Computer vision Lab 8

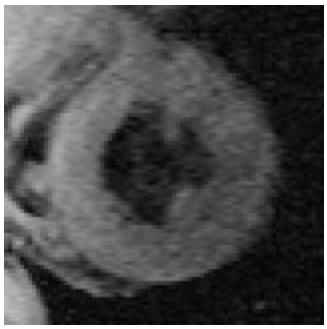
Image Segmentation

Active contour models

Active contour models, also known as snakes or deformable contours, are a top-down approach to object segmentation. The technique aims to fit a contour to the boundary of an object, given an initial contour

near the desired object. In the first part of this report, an active contour model is fit to the image of a heart (see Fig.1).

Fig.1 - Heart image



The segmentation process was implemented in Matlab, using a function called “snake” which can be found with the Matlab files attached to this report, in the “snake.m” file.

Function Parameters

The segmentation function has several parameters which can be used to tweak how the contour model is fitted to the image. The key parameters, their descriptions and their initial values set for the analysis of the heart image are summarised in Table.1.

Table.1 - Snake function parameters

Parameter	***Description***	***Initial value***
X, Y	*2D Position vector defining a closed contour shape where the contour will be initialized.*	*13 point vector defined by t in the interval from 0 to 2π with steps of 0.5. Then:* $x = 70 + 3 * cos(t)$ $y = 90 + 3 * sin(t)$
Alpha	*Controls the elasticity as the contour is tightened.*	*0.1*
Beta	*Controls the rigidity of the contour.*	*0.01*
Kappa modifier	*Weights the contribution of the external force that stops the snake once it arrives at the desired boundary*	$Kappa = \frac{0.2}{maximum(\lvert\Delta(image)\rvert)}$
Lambda	*Weights the contribution of the balloon force. Positive values make the snake expand.*	*0.05*

Fx,fy	*External force field, normalized to maximum magnitude 1.*	$[px, py] = -\Delta(image)$
Maxstep	*Maximum step-size to modify the contour by on each transformation.*	*0.4*

Parameter tweaking

A segmentation function was created ("segmentation.m") in order to tweak the contour model's parameters and plot the effects of the modified parameter. The function works by accepting a modifier for alpha, beta, kappa, lambda and maxstep (the other parameters were not varied). The function would also be given a function handler to set the segmentation arguments to their initial value, which would allow for individual tweaking of each parameter. The segmentation function would then perform the active contour segmentation once with the initial values, and then 5 times for each variable parameter, only modifying one parameter at a time. The range of a parameter's value was calculated as follows:

Eq.1 - Parameter tweak range

$$modifiedParam = paramValue * paramModifier$$
$$rangeStep = modifiedParam$$
$$minValue = modifiedParam - 2 * rangeStep$$
$$maxValue = modifiedParam + 2 * rangeStep$$

This method allowed not only for efficient semi-automatic tweaking of parameters, but also would consistently produce 5 plots for each modified parameter, unless an out of range error was produced, making it simple to visualise using Matlab's plotting tools.

