Part II

Image analysis with geometric priors

IN IMAGE ANALYSIS, the terms *prior knowledge* and *contextual information* refer to all the information about the problem that is available *in addition* to the image data. This additional information may refer to some local appearance (how bright, dark, smooth, textured... something is) or to some geometric property (position, size, orientation, shape... of this something).

There are numerous ways of using prior information when solving image analysis problems, and, in this part of the lecture note, we look into three well-established approaches. First, we cover Markov random fields (MRF) where local contextual information is part of a probabilistic framework that can be efficiently optimized using graph cuts. Second, we present a segmentation model based on a parametric deformable curve, and for this, we introduce Mumford-Shah functional, Chan-Vese algorithm, and snakes. Lastly, we cover layered surface detection, a model useful when dealing with distinctive geometry.

The three approaches covered in this part of the note have many common points and may be combined in different ways. For example, layered surfaces use the same graph-cut solver as used in MRF. And curve-based segmentation can be driven by layered surfaces.

5 Markov random fields

Markov random fields (MRF) is a model that can be used for various tasks in computer vision and image analysis. All MRF formulations involve labeling, i.e. assigning labels to some entities, called *sites*. Typically, we are assigning labels to image pixels. MRFs are characterized by the Markov property, i.e. that the probability of a site (pixel) being assigned a certain label is only dependent on the neighborhood of the site.

In exercises on MRF, we will use the MRF model for image segmentation. Segmentation can be formulated as assigning a (discrete) label to each pixel in the image. Often, we would like the segmentation to be smooth. Using MRFs we can incorporate smoothness as the local contextual information into the segmentation model. We do this by giving a low probability for a configuration where many neighboring pixels have different labels. To do so we add a term, *a prior*, next to the usual *data-term*, which in this context is also called *a likelihood term*. Provided an image, we aim at finding a label configuration that maximizes the *a posterior* (MAP) probability which is a combination of a likelihood (data) term and the term modeling a smoothness prior.

One characteristics of MRF, *Markov-Gibbs equivalence* following directly from Markov property, is that the probability of the MRF configuration is an exponential of the negative configuration energy. It is more practical to work with energies, as energy contributions add up, while probabilities multiply. Therefore, instead of maximizing the posterior probability we will minimize the posterior energy of the configuration f given by

$$E(f) = U(f|d) = U(d|f) + U(f)\,, \tag{5.1}$$

where $E(f)$ is a (segmentation) energy of the configuration f for a certain data (image) d. Here, $U(f|d)$ is a posterior energy and $U(d|f)$ is a likelihood energy.

Another important property of MRF is that both likelihood and prior may be expressed as sums of local contributions called *potentials*. In the exercise, we will work with so-called *one-clique potentials* (single pixels) and *two-clique potentials* (pixel pairs). In terms of clique potentials (5.1)

becomes

$$E(f) = \sum_{\{i\} \in \mathcal{C}_1} V_1(f_i) + \sum_{\{i,j\} \in \mathcal{C}_2} V_2(f_i, f_j)$$

where $\mathcal{C}_1$ is the set of one-cliques (for example single pixels), and V_1 is a one-clique potential used for modeling the likelihood term, $\mathcal{C}_2$ is the set of two-cliques (for example pixel pairs), and V_2 is a two-clique potential used for modeling the prior term.

As such MRF framework allows us to compute a (global) probability of a whole configuration when some (local) property is modeled as a sum of clique potentials.

As discussed in the book by Li[1], Chapter 1, Introduction, first paragraph, the main concerns of the MRF framework are *how to define clique potentials* (modelling part), and *how to find the optimal solution* for a given energy function (optimization part).

[1] Stan Z Li. *Markov random field modeling in image analysis*. Springer Science & Business Media, 2009

5.1 Gender determination, an easy introduction to MRF

We start with the extremely small 1D example with the aim of introducing MRF terminology, demonstrating the modelling possibilities provided by MRF, and the use of the terms *likelihood* or *data term*, *prior* and *posterior*.

