Stereo Correspondence

1st Yuan Yang
yyang998@gatech.edu
Computer Vision

Abstract

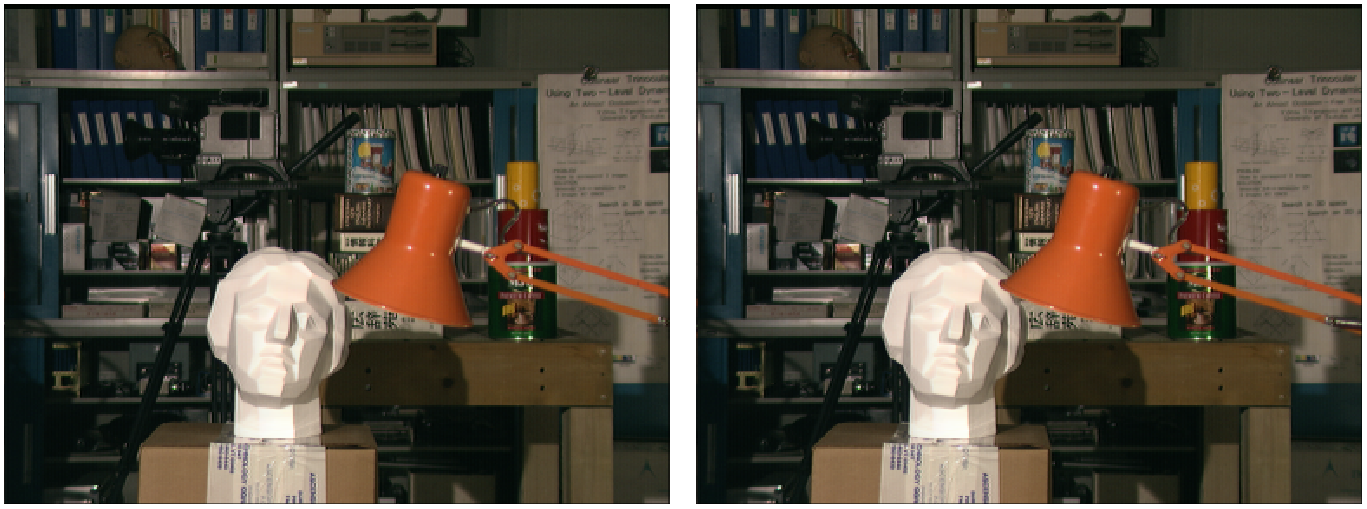
The experiments presented in this work are designed to solve the stereo correspondence problems. Two methods (window-based and energy minimization) are implemented to obtain the disparity map (depth information) between a pair of left-right views. The performance of the energy minimization method was tested and found to be better than the window-based method on some of the data in the Stereo Data-sets provided by Middlebury [1].

I. Introduction

A. *Stereo Correspondence Problem*

As shown in Figure 1, given a pair of left-right stereo images, the two images represent different views of the same scene. The difference in the two views should be relatively small. When examining the relative position between the black wire on the table lamp and the yellow can in the background of the image, it is obvious that the camera has shifted from capturing the left view to capturing the right view. The **stereo correspondence problem** is to find the corresponding pixels and their disparity in the image pair. The views of the image pair in this project are pre-processed and rectified to only move horizontally; therefore, it can be assumed that for a pixel in the left view, its corresponding pixel in the right view should be on the same row. Regarding **disparity**, when the camera shifts a certain distance, objects that are closer to the camera (e.g., table lamp, statue) have a larger disparity compared to objects in the background (e.g., cans, books). This is because the pixels associated with closer objects would move by a larger number of column positions.

The goal of this project is to find a disparity map between the left and right views. For each pixel (x, y) in the left view, using its disparity value in the disparity map, we can find the corresponding (most similar) pixel $(x - \text{disp}, y)$ in the right view. The disparity value in the map should be an integer to shift axis positions in the images.


(a) Left View (b) Right View
Fig. 1: Left-Right Stereo Image Pair Example

B. Literature Study and Existing Methods

1) ***Window-Based*** *[2] [3] [4]:* As introduced in the last section, to obtain the disparity map, we need to find the pixel in the right image with the most similarity compared to a pixel in the left image. Since we also want to track the pixels associated with each object in the scene, instead of comparing pixel by pixel, window/kernel-based similarity checking methods could be used.