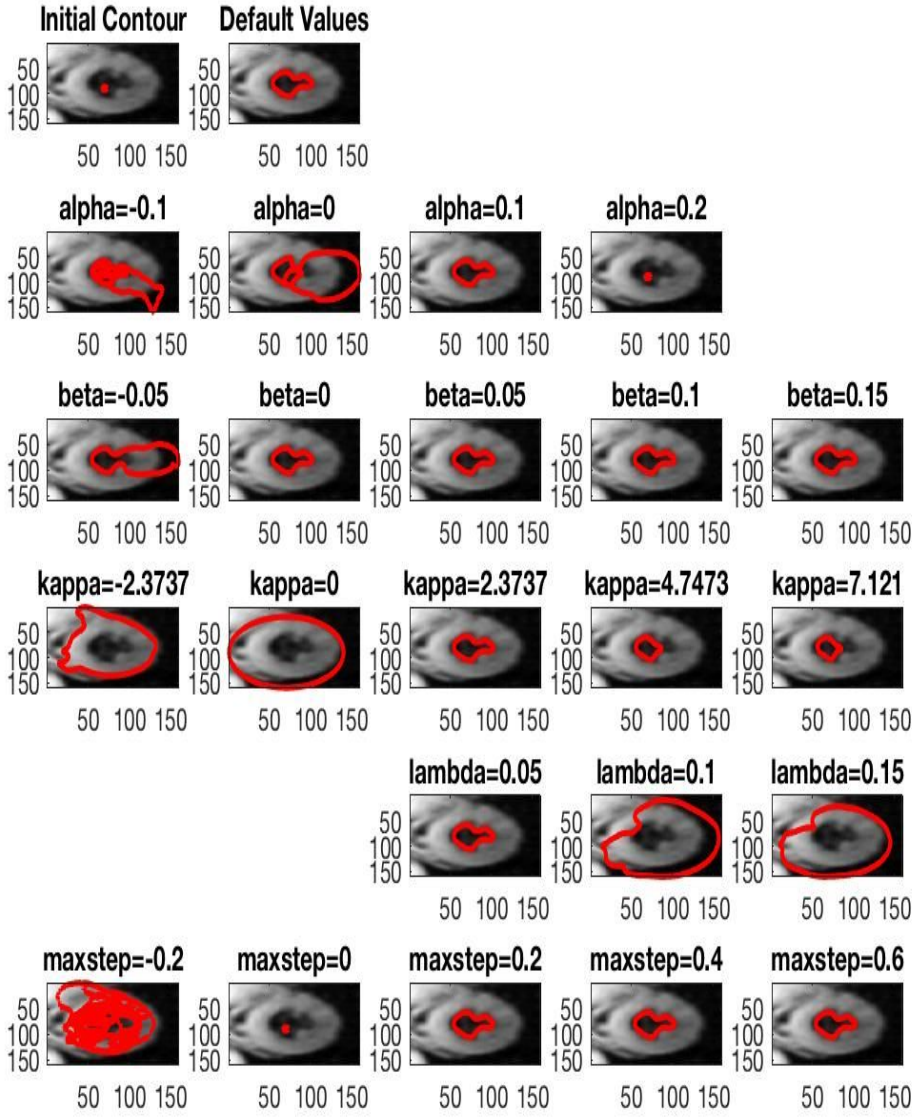
Func.1 - Segmentation function call

```
function fig = segmentation(img, alphaTweak, betaTweak, kappaTweak,
lambdaTweak, maxstepTweak, loadInitialValues)
```

Heart Segmentation

Fig.2 shows the application of the active contour segmentation to the image of the heart. The first row shows the initial and final contours for the segmentation with initial values. The other rows only show the final contours, as the initial contour position was not modified. Each row contains the final contours for the modification of each parameter, and in the title of each cell the modification performed is shown. Where there is no image, it means that the contour function could not perform because the modified parameter was outside of the acceptable range for this segmentation model.

Fig.2 - Parameter tweaking contours



It is observed that by using the default values, an optimal fit is already found.

When modifying **alpha**, it was observed that a low alpha (<=0) resulted in a bad contour fit. The contours overflowed from the object that was suppose to be segmented. As alpha is increased, it approximates the object better with an optimal value of 0.1 as far as can be observed using the custom range. When alpha is too high (>=0.2) it no longer retains the shape of the segmented object and the contour collapses into a point. This is because alpha modifies the elasticity of the contour, which contributes to its internal energy. Therefore as alpha is decreased, the elasticity and thus internal energy of the contour decrease, meaning that it is easier to stretch and it then overflows from the wanted segmentation. On the contrary, when alpha is a high value, the contour has a high internal energy and retains its shape more strongly, meaning that it is difficult to stretch and will collapse into a point if force not strong enough are used to expand the contour.

Eq.2

$$Einternal = \alpha \left|\frac{dv}{ds}\right|^2 + \beta \left|\frac{d^2v}{ds^2}\right|^2$$

As the **beta** parameter was modified, it was observed that a low beta (<=-0.05) caused a similar overflowing of the contour as when alpha was a low value. However, unlike alpha, when beta was increased (>=0), the contour did not collapse in on itself. In fact, all the beta values above 0 were a good fit, and although similar, a small change is observed between beta=0 and beta=0.15. It is just that the contour does not seem to be as sensitive to an increase in beta as it is to alpha. This could be explained by the fact that the range did not go high enough for beta, however from beta=0 to beta=0.15 the final contour produced shows little change. A more likely explanation can be deduced from Eq.2; the fact that beta modifies the second derivative of the contour points on the curve, not the first, which means that a change in the contour position will be much more sensitive to a change in alpha than beta. In fact, this can be observed for the lower values of alpha and beta, where the final contour of a low alpha value has overstretched more than a similar beta value, for example when alpha=0 & beta=0.