Imagine entering a bar and observing 6 persons standing along the counter. You estimate persons heights (in cm) and record this data as

$$d = \begin{bmatrix} 179 & 174 & 182 & 162 & 175 & 165 \end{bmatrix}.$$

You want to estimate the persons gender, i.e. you want to assign either a label M or F to each person. You may consider each person independently according to only its own height, but you also want to utilize your knowledge of the contextual information. You decide to pose the problem as a MRF with the neighbourhood given by the first neighbor (person to the left and person to the right).

But first, let's consider the likelihood (data) term. You know that the average male height is 181 cm, the average female height is 165 cm. You also know that the height for each gender may be described as following a normal distribution where you assume the same standard deviation for both genders. For this reason you define the likelihood terms as one-clique potentials

$$V_1(f_i) = (\mu(f_i) - d_i)^2$$

where d_i is the height of the person i, f_i is a label assigned to the person i (i.e. either M or F), and $\mu(f_i)$ is either $\mu_M = 181$ or $\mu_F = 165$. The likelihood energy of a configuration $f = [f_1 \dots f_6]$ is the sum of all

one-clique potentials

$$U(d|f) = \sum_{i=1}^{6} V_1(f_i)\,.$$

For example, configuration $\begin{bmatrix} M & M & M & M & M & F \end{bmatrix}$ has likelihood energy of 451.

To find the configuration which minimizes the likelihood energy you can consider the one-clique potentials for all i and both labels

$$\begin{array}{lcccccc} (\mu_M - d_i)^2 & : & 4 & 49 & 1 & 361 & 36 & 256 \\ (\mu_F - d_i)^2 & : & 196 & 81 & 289 & 9 & 100 & 0 \end{array}$$

Obviously, the minimal likelihood energy is obtained if we choose a label which minimizes the cost for each i, resulting in a labeling

$$f^D = \begin{bmatrix} M & M & M & F & M & F \end{bmatrix}, \tag{5.2}$$

and giving $U(d|f^D) = 99$. Another thing to notice is that additional cost for deviating from this labeling varies, depending on which label we change. For example, it costs additional 352 to label the forth person as male, while it only costs additional 32 to label the second person as female.

Now we want to incorporate the contextual (prior) information about the gender of the people standing along the bar counter. To make this example similar to modeling smoothness, let's say that you expect a configurations with men stand next to women to to occur less frequently then configurations where genders group. For this reason, you may decide to incorporate a cost which penalizes a less-frequent configuration. For prior energy you therefore define 2-clique potentials as

$$V_2(f_i, f_{i'}) = \begin{cases} 0 & \text{if } f_i = f_{i'} \\ 100 & \text{otherwise} \end{cases}.$$

The prior energy is the sum of all 2-clique potentials for all 2-cliques (all pairs of neighbors) in a configuration. Utilizing the fact that two-clique contains five neighbour pairs $(i, i+1)$ for $i = 1, ..., 5$, we write

$$U(f) = \sum_{i=1}^{5} V_2(f_i, f_{i+1})\,.$$

Obviously, this prior energy is zero (i.e. minimal) for a configuration with all labels being equal, while a configuration alternating between a male and a female yields a maximal prior energy of 500. The prior energy for the configuration f^D which we earlier showed to minimize the likelihood energy is $U(f^D) = 300$, somewhere in between the smallest and the largest prior.

According to (5.1), the posterior energy of configuration f^D is

$$U(f^D|d) = U(d|f^D) + U(f^D) = 99 + 300 = 399\,.$$

The question is, can we find a configuration which yields a better (smaller) posterior energy? And finally, which configuration minimizes posterior energy?

For our small problem, we can simply try all different configurations (there are $2^6 = 64$ in total). Relatively easy we can confirm that a configuration

$$f^O = \begin{bmatrix} M & M & M & F & F & F \end{bmatrix}$$

with $U(f^O|d) = 163 + 100 = 263$ is an optimal configuration.

