Application of Graphical Model in 3D Reconstruction/Stereo Image Segmentation

Bing Li

University of Chicago, Chicago, IL 60637

Abstract

In this project, I'll try to implement a basic program for 3D reconstruction/stereo image segmentation using graphical models. As we all know, the task of 3D reconstruction is to build 3D models, i.e. to learn the depth for all the objects from multiple 2D photos taken at different positions and angles, which are sometimes unspecified. 3D reconstruction is one of the main topics in the field of computer vision. The main difficulty of 3D reconstruction/stereo image segmentation is to associate equivalent points across different pictures such that the depths or say the 3D locations of those points could be inferred. In class it's briefly mentioned that the graphical model is a good candidate since the Markov Random Field could be used to guess the shift of each pixel between 2 pictures. In this project, I'll try graphical models first. And it's good to think about other complexities associated with 3D reconstruction/stereo image segmentation, for example, some objects are totally different seen from different directions and some points on one picture might be absent from another picture, either blocked or dark due to mirror reflection. It's interesting to know that if we can find solutions to these problems within graphical models. The experience in 3D reconstruction could also be useful in tasks like face recognition, since 2 pictures of the same person could be quite different when observed in different directions and environments.

I. INTRODUCTION

Image segmentation has vital importance in computer vision since the first step for computers to understand the surrounding scene and story is to recognize all objects. Image segmentation/3D reconstruction can be divided into two main types, the stereo method and the monocular method. The stereo image segmentation has close relation to 3D reconstruction because both aim to find the depth for each pixel. In this final project I will focus on the stereo matching method. Stereo matching, with the full power of the perspective relations between two images, is no doubt much more effective than the monocular method using only one single image. However stereo matching could be improved a lot with extra monocular information like texture variations, gradients, defocus and color. A typical example of monocular 3D reconstruction was done by Ashutosh Saxena, Sung H. Chung and Andrew Y. Ng from Stanford in 2007, where they used supervised learning and multiple-layer MRF. In terms of stereo matching method, the basic idea is to transform the matching problem into a problem of minimizing an energy functional over a RMF model and use approximations to avoid NP complexities. The state-of-the-art approximation techniques include graph-cuts and belief propagation. Difficulties associated with stereo matching problem also include the occlusion problem and image noise, which needs more carefulness with definition of the energy functional. In this project, I first tried the naive scanline algorith and then tried to implement the simple-tree method described in the paper "Simple but Effective Tree Structures for Dynamic Programming-based Stereo Matching" by Michael Bleyer and Margrit Gelautz. Finally I described the advantages and disadvantages of each approach and also suggestions how to improve them.

II. DATA PREPROCESSING

I used the Middlebury data sets to test the program. One data set includes a left-view photo, a right-view photo and corresponding ground-truth depth images as shown in Fig 1. The 2 photos were taken by 2 cameras at the same height so they are well aligned in the vertical direction. However there is a large horizontal global shift. Apparently, the pottery in the left image is almost missing in the right image. So firstly I made a rough alignment by finding the optimal global shift. The basic idea is that the two pictures are quite consistent

at large scale even though there exist slight disparities locally. I samples a 21-by-21 matrix of pixels as shown in Fig 2 and calculated the total color distance across two pictures, and then plotted distance over global shift. Minimizing this distance gives the optimal global shift of 150 pixels. And to lower the computational complexity, I combined every 5-by-5 square of 25 original pixels into a larger pixel whose color is just the average color in that square. In this way I converted a problem with size 1110×1390 to one with size 222×248.

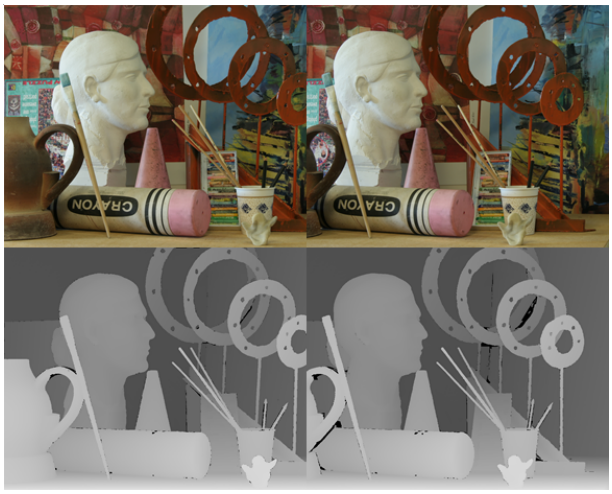


FIG. 1: The Middlebury data set "Art": the upper are real pictures and the lower ones are ground-truth image segmentation result. Lighter color means closer and smaller disparities.