The **kappa** parameter produced an oval shape at kappa=0. A small kappa (<=-2.37) gave a deformed oval, which encompassed too much of the wanted segmentation. A kappa=2.37 gave an optimal fit to the heart. Then for kappa values larger (>=4.74), it was observed that the contour would shrink smaller than the desired segment. This is because kappa modifies the external energy of the contour expansion. The larger kappa is, the more resistance against the internal energy there is, and the less the active contour can expand. This is why when kappa=0, kappa's contribution to the external force is reduced to 0, and therefore the contour could keep on expanding until it was an oval. As well as this it explains that when kappa increases, the external energy also increases, meaning that the active contour is harder to expand, leading to the small final contours seen for higher values of kappa.

The **lambda** parameter did not produce any images when it was a negative value. Its fit at lambda=0.05, was the best and when it was increased (lambda >= 0.1) it overflowed from the wanted segmentation. The reason for this is that lambda represents the contribution of the balloon force (i.e. the force that causes the active contour to expand). The greater this force, the larger the external energy would have to be to stop

the contour from expanded. This is why it is seen that as lambda increases, the contour shape also increases, and also becomes more smooth (like an inflated balloon).

When **maxstep** parameter was low (<=-0.2), the active contour was very long and un-smooth, similar to a long piece of string dropped on a surface. As maxstep was increased, it was observed to first contract (at maxstep=0), and then increase to fit the heart similarly to when using the default values. No change was observed to the final active contour when maxstep was increased (>=0.2). Maxstep represents the size of the step that the contour is modified upon each transformation. Therefore, as long as the step is small enough, the correct contour will be achieved, although perhaps over more iterations. The observation for a negative max step is an anomaly of the function. Perhaps a better way of dealing with a negative max step would be to set it as the default value of 0.4 when it is found to be negative.