Note how smoothness cost of 100 (later we call this parameter β) influences which configuration is optimal. In general, the choice of the smoothness parameter depends on your confidence in the prior, compared to the data. Note also that by incorporating the prior information we made sure to find what we expected to find in the first place.

Finally, note the distinction between modeling (setting up the problem by defining a likelihood term and a prior term) and optimization (finding the configuration which minimizes the posterior energy) which in this case involved trying all configurations.

5.2 *MRF modelling for image segmentation*

In this exercise we define an energy function for segmenting a noisy image, similar to the problem in Li Section 3.2.2. Here, we will compute the energy of different configurations to confirm that minimizing the segmentation energy leads towards the desired solution. The model we use is very similar to the model used for gender determination. In this exercise we use synthetic data (i.e. we produce the input image by adding noise to a ground truth image) shown if Figure 5.1. This allows us to evaluate the quality of our energy function. In the text the input image is denoted D (data) and ground truth segmentation S_{GT} where elements of S_{GT} are from the set $\{1, 2, 3\}$ corresponding to the darkest, medium gray, and brightest class.

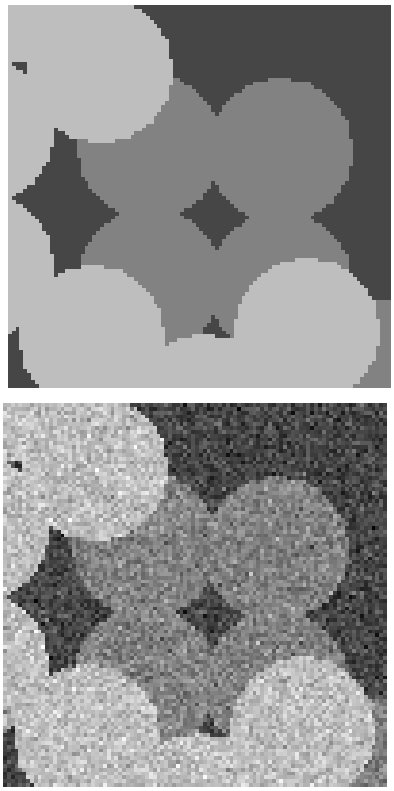

Figure 5.1: A ground truth (the desired segmentation should resemble ground truth) and a noisy image (input data).

Looking at the histogram of the pixel intensities and the intensities divided into the ground-truth classes, Figure 5.2, we observe overlapping distributions, so we can not expect a good segmentation if considering only individual pixel intensities.

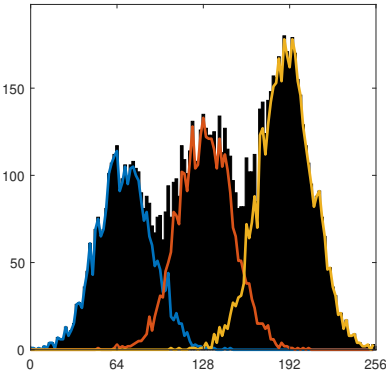

Figure 5.2: Intensity histogram of the noisy image and the histograms for the three segments.

Now we pose image segmentation as a MRF. Sites are pixels, labels are from $\{1, 2, 3\}$, and we choose a first-order neighborhood (four closest pixels). As in the previous example, we define the one-clique potentials for the likelihood energy as the squared distance from the class mean

$$V_1(f_i) = (\mu(f_i) - d_i)^2$$

where d_i are intensities of the (noisy) image, f_i are pixel labelings given by the configuration, and values μ are estimated from the histogram and set to $\mu_1 = 70$, $\mu_2 = 130$, $\mu_3 = 190$. As before, the likelihood energy is

$$U(d|f) = \sum_i V_1(f_i)\,,$$

where summation now covers all image pixels. Similarly to the previous example, we define 2-clique potentials for discrete labels which penalizes neighbouring labels being different

$$V_2(f_i, f_{i'}) = \begin{cases} 0 & f_i = f_{i'} \\ \beta & \text{otherwise} \end{cases}\,,$$

and prior energy

$$U(f) = \sum_{i \sim i'} V_2(f_i, f_{i'})$$

where summation runs over all pairs of neighbouring pixels[2] and β is a smoothness weight, which for `uint8` pixel intensities you may set to 100. The posterior energy is now given by (5.1).

