CS 6320: Computer Vision Spring 2019

Homework 3

Abhinav Kumar(u1209853)
abhinav.kumar@utah.edu

Due date: 11 March, 2019

1 Belief Propagation

Implement a belief propagation program that uses a factor graph and cost tables as input, and producesthe states/labels for all the variables. The BP program can be summarized in 5 steps as shown in the lecture(Slide 9 of http://www.eng.utah.edu/~cs6320/cv_files/BeliefPropagation.pdf).First, we initialize all messages from variable to factor nodes with zeros or random values. The algorithm iterates through steps 2 to 5 based on two terminating conditions.

- The first terminating condition checks if the labels of all the variables change with subsequent iterations or not. If there is no change, the algorithm is terminated.
- The second terminating condition checks if the number of iterations has exceeded a threshold or not.

Tips: During the implementation of BP, please assume that all variables take the same number of labels.

1. [20 points] Write the code.

   Algorithm :

   - Intialize all var to fac node messages to zero. Each message is a vector with dimension equals the number of labels
   - Compute the local minima at each of the factor nodes and therefore the message sent by factor node i to variable node j by using the equation

$$m_{s_j}(i \to j) = \min \left[C(j = s_j) + \sum_{k \in N(i)-j} \sum_{s_k=0}^{\#S-1} m_{k=s_k}(k \to i) \right] \tag{1}$$

   - The belief at each of the variable nodes j is given by

$$b_{s_j}(j) = \sum_{i \in N(j)} m_{s_j}(i \to j) \tag{2}$$

- The message from variable node j to factor node i is

$$m_{s_i}(j \to i) = b_{s_j}(j) - m_{s_j}(i \to j) \tag{3}$$

- The label of the variable node j is given by

$$label(j) = \underset{s_j}{\operatorname{argmin}}\, b_{s_j}(j) \tag{4}$$

Output :
The output of the code shows all the labels and are shown as [0, 0, 0, 0, 0, 0, 0] which is expected since even one of the bit going one makes the cost for one of the nodes infinite.

2 Stereo Matching

The basic idea of stereo matching is given here. You are given two images that we normally refer toas left and right images. We can think of each image to have several horizontal scanlines. The images are said to be rectified if every pixel in the left image has a matching pixel in the right image at the same scanline. In a typical stereo pair, a pixel $p(x, y)$ in the left image usually has a matching pixel $q(x', y)$ in the right image. Note that the y coordinate is the same on both the left and right images. Every pixel in the left image gets shifted by a small amount d on the right image in the following manner: $x' = x - d$. Here d is supposed to be the disparity for the pixel $p(x, y)$ in the left image. Note that a pixel that is further away from the camera will have a very small disparity value. A pixel that is very close to the camera will have a high disparity value. In other words, the depth D of a pixel is inversely proportional to the disparity d.

We will use BP to minimize the following energy function to find the optimum disparity labels/states:

$$E(d) = E_{data}(d) + \lambda E_{smooth}(d) \tag{5}$$

The first term is the data cost and it is given by

$$E_{data}(d) = \sum_{x,y} C(x, y, d(x, y)) \tag{6}$$

The cost $C(x, y, d)$ in matching a $p(x, y)$ with another pixel $q(x - d, y)$ can be a simple pixel-wise intensity difference as shown below:

$$C(x, y, d(x, y)) = |I_L(x, y) - I_R(x - d, y)| \tag{7}$$

Let us consider the left image to be the reference image. Each pixel in the left image can be seen as a variable that can take different disparity values (say $1 - 50$). As shown in Figure 2, the data term consists of MN (M is the height of the image and N is the width of the image) unary factor nodes where each factor node is attached to every pixel in the image. Each of the cost tables for unary factor node will only consist of 50 values (assuming that each pixel can take only 50 disparity states)

We will use Potts model for the smoothness term as shown below:

$$E_{smooth}(d) = \sum_{i,j \in E} f_{Potts}(d(x_i, y_i), d(x_j, y_j)) \tag{8}$$

where

$$f_{Potts}(d(x_i, y_i), d(x_j, y_j)) = \begin{cases} 1, & \text{for } |d(x_i, y_i) - d(x_j, y_j)| > 0 \\ 0, & \text{for } |d(x_i, y_i) - d(x_j, y_j)| = 0 \end{cases} \tag{9}$$

1. [50 points] Write a stereo matching program to compute the disparity map using the BP algorithm.

   Algorithm:

   - Intialize all var to fac node messages to zero. Each message is a vector with dimension equals the number of possible values of disparity map.
   - Compute the local minima at each of the unary factor nodes and therefore the message sent by the unary factor node $u(x, y)$ to variable node $p(x, y)$ by using the equation

$$m_d(u \to p) = |I_L(x, y) - I_R(x - d, y)| \tag{10}$$

- Compute the local minima at each of the pair factor nodes and therefore the message sent by the pair factor node $v(x, y)$ to variable node $p(x, y)$ by using the equation

$$m_d(v \to p) = \min \sum_{k \in N(v)-p} \sum_{d_1=0}^{\#S-1} [\lambda f_{Potts}(d, d_1) + m_{d_1}(k \to v)] \tag{11}$$

- The belief at each of the variable nodes p is given by

$$b_d(p) = m_d(u \to p) + \sum_{v_i \in N(p)} m_d(v_i \to p) \tag{12}$$

- The message from variable node p to unary factor node u is

$$m_d(p \to u) = b_d(p) - m_d(u \to p) \tag{13}$$

- The message from variable node p to pair factor node v is

$$m_d(p \to v) = b_d(p) - m_d(v \to p) \tag{14}$$

- The label or the disparity of the variable node p is given by

$$label(p) = \underset{d}{\operatorname{argmin}}\, b_d(p) \tag{15}$$

- The termination condition should be either 20 iterations or when the labels stop changing in subsequent iteration

2. Use different values for $\lambda = 1, 10, 50, 100$. Show the solutions and the value of the final cost function. Show the final result in a disparity image.
   We implement this algorithm in Python and the results are shown below :

   Image 1 Pair
   The cost function is shown in table 1 and the outputs for this image pair is shown in figures 1 to 4.

λ	Final Cost
1	770482
10	2932893
50	9552980
100	18024621

Table 1: Final Cost Function values of image 1 pair with different values of λ

Figure 1: Image 1 pair images and disparity map with $\lambda = 1$

Figure 2: Image 1 pair images and disparity map with $\lambda = 10$

Figure 3: Image 1 pair images and disparity map with $\lambda = 50$

Figure 4: Image 1 pair images and disparity map with $\lambda = 100$

Image 2 Pair

The cost function is shown in table 2 and the outputs for this image pair is shown in figures 5 to 8.

λ	Final Cost
1	766146
10	2649353
50	9594762
100	18160196

Table 2: Final Cost Function values of image 2 pair with different values of λ

Figure 5: Image 2 pair images and disparity map with $\lambda = 1$

Figure 6: Image 2 pair images and disparity map with $\lambda = 10$

Figure 7: Image 2 pair images and disparity map with $\lambda = 50$

Figure 8: Image 2 pair images and disparity map with $\lambda = 100$

Observations :

- As we increase the λ, we are able to see more smoother disparity maps. This is expected since the pairwise costs now increase and therefore the Belief Propagation emphasizes more on ensuring the smoothness between the pixels.
- The cost function values decrease with the iterations suggesting that the label values is moving towards convergence.
- Another interesting thing was the vectorized implementation of the code. Without the vectorized implementation, the code runs extremely slow.