Active contour models in noisy images

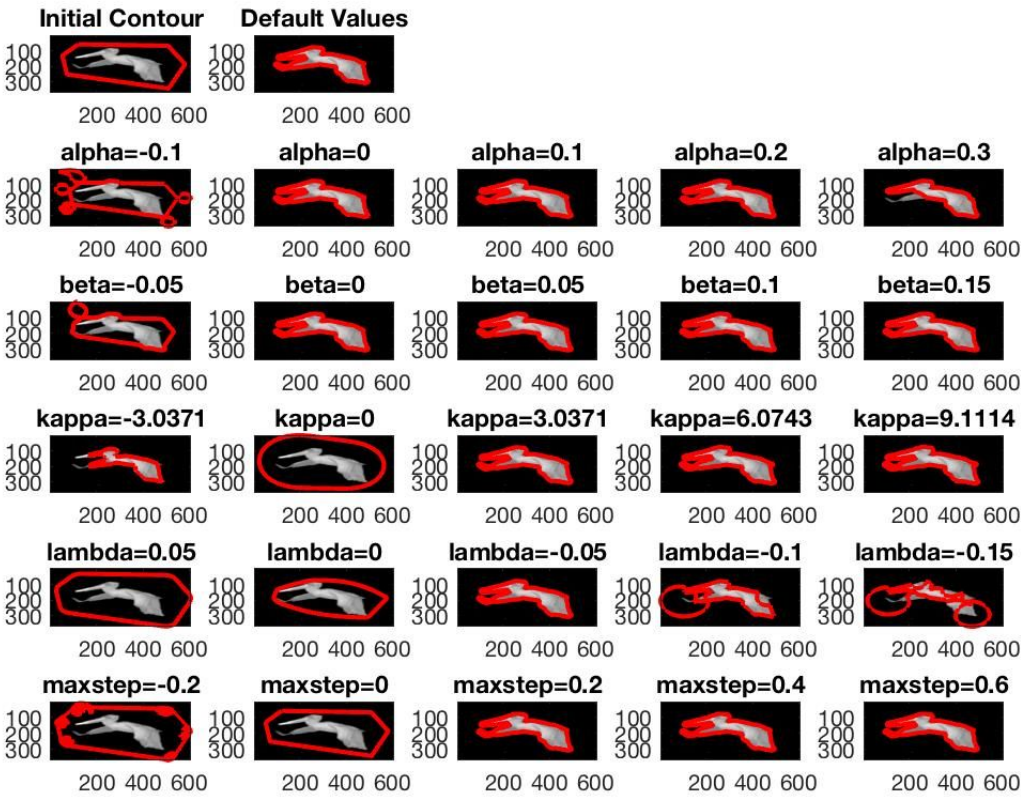
Bird segmentation

After the segmentation technique had been applied to the heart image and the optimal parameters found, the segmentation parameter was applied to the image of a flying bird (see Fig.3). However, this time the aim was to add noise to the image, and see if the object could still be segmented.

Fig.3 - Bird image

First of all, the same technique as before was used, but with different initial values to the heart. The only different initial values were: $Kappa = \frac{0.3}{maximum(|\Delta(image)|)}$; lambda=-0.05. The negative value of lambda means that the contour shrinks, instead of expanding. First of all, the segmentation was performed without adding any noise.

Fig.4 - Bird segmentation no noise



It can be observed that the default values again give an optimal fit to the image. Once the plain segmentation had been performed, noise was added to the image before segmentation, using Matlab's *imnoise* function. First of all using the noise type "salt & pepper", which varies pixels on and off. A density of 0.0001 for the noise was used.

Func.2 - Applying salt & pepper noise to image with matlab

```
img = imnoise(img, 'salt & pepper', 0.0001);
```

When the segmentation was performed with the same default values but to the bird image with salt & pepper noise, it was observed that they were no longer an optimal fit (see Fig.5).

Fig.5 - Bird segmentation salt & pepper noise

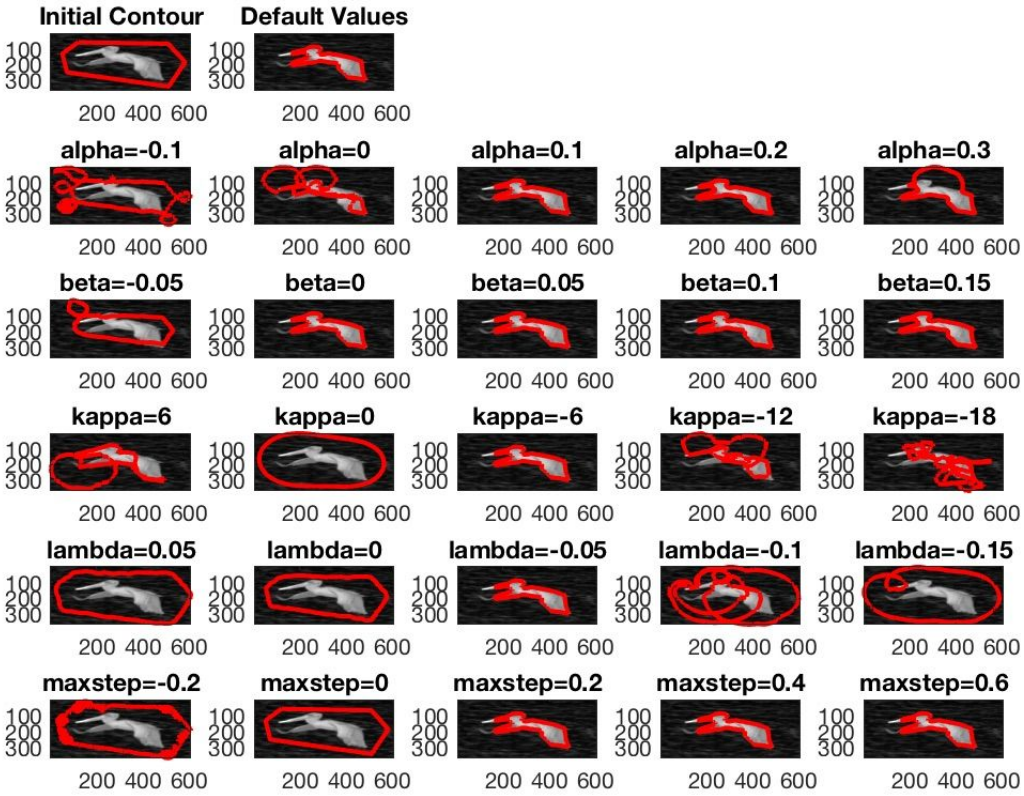
However, it is observed that the best fit from the parameter variation was when kappa=-7.8416. Also note that the contour fit expands when kappa is 0. Therefore as a compromise this parameter value was selected and the segmentation performed again with initial kappa=-6.