To evaluate the similarity between a window in the left view and a window in the right view, several matrices can be calculated, such as the Sum of Absolute Differences (SAD), Sum of Squared Differences (SSD), and Normalized Cross-Correlation (NCC). During the process, the chosen metric should be minimized to reflect optimal similarity. This project used the SSD metric, and the details of the algorithm are explained in the Algorithm section below.

2) ***Energy Minimization****:* The issue with window-based methods is that they could introduce artificial patterns when edges are occluded, leading to a noisy disparity map. Therefore, several researchers proposed the Energy Minimization theory to enhance accuracy and performance. This theory is developed based on the assumption that pixels that are adjacent or near should have similar disparity since it is more likely that these pixels are within the same object in the scene or their depths are similar [4] [5]; this is referred to as ”smoothness”. Therefore, the information of neighboring pixels should also be considered. The Energy function was developed to reflect the information of both the pixel itself and its neighboring pixels.

The Energy function for one pixel can be represented using:

$$E(d) = E_{data}(d) + \lambda E_{smooth}(d) \tag{1}$$

To evaluate the performance of the current disparity value d for one pixel, the total energy presented in Equation 1 should be minimized. $E_{\text{data}}(d)$ is the data term that represents the similarity between left and right pixels with disparity value d, while $E_{\text{smooth}}(d)$ is the smoothness term that enforces smooth disparity across a pixel’s adjacent neighbors.

There are various ways to compute these two terms and minimize the energy. The implementation and algorithm details are introduced in Algorithm section below. Some major methodologies designed to minimize energy are belief propagation [5] [6] [9], graph cut [7] [8] [9], and dynamic programming. Both the Belief Propagation and Graph Cuts methods were demonstrated to have satisfying performance in obtaining disparity map.

To explain these two methods, it is first necessary to introduce Markov Random Field (MRF) [5], which is a graph model that contains nodes and edges and is often used to represent spatial dependencies. As shown in Figure 2, the stereo problem can be represented using MRF. Inside MRF, there are two types of nodes: intensities of every pixel and corresponded optimal disparity values. For a 3×3 image, as shown in Figure 2, the blue nodes are the 9 pixels, while the 9 red nodes are the corresponding disparities. The edges indicate the dependencies between nodes. The data term in Equation 1 can be treated as the edges between the pixel node and its possible optimal disparity node, while the smoothness term can be treated as the edges between a disparity node and its four neighboring disparities (because we need to ensure the smoothness for adjacent pixels). The goal is to minimize the data term and the smoothness term and to solve the red nodes (optimal disparity values) for pixels. Both Belief Propagation and Graph Cuts could be used to efficiently find approximate minima.

The Graph Cuts method [7] finds the minima using the Min-Cut/Max-Flow algorithm on MRF: find a min cut such that the sum of the edge costs on this cut is the minimum across all possible cuts.

For the Belief Propagation method [5], the energy is minimized by passing messages between neighboring pixels. Similar to dynamic programming, the message could be eventually passed to the whole image. The messages passed through contain information to notify the current pixel of the cost of choosing each different disparity value. Therefore, a pixel can choose a value that costs the least and further minimize

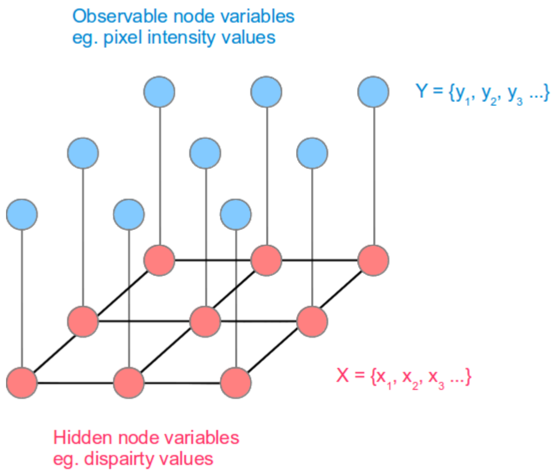


Fig. 2: MRF for Stereo Problem - for 3×3 image [5]

the energy. I implemented a setup of Loopy Belief Propagation (LBP) since it is more straightforward. In the next section, the details of this algorithm, how message passing can be used to minimize the energy, and some implementation choices are introduced.