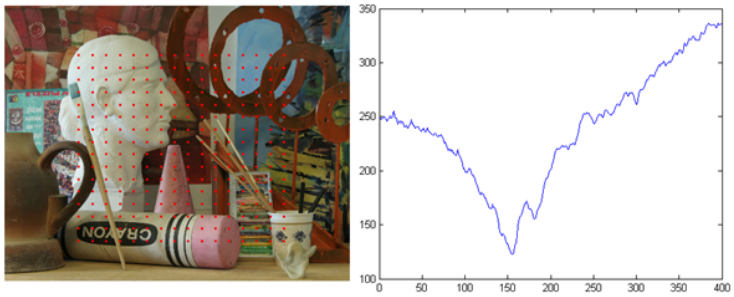


FIG. 2: The left panel is a 15 by 15 sample array used for the rough alignment. And the right panel shows a plot of the total color distance of 225 point pairs versus the global shift. And it's easy to see that the minimum is about 150 pixels.

III. IMPLEMENTATION DETAILS

In this project, I tried several simple stereo methods and compared their performances on the "Art" picture. First I explored . Then I

A. Results Without Smoothness Regularization

In the first part, I keep only the color similarity term in the energy function, in other words I'm trying to see what's the best one can do without smoothness constraints on the disparity array. More specifically for some pixel p_i in the left image, I searched in the neighborhood of the ith pixel q_i in the right image (from i-k to i+k, k = 18) for the one with the most similar color as p_i.

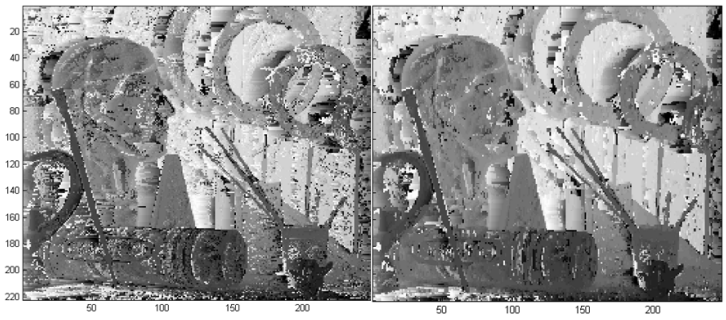


FIG. 3: result without smoothing

Drawback : (1) noisy, didnt punish non-smoothness (consider a large region with the same color) (2) bad performance wherever occlusion happens, i.e. the pixel doesnt appear in pic2; (3) texture is important, the dirtier, the better.

B. Optimization of the Energy Functional

Usually the stereo matching problem can be transformed into an optimization problem. need to minimize a energy functional E(D) which maps the disparity function to energy or cost. The energy functional is made of two terms, the pixel matching term and the discontinuity term.

$$\sum match(p_i, q_{i+di}) + \sum discontinuity(d_i, d_{i+1})$$

C. Scanline Method with DP and its Limitation

Scanline method is the simplest approximation in which we cut off all the vertical interconnections in the 4-grid graph, such that we can optimize on each horizontal line separately and simplify the problem significantly. As is well known the optimization in a line can be solved by dynamic programming algorithm as shown below: For each line: (1) Define l(pi, dpi) as the minimum cost/energy from the left up to pixel i if the disparity of pi is dpi; (2) Update table l(p,dp) from left to right, i.e. pixel i = 1 to n; (3) trace back i from n to 1 find the configuration which gives the minimal loss(energy). Drawbacks of scanline algorithm : (1) The graph is simplified too much and different lines dont talk to each other; (2) There is so-called streaking problem (apparent in the image).

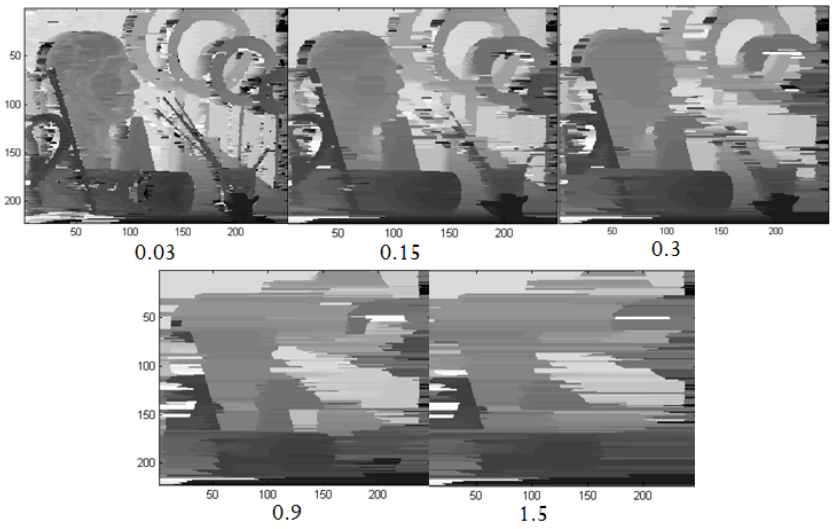
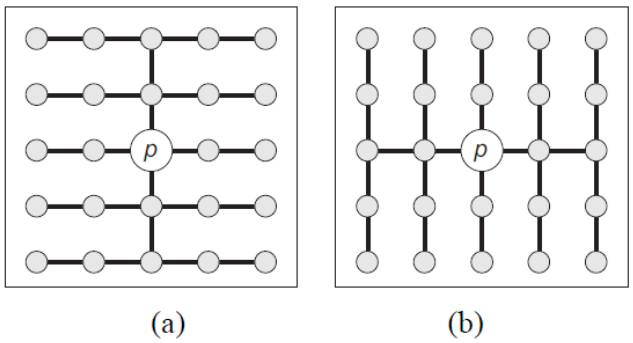