Salt & pepper adjusted kappa

From the new segmentation analysis (see Fig.6), it can be seen that none of the images are as optimal segmentation fits as when there was no noise. However, the fit is better than before, and it can be

observed that, for example when alpha=0.1, the bird is almost optimally segmented, except for the tip of its beak and its right wing.

Fig.6 - Active contour segmentation Salt & Pepper adjusted kappa



Having identified the (nearly) optimal parameters for the segmentation with the S&P noise applied to the bird image, it is clear that a better fit could be achieved, as the beak of the bird and the tip of the far wing of the bird are missed by the contour. This was difficult to achieve because, as can be seen by the heatmap of the gradient of the bird image (Fig.7), there is not much of a gradient difference between the tip of the wing and the background. Also, the extension of the beak means that the contour would have to stretch abnormally far. Perhaps a separate technique would have fared better in the segmentation of the beak, as other techniques, such as the level set approach, are not restricted by fact that the closed contour shape is causing the contour points to attract each other. The problem here was that in order to segment the beak using the snake, a high internal energy is required. This high energy is difficult to balance without the active contour overflowing into other areas of the image.

Fig.7 - Heat Map of bird image gradient

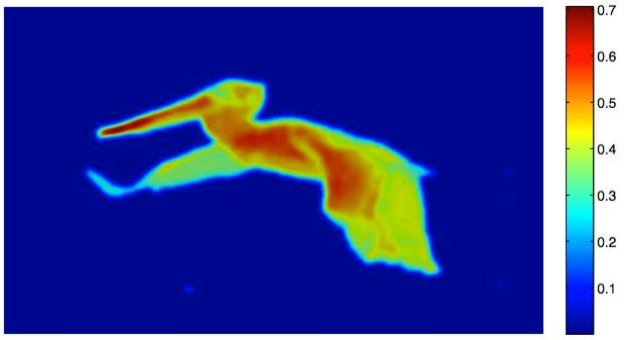
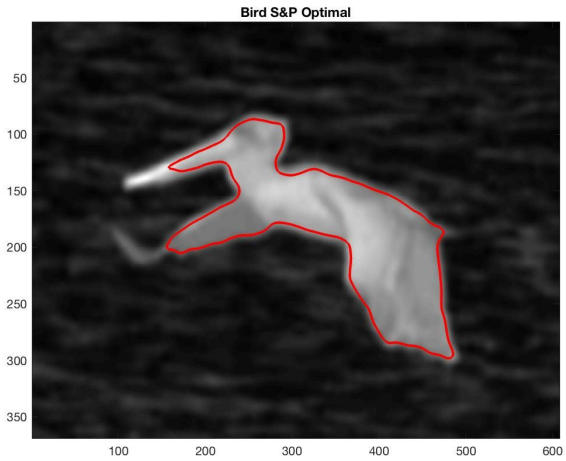


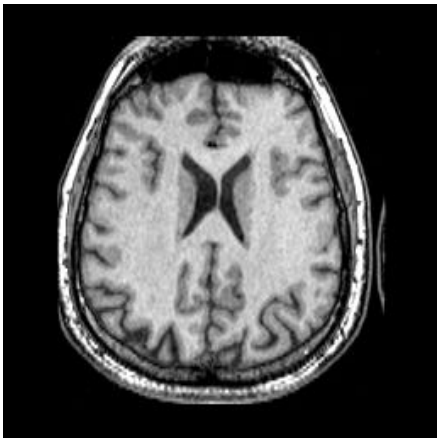
Fig.8 - Near optimal S&P bird segmentation



Exploring segmentation by level sets

Although the active contour method was simple to tweak and useful for segmenting the heart and the bird, it becomes problematic when attempting to segment an image that is split into more than one segment, or for extended segments, for example the beak of the bird in the previous example. The next example attempts to segment the ventricles seen in a brain scan, as seen in Fig.9.