II. Algorithms and Strategies Explanation

A. *Window-based - SSD*

The implementation of window-based method is straightforward. For a window c_l in the left view, given a possible disparity value d, the window c_r in the right view could be obtained by shifting the column number of pixels in the window by d since the view only moves horizontally. Then, the Sum of Squared Difference (SSD) between two windows c_l and c_r can be calculated. For a window c_l, there could be a range of possible disparity values. For every disparity, one SSD result could be calculated. Therefore, for a window c_l, there could be multiple SSD results (number of possible disparities). The optimal disparity value for current window $d_{\min}$ is the disparity that generates the minimum SSD value among all the SSD results.

B. *Energy Minimization - LBP*

Loopy Belief Propagation (LBP) is used to adapt energy minimization theory to improve the stereo results. The implementation of this algorithm is based on the Energy Function:

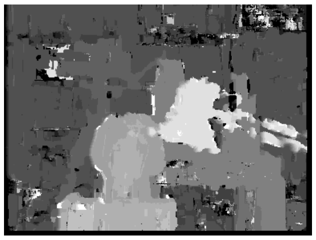
$$Energy(pixel, disp) = \sum_i DataTerm(pixel_i, disp_i) + \lambda \sum_j SmoothTerm(disp_i, disp_j) \tag{2}$$

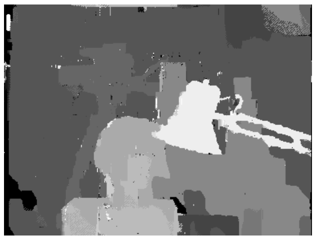
where $pixel$ is the blue node in MRF, $disp$ is the red node in MRF, i is the pixel index, j is the neighbors of i. For a pixel at i, the neighbors are the four pixels around it (up, down, left, right pixels). The goal is to obtain the disparity $disp$ that minimize the Energy.

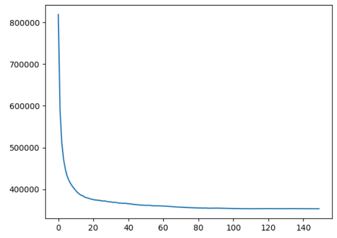
Equation 3 is used to compute $DataTerm$:

$$DataTerm(pixel_i, disp_i) = \min(\frac{1}{3}||left(pixel_i) - right(pixel_i - disp_i)||_1, \tau) \tag{3}$$

Note that the $(\text{pixel}_i - \text{disp}_i)$ term only shifts the pixel position horizontally (subtracting the column number by disp_i). Also, this term is for an RGB image that has three channels. This data term should be minimized by the disparity value.


(a) SSD


(b) LBP



(c) LBP - Energy vs. Iteration#

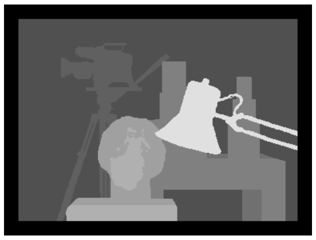

(d) Ground Truth

Fig. 3: Result for Lamp Image Pair

For smoothness term, Potts model is used to compute it.

$$SmoothTerm = \begin{cases} 0, & \text{if } \ disp_i - disp_j = 0 \\ 1, & \text{otherwise.} \end{cases} \tag{4}$$

The next step is to determine how to pass the messages through the MRF to compute beliefs and further minimize the energy. As already introduced previously in the last section, the message passing only happens on red nodes, which are the disparities. For a node x, the messages are passed to it from its neighbors in four directions. Each message contains the cost for the current node receiving a message to have a specific disparity value $disp$.

$$message_{neigbor->x}(disp) \tag{5}$$

The LBP algorithm iterates through a specific number of time steps. All messages are initialized to 0. At each iteration, every node sends a message to its neighbors. For a node i, when sending a message to its ”up” neighbor j, the message contains the sum of both the data term and the smoothness term, along with the sum of the messages that node i received from its other 3 neighbors (down, left, right) except for j. And this message should be minimized over the disparity value. Also, the message passed through should be normalized.

At the end of each iteration, the belief term is computed by summing the data term and the sum of the optimal messages (minimized) passed from all the neighbors. Then the belief term is used to find the optimal disparity that minimizes the energy.