[2] In other words, the summation over $i \sim i'$ means that we sum up contributions of every pair of neighbouring pixels. If we have a image of size $h \times w$, there is $h \cdot (w-1)$ pixel pairs being vertical neighbours and $(h-1) \cdot w$ pixel pairs being horisontal neighbours.

We want to check that our optimal function leads to the desired result. That is, we want to make sure that posterior energy gets smaller when we approach the desired result. Therefore we want to compute the likelihood, prior and posterior for some reasonable segmentations (MRF configurations). For the purpose of this testing, we produce at least two segmentations of the noisy image D. This can be a segmentation obtained by thresholding D at intensity levels 100 and 160 (valleys of the histogram). The second segmentation may be computed by median filtering S_T using an appropriate kernel. You are welcome to produce additional configurations, e.g. by applying a Gaussian filter to D prior to thresholding, or by using morphological operations. For all candidate configurations you should take a look at the intensity histograms of three classes, similarly as for the S_{GT} earlier.

To observe how likelihood (V_1), prior (V_2) and posterior ($V_1 + V_2$) change for different configuration, you need the functions which compute these energies for a given image D, a configuration S and the MRF parameters μ (intensities for segmentation classes) and β (smoothness term). You may chose to write one function which returns V_1 and V_2, or two separate functions.

Tasks You can solve the following tasks from scratch, using only image data. If you want additional help in solving the exercise, use a provided notebook `circles_modelling_EMPTY.ipynb` (or an almost identical `.py` script) where large parts of steps 1. and 4. are already implemented.[3]

[3] As python is 0-indexed, we use the segmentation labels 0, 1, and 2 in the suggested implementation.

1. Get hold of the gray-scale image D and one configuration for the segmentation S. To begin with, this may be a ground truth segmen-

tation. Notice that a ground truth segmentation (labeling) is not the same as a noise-free image: elements of the segmentation are labels $1, 2, \ldots$ while elements of the noise-free image are pixel intensities $\mu_1, \mu_2, \ldots$.

2. For computing V_1 you need D, S and μ. First, compute an intensity-realization of S, that is an image where each occurrence of label f_i is replaced by $\mu(f_i)$. It is then easy to compute V_1 as a sum of squared differences.

3. For computing V_2 you need S and β. Recall that an almost identical problem was solved in week 1.

4. Produce some other configurations S, by any means that you find appropriate: thresholding, manually drawing, modifying ground truth... Apply your two functions to all configurations, and display the likelihood, prior and posterior for every configuration.

5. If we consider only the likelihood, which configuration is the most probable? If we consider only the prior energy, which configuration is the most probable? What if we consider the posterior energy?

6. Would you expect that minimizing the posterior energy leads to a good segmentation? If not, try adjusting β.

5.3 *Graph cuts for optimizing MRF*

The interactions modelled by MRF prior make optimization (finding an optimal configuration) of the MRF very difficult. General MRF optimization methods may be very slow, but efficient graph cut algorithms can be used for a subset of problems.

A binary (two label) MRF problem with submodular second order energy (loosely speaking an energy favoring smoothness and having only one-clique and two-clique potentials) can be exactly solved by finding a minimum s-t cut of a graph constructed from the energy function [4,5,6]. A minimum s-t graph cut can be found e.g. using the Ford and Fulkerson algorithm, or an efficient freely available graph cut implementation by Boykov and Kolmogorov. A multiple-label discrete MRF problem can also utilize graph cuts via iteratively solving multiple two-label graph cuts, e.g. by using α expansion.

In the following exercises we are using graph cuts to optimize discrete MRF. Students using python may use `PyMaxflow` package documented at `http://pmneila.github.io/PyMaxflow`. MATLAB has a built-in function `maxflow` which also implements Boykov's algorithm.

