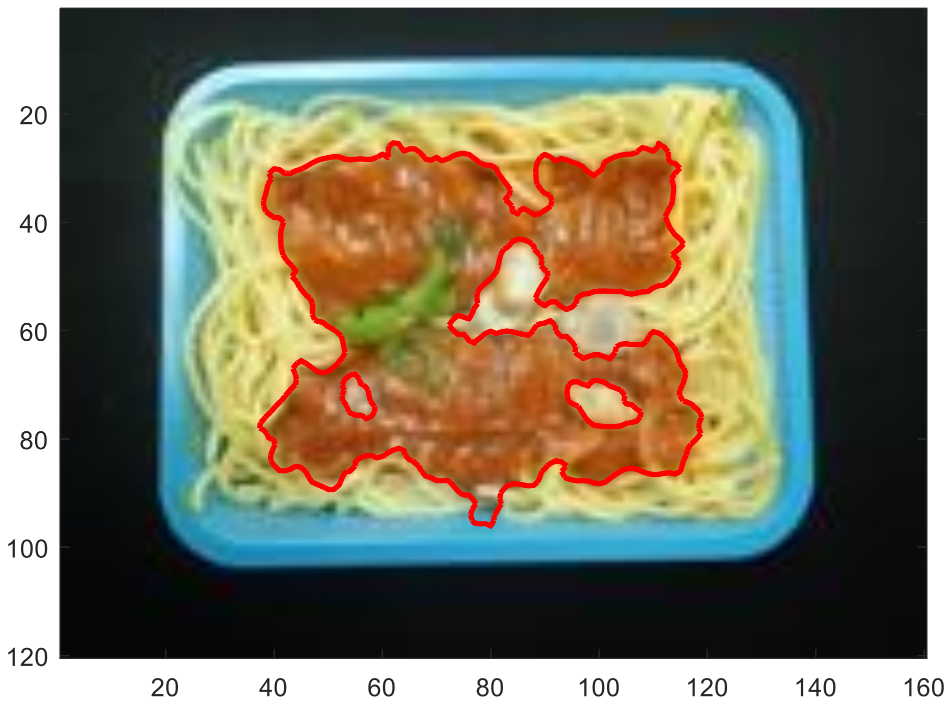


Figure 28: Level sets technique | Result

As we can observe in *Figure 28*, and as we expected this configuration is the best one and perfectly separates the pasta from the tomato. In this example we can conclude that the level sets technique performs far better than the snake one.