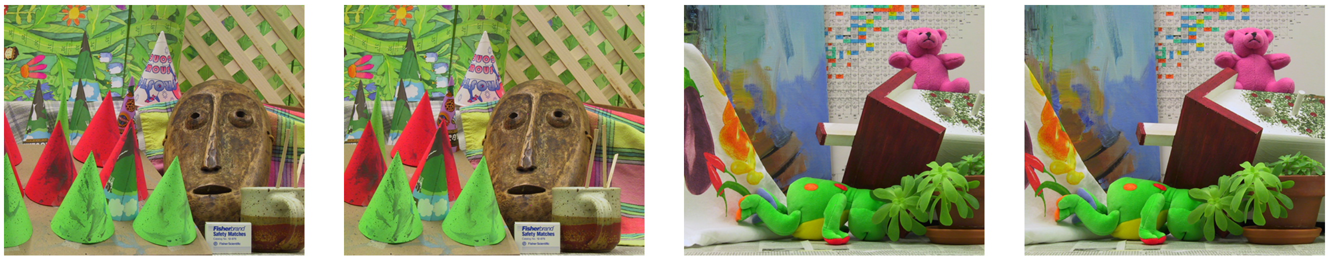
After finding the optimal disparity, the energy for this iteration could be computed using the Energy Function and this disparity value.

As the iteration progresses, the energy should be minimized along the way, and the disparity map at the last iteration serves as the final disparity map result.

III. Experiments and Results

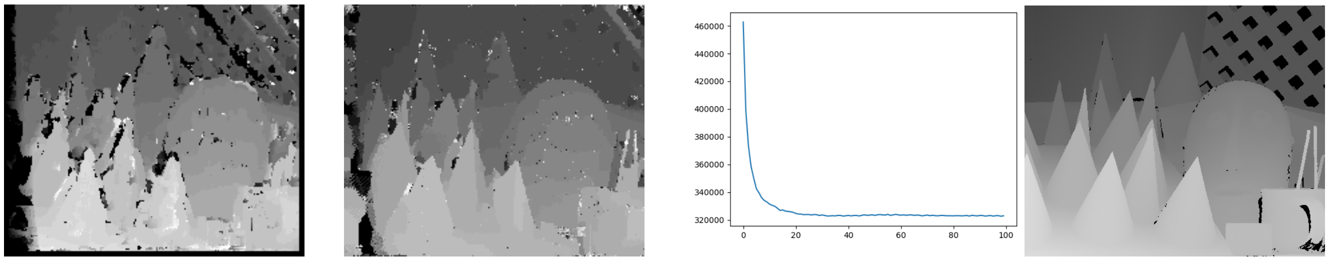
The data used in work is from 2003 and 2014 Stereo Data-sets provided by Middlebury [1]. Some of the images are resized using OpenCV’s resize() to speed up the experiments.

In Figure 3, the image pair from Figure 1 in the lecture is initially employed to evaluate the performance of SSD and LBP. The energy minimization process (Energy vs. Iteration Number) is plotted as well. As depicted in Figure 3(a), the SSD result is notably noisy and contains artifacts absent in the ground truth data (Figure 3(d)). Upon transitioning from the window-based SSD to LBP, the generated result (Figure 3(b)) illustrates the superior performance of LBP. The object edges are much clearer compared to the SSD result and closely resemble the ground truth. The inclusion of a smoothness term in the Energy Function proves beneficial for enhancing the smoothness of edges, further indicating the assumption made in the introduction and algorithm sections – that closer pixels should have similar disparity values.



(a) Left View - Cones (b) Right View - Cones (c) Left View - Teddy (d) Right View - Teddy

Fig. 4: Cones and Teddy - Two Left-Right Image Pairs



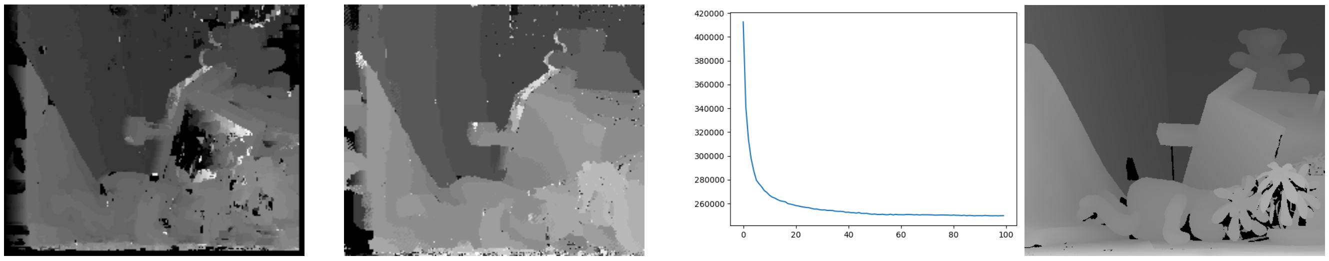
(a) SSD (b) LBP (c) LBP - Energy vs. Iteration# (d) Ground Truth