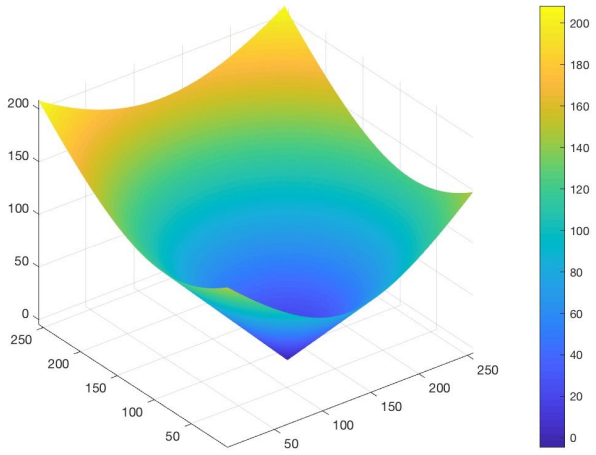
Fig.9 - Image of a brain with the ventricles seen as dark segments in the centre.



For this sort of segmentation, a “level sets” segmentation approach can be used.

In the file accompanying this report, “levelset_demo.m”, level sets segmentation is performed on the image of the brain. The initial levelset_model used by the demo is that of a binary mask, producing a cone-like shape as seen in Fig.10.

Fig.10 - Initial level-set function (f) used for the brain image in levelset_demo.m



To perform the level-set, the function f (also known as phi ϕ), is passed to the level-set function defined in the “levelset.m” Matlab script, along with other model parameters as seen in Func.3.

Func.3 - Level-set function from “levelset.m” file

```
function f = levelset(f,im,mu,nu,c1,c2,lambda1,lambda2,kappa,g,tau)
```

Level-set Parameters

Parameters	*Description*
f	*Initial value of the level set function phi, which should be negative inside, zero on the boundary and positive outside. The initial value of phi does not have to be a signed distance function but the convergence might be more difficult if it deviates from it too much.*
im	*Image being segmented.*
mu	*Coefficient mu for the curve length part of the criterion.*

nu	*Coefficient nu for the area part of the criterion. It plays the role of the balloon force. A positive (negative) nu makes the contour shrink (grow).*
c1, c2	*Expected intensities c_1, c_2 of the inside (outside) of the object to be segmented for the Chan-vese intensity criterion*
lambda1, lambda2	*Coefficients lambda_1, lambda_2 for the Chan-vese intensity criterion*
kappa	*Coefficient kappa to penalize deviation of phi from the signed distance function.*
g	*Distance metric weight g. Defaults to constant 1 everywhere.*
tau	*Time step tau. A larger tau can often be used to accelerate the convergence.*

Levelset demo

When the "levelset_demo.m" is run the final segmentation can be observed to be in two separate shapes, something that could not be achieved with the snake approach (see Fig.11). It can be seen that the two ventricles of the brain are well segmented. Apart from the parameter of the initial level-set function, shown in Fig.10, the level-set demo used the following configuration of the levelset function:

- mu=500
- nu=0
- c1=30
- c2=100
- lambda1=0.5
- lambda2=0.5
- kappa=0.05
- g=matrix_of_ones
- tau=1

A second demo was also performed with c1 and c2 swapped around. The results can be seen in Fig.12.