[4] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001

[5] V Kolmogorov and R Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2): 147–159, 2004

[6] Y Boykov and V Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004

To begin with, we look back at the small example with gender labeling. Recall that the heights (in cm) of 6 persons are

$$d = \begin{bmatrix} 179 & 174 & 182 & 162 & 175 & 165 \end{bmatrix}$$

and we want to estimate the persons gender. For likelihood we use squared distance from the means $\mu_M = 181$, $\mu_F = 165$. For the prior we use $\beta = 100$ as a penalty for neighbouring labels being different.

We want to construct a s-t graph corresponding to this problem. The construction is not unique. When choosing an approach, the focus is often on constructing a graph with fewest edges, as suggested in Li book Section 10.4.2. However, you might prefer constructing a more intuitive graph despite having a higher number of edges. This approach is sketched in Figure 5.3. Terminal edges (linking to source and sink) are used for the likelihood energy terms, while internal edges model the prior energy terms. Confirm that a cost of an s-t cut in this graph equals to the posterior energy of the corresponding configuration.

Depending on the graph cut package which you use, you need to pass the information about the graph to the solver. Read the package documentation and look at the provided examples to see how to construct the graph. One way of representing a graph (used by Boykov's original MATLAB implementation) is by the means of two matrices shown in Figure 5.4, but python wrapper requires you to pass graph weights to the solver in loops.

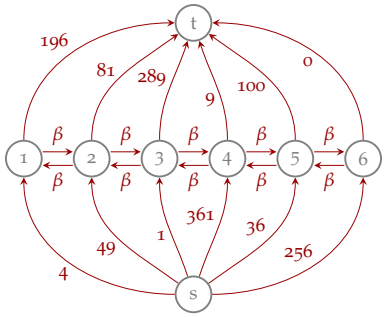

Figure 5.3: A sketch of a s-t graph for a gender labeling problem.

$$E_{\text{terminal}} = \begin{bmatrix} 1 & 4 & 196 \\ 2 & 49 & 81 \\ 3 & 1 & 289 \\ 4 & 361 & 9 \\ 5 & 36 & 100 \\ 6 & 256 & 0 \end{bmatrix}$$

$$E_{\text{internal}} = \begin{bmatrix} 1 & 2 & \beta & \beta \\ 2 & 3 & \beta & \beta \\ 3 & 4 & \beta & \beta \\ 4 & 5 & \beta & \beta \\ 5 & 6 & \beta & \beta \end{bmatrix}$$

Figure 5.4: Representing a s-t graph using two matrices, one containing weights of terminal edges and one matrix for internal edges.

Tasks

1. Install the required `maxflow` package.
2. Get a notebook (or script) which solves the gender labeling problem.
3. Run the script and get familiar with the functionality of the graph-cut solver.
4. Which configuration is optimal? Change $\beta = 10$ and solve again. Which configuration is optimal now? Try also $\beta = 1000$.
5. Try changing the data, as sugested in the notebook.

5.4 *Binary segmentation using MRF*

Now we can set up and solve a binary segmentation using MRF. When using a method for the first time, it is always a good idea to start with an easy example, where it is clear what the desired output is. This may be a simple image corrupted by noise, for example a noisy image of the DTU logo.

After that, we will segment the bone image `V12_10X_x502.png` shown in Figure 5.5. The image is a slice from a CT scan of a mouse tibia. You can visually distinguish air (very dark), bone (very bright) and cartilage (dark). The task here is to segment the image in two segments: air and bone. Cartilage should be segmented together with air. In the next exercise we look at multilabel segmentation, allowing us to distinguish all three materials as in Figure 5.6.

The model we use is still the same as in the previous exercises, with the likelihood as the sum of squared distances, and the prior penalizing neighboring labels being different.

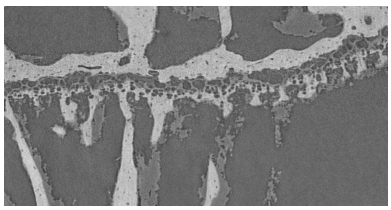

Figure 5.5: A bone image.