Fig. 5: Result for Cones Image Pair

Additionally, Figure 3(c) illustrates the energy minimization process. Over 150 iterations for message passing, the energy successfully decreases to converge just under 400k at iteration #100 for the current LBP disparity map. According to [5], the energy obtained from the ground truth is 720k. The observed difference between the final minimal energy of LBP and the ground truth energy suggests that the implemented LBP algorithm in this work is more aggressive, possibly resulting in some overshooting effects during energy minimization. Future work may involve exploring different algorithms for calculating the data term and smoothness term. Preprocessing the image might also be considered to achieve improved results.

Two extra image pair from 2003 Stereo Data-sets are also used: Cones Image Pair and Teddy Image Pair (Figure 4). These images are downsized by 50% to speed up the stereo matching process.

The results for the Cones Pair are presented in Figure 5. The energy minimization process successfully converged, and LBP outperformed the SSD method, showing clear edges and fewer artifacts. However, in the case of LBP, some edges in the background (specifically, edges in the fences in the background) were not detected in the disparity map. Additionally, there were instances of cones overlapping with occlusions, resulting in artifacts in the edges of some cones in the background. A potential avenue for future improvement involves more extensive tuning of hyperparameters, considering that LBP has more tunable parameters than the SSD method. Furthermore, changing the message-passing method from Min-Sum to Sum-Product, as suggested in [5], could lead to better results.



(a) SSD (b) LBP (c) LBP - Energy vs. Iteration# (d) Ground Truth

Fig. 6: Result for Teddy Image Pair

Figure 6 presents the results for the Teddy Pair. Although the energy minimization process successfully converged, the disparity map quality produced by LBP is not satisfactory. While the result is less noisy compared to SSD, some important edges appear to be lost or smoothed out, such as the edges associated with the teddy bear in the background. This could be attributed to the nature of the image pair, where numerous foreground objects may act as distractions for those in the background. A potential improvement could be achieved by adjusting the algorithm used for calculating the smoothness term to better handle objects in the background, which typically have less disparity. Furthermore, the limited detection of the teddy bear (only detected part of the bear) by LBP may be caused by inefficient messaging updating, resulting in suboptimal handling of neighboring messages within the teddy bear. To enhance this, Graph Cuts methods could be employed for energy minimization instead of relying on beliefs generated from messages.

IV. Conclusions

Two methods, SSD and LBP, have been implemented in this study to address stereo matching problems. Overall, the LBP method outperforms SSD. LBP performs well on simple image pairs with clear objects and minimal overlapping. However, when dealing with images containing more overlapping objects, especially in the foreground, the performance of LBP is not ideal when compared to the ground truth.

Some possible future improvements includes:

- Switch to the Graph Cuts method for energy minimization.
- Further tune all hyperparameters included in LBP more extensively.
- Adjust the algorithm used for calculating the smoothness term to better handle objects in the background.
- Change the message-passing method from Min-Sum to Sum-Product.
- Preprocess images.

References

[1] https://vision.middlebury.edu/stereo/data/scenes2014/

[2] Szeliski 2010 (chapter 11)

[3] Forsyth and Ponce 2012 (chapter 13)

[4] https://www.cs.cmu.edu/~16385/s17/Slides/13.2_Stereo_Matching.pdf

[5] https://nghiaho.com/?page_id=1366

[6] Sun et al. ”Stereo Matching Using Belief Propagation”

[7] Boykov et al. ”Fast Approximate Energy Minimization via Graph Cuts”

[8] Kolmogorov. ”What Energy Functions Can Be Minimized via Graph Cuts”

[9] Freeman et al. ”Comparison of Graph Cuts with Belief Propagation for Stereo, using Identical MRF Parameters”