Fig.11 - Final segmentation from “levelset_demo.m”

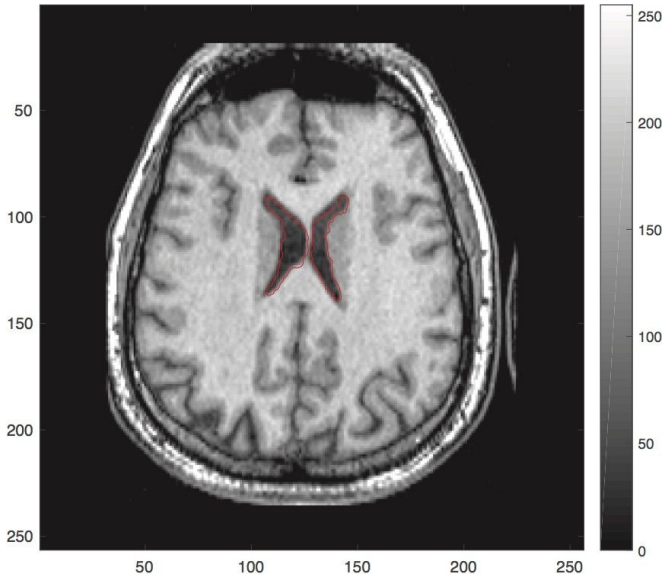
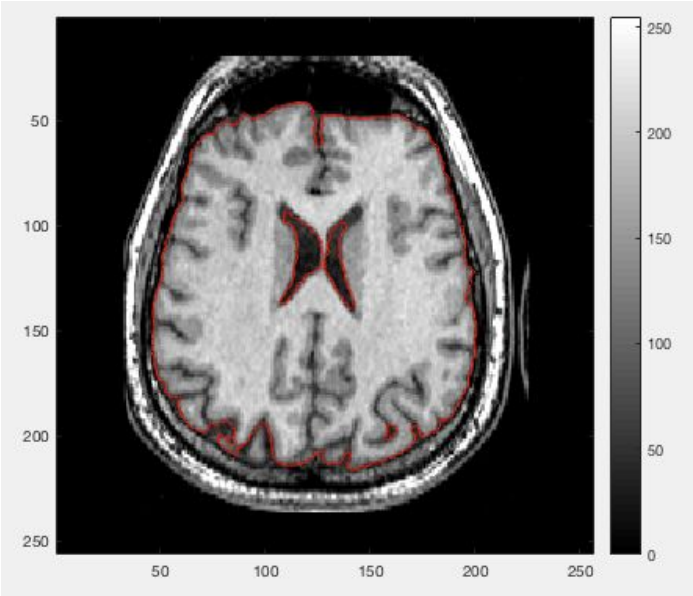


Fig.12 - Final segmentation from “levelset_demo2.m”



It can be observed that in the second demo, the swapping of C1 and C2 has caused the segmentation to segment the whole brain and exclude the ventricles, whereas in the first demo, only the ventricles were segmented.

Application of segmentation by level sets for food analysis

The last segmentation analysis was performed on the image of a spaghetti with sauce on it (see Fig.13). A level-set segmentation approach was used, which can be found in “segmentation_food.m” script.

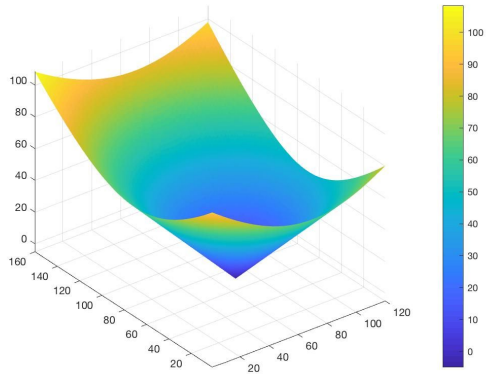
Food Level-set Segmentation

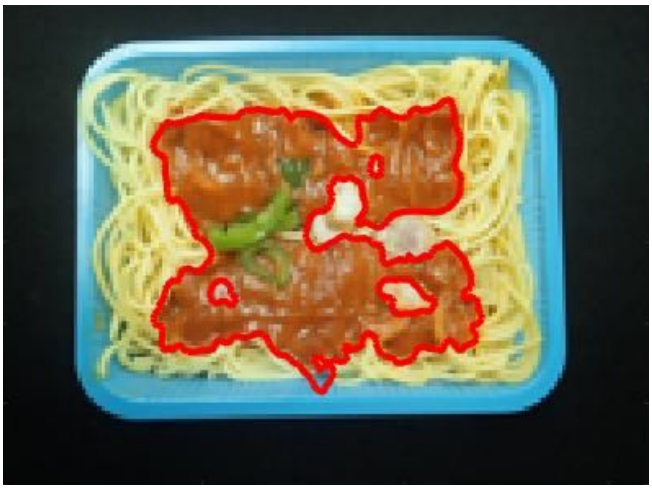
The first parameters found to generate optimal segmentation of the image were the following:

- mu=50
- nu=0
- c1=100
- c2=200
- lambda1=0.5
- lambda2=0.5
- kappa=0.05
- g=matrix_of_ones
- tau=1

The results of the level-set segmentation function can be seen in Fig.16.