Tasks

1. The DTU logo is of type `uint8` and should be converted into double precision (float) before any computation. You also want to divide image intensities with 255, as this will simplify the weighting between the likelihood and the prior term. [7]
2. For the DTU logo use $\mu_1 = 90/255$ and $\mu_2 = 170/255$.
3. To pass the weights of terminal and internal edges to graph solver, python users should use the functionality provided by `maxflow`. See the example *Binary image restoration* but notice that it uses a different energy formulation for likelihood term.
4. For the DTU logo, choose parameter β which yields in a good segmentation.
5. The bone image is of type `uint16` and after converting it into double precision you may also want to divide image intensities with $2^{16} - 1$.
6. For bone image, you need to determine the mean intensities of the air and bone (μ_1 and μ_2) by inspecting the histogram. Look at maximum likelihood configuration, to confirm that your μ_1 and μ_2 are suitable. Choose a (small) parameter β and compute the optimal configuration for this β using your graph cut solver. Then, adjust β to obtain a visually pleasing segmentation with reduced noise in air and bone choose. Observe how changing β affects the segmentation.
7. For your results, you may want to produce a figure showing histogram of the entire image, and on top of that the intensity histograms of the air and bone classes, similar to how it was done in the modelling exercise.

[7] When creating graph using `maxflow` package, remember that you now need float capacities.

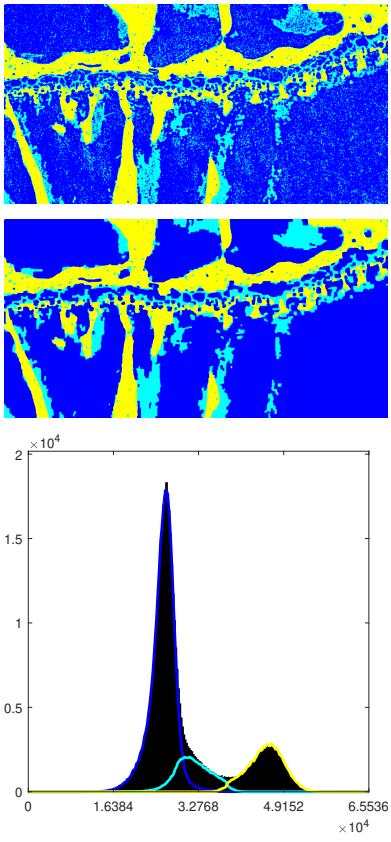

Figure 5.6: A segmentation of bone using maximum likelihood (top) and maximum posterior (middle). Histograms show intensity distributions for the three segmented classes.

5.5 *Multilabel segmentation usign MRF (optional)*

Multilabel segmentation is obtained using an iterative α expansion algorithm. In python, use the `maxflow.fastmin.aexpansion_grid`function which is a part of `maxflow.fastmin`. For those using MATLAB we can

provide a function `multilabel_MRF` which implements α expansion. Read the help text of the function for explanation on input and output variables.

You should first verify the quality of the solution provided by the α expansion algorithm by segmenting the synthetic image, for example circles. How does the energy of the graph cut solution compare to the energies of the configurations found in the modeling exercise? Try changing β to see how it affects the result.

Use the α expansion algorithm to segment the bone image into air, cartilage and bone class. The challenge here is to distinguish between air and cartilage. You should aim at producing a visually pleasing result with cartilage as solid as possible (without noisy pixels segmented as air) and air as clean as possible (without noisy pixels segmented as cartilage). A good result can be obtained by tweaking two parameters: the mean value for the cartilage class and the smoothness weight β. The mean intensity for air and bone class can be estimated from the histogram.When adjusting a mean value for cartilage, choose first no smoothing ($\beta = 0$) and try to obtain a reasonable (but noisy) segmentation. Then increase β to remove the noise.

After you have tuned the parameters and obtained a nice segmentation, try segmenting the other bone image (`V8_10X_x502.png`). Can you use the same parameters? Why?