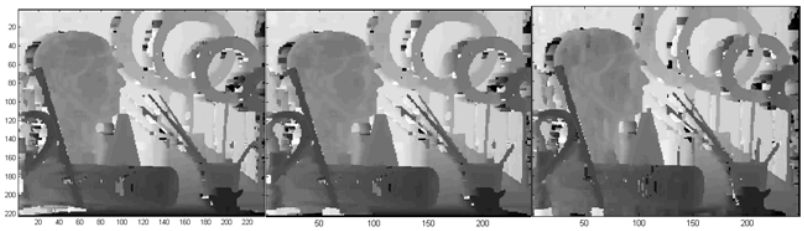
FIG. 4: In this picture, I showed results for different ratios between smoothing term and matching term. As we can see, as the smoothing term dominates, it takes more energy to jump from one disparity to another, so that the disparity map is smeared out. The optimal ratio is 0.15.



D. Simple Tree Method with DP

Efficient DP-based optimization also works on tree structures. Tree structure has the potential of giving better performance compared to scanline since now we have both horizontal and vertical connections. A possible way to construct a tree is to start with the full graph, and then the tree is constructed by discarding edges that show a high gradient in the intensity image, which is more likely to be the boundary between two distinct objects, and finally we end up with a tree graph. Disadvantage: still discard too many edges. Therefore people proposed the method called the simple-tree method, in which, for each pixel we construct two individual trees(V and H as shown in the picture below), with that pixel as the root. Then we find the disparity for that root pixel p that minimizes the energy on this simple tree and assign it to this root pixel. Since we have two trees for each pixel, we do vertically first and then use this result to bias our matching function and then do the vertical direction to find the final disparity assignment. Then we repeat for all pixels. In this way we employed both vertical and horizontal information to a large extent.

$$m'(p,d) = m(p,d) + \lambda \cdot (V(p,d) - \min_{i \in \mathcal{D}} V(p,i))$$



Lambda = 0 (H limit) | Lambda = 1 | Lambda = inf (V limit)

FIG. 5: In this figure, I showed the simple-tree result for different values of λ. As λ is 0, which is the Horizontal tree limit, horizontal features get enhanced. As λ is infinity we go back to the vertical tree limit. And as λ is somewhere in the middle, we can see that the result has be benefits from both(The gap in the ring is closed and the rightmost stick is smooth.)

Algorithm(for Horizontal tree): (1) Run DP scanline for each horizontal line, both forwards(F(p,d)) and backwards(B(p,d)); (2)Calculate intermediate data structure C, V, H;

(3) C(p,d) is the cost of a whole line with optimal disparity solution conditioned that p is assigned d; (4)H(p,d) is the minimal cost of the whole tree, calculated by DP on C.

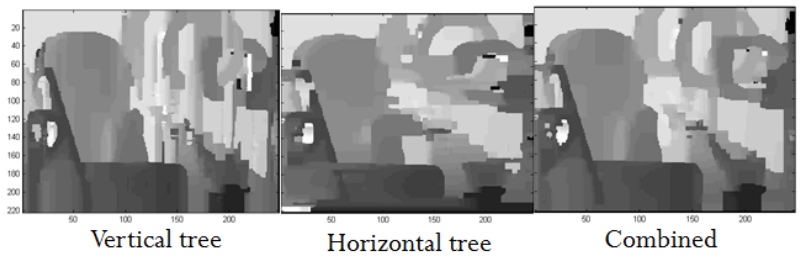


FIG. 6: To better display the distinction between H and V tree result, I chose a larger smoothing factor.

---

[1] M. Bleyer and M. Gelautz. Simple but effective tree structures for dynamic programming-based stereo matching. In VISAPP, volume 2, pages 415C422, 2008

[2] Bleyer, M., Rother, C., Kohli, P., Scharstein, D., Sinha, S. (2011). Object stereojoint stereo matching and object segmentation. In Conference on computer vision and pattern recognition

[3] Ashutosh Saxena, Sung H. Chung, Andrew Y. Ng, 3-D Depth Reconstruction from a Single Still Image