Fig.13-16 - Original image (top-left), initial level-set function (top-right), initial contour position (bottom-left), final contour (bottom-right)



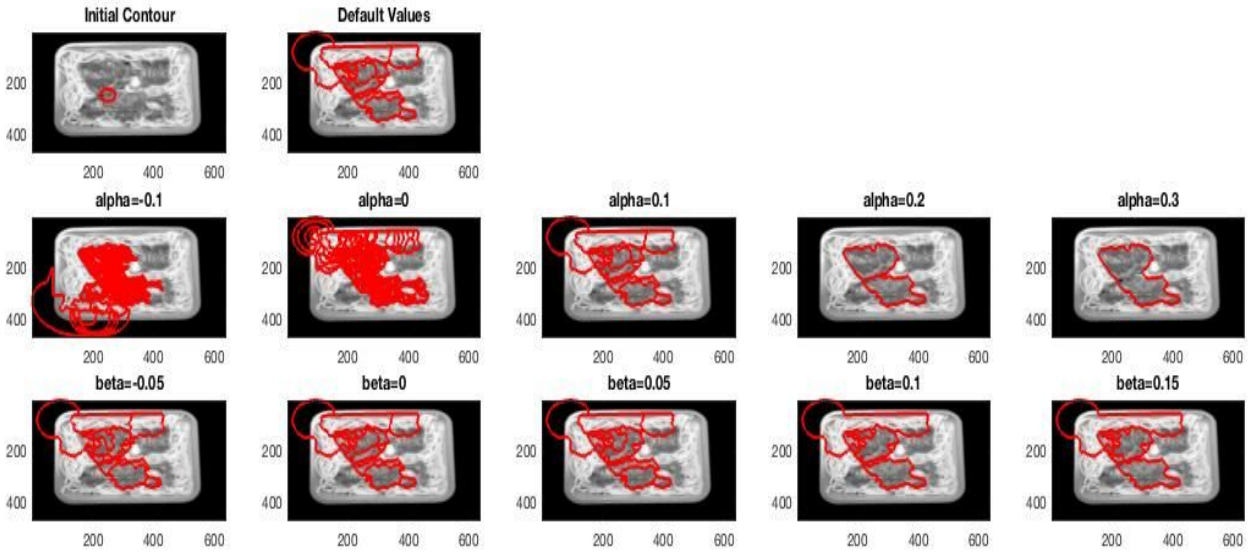


It can be observed that the segmentation of the sauce is good, and even allows for empty boundaries within the sauce to be discarded. This allowed for a segmentation of the image that could not be achieved with an active contour segmentation. However, for comparison purposes it was attempted.

Food Segmentation with Snake

The snake segmentation was also used to attempt to segment the tomato sauce in the pasta. As initial parameters, the same were used as in the initial heart problem at the beginning of this report, but with the initial contour adjusted to lie within the pasta sauce and the kappa modifier adjusted to 0.4. The results of tweaking the parameters with the snake segmentation image are shown in Fig.17.

Fig.17 - Attempted segmentation with snake on food image



It can be observed from Fig.17 that the snake approach struggles to segment the sauce in the pasta. The best result is observed for alpha=0.2 and alpha=0.3. However, from the segmentation it can be seen that the snake approach can not deal with an object which is split into several segments easily.

The snake segmentation approach is simple to use and useful when the object can be approximated in a single area, allowing it to be surrounded by, or expanded into by a rubber band. However. As has been observed when the object is split into several areas the snake approach can not compete with the level-sets approach.

The level-sets approach is more complex to initialize as it requires the initial set-level function, which is used as a binary mask of the interior region of the curve of a segment. However, the approach allows topologically invariant segmentation which performs well at segmenting distributed objects or an object which extends abnormally far in a particular direction.