Vertex Classification for Segmented Images

Evan Rosen

with Stephen Gould and Rick Fulton

Abstract. Depth estimation is an essential component for scene understanding. With this in mind, we take a novel approach to 3D depth reconstruction which leverages off of the highly structured nature of our environment. We believe that high level structural cues, such as planes, edges, and vertices play an important role in resolving depth ambiguities and attempt to evaluate the significance of such factors. This approach is partly inspired by the constraint satisfaction of Waltz's algorithm in "Understanding of Line Drawings with Shadows" (1975). Waltz was concerned with finding the correct 3D interpretation of a line drawing by using the geometric constraints which connect edges place on each other. But in applying this method to real images, a number of issues arise. It is not clear where the 3D discontinuities lie in the image and as a result we cannot use hard constraint satisfaction to solve for the true depth. We attempt to distill the basic intuition and capture the geometric relationships between adjacent parts of an image probabilistically and generate a global depth reconstruction using a CRF. This project is largely a theoretical exploration of this approach and here we will specifically discuss the subtask of vertex classification.

1 Model

One of the first questions for this model is how to represent the relationship between parts of the image and the corresponding 3D patch which we are trying to estimate. Our initial approach was to subsample the image into a smaller number of rectangular superpixels which we map to 3D by assigned a depth. Each superpixel is assumed to be planar and parallel to the image plane. The advantage of this parameterization is that we only need to estimate one variable for each superpixel. The drawback is that we need to operate at a reasonably high superpixel resolution in order to avoid the smoothing of edges which do not fall on the superpixel grid. This is particularly detrimental to our attempts to use edge relationships to constrain the depth of these superpixels because we cannot even explicitly represent edges but rather have only indirect representation in the form of the depth differences between adjacent superpixels. Though this approach was a useful first attempt, we have settled upon a slightly different model.

Each superpixel is now parameterized by a normal vector to indicate orientation and an offset to indicate distance from the camera origin to the superpixel centroid. And, instead of rigidly defining the superpixel shape beforehand, we

use a segmentation algorithm by Ren and Malik [1]. An important property of this superpixel partitioning is that by oversegmenting the image we expect to minimize the number of 3D discontinuities which do not fall on superpixel boundaries at the cost of including many segment boundaries which do not correspond to depth discontinuities. If we wish to use edges as an explicit part of our model, the presence of these false positives is much less damaging than that of false negatives. It is worth noting that such an approach assumes a high degree of approximate planarity in the real world, for it does not capture smooth depth discontinuities.

Thus we have set of superpixels constrained in such a way to match the notions of smoothness, colinearity, planarity, and the presence of corners. In other words, we penalize a possible configuration of superpixel orientations and offsets depending on whether the configuration respects our estimates on the presence of edges and vertices. For example, if we had two adjacent superpixels with identical uniform color and there was no intervening edge in the image, we would like to constrain any global configuration to assign these two superpixels similar normals. In the case of vertices, if we had three adjacent superpixels which met at a perfect 'Y', and which we had other reasons to believe looked like a convex corner, we would like to constrain any global configuration to assign each of the normals to be perpendicular to each other in the correct way. This report focuses on finding exactly what does count as a good indicator of whether there exists a certain type of vertex between adjacent segments.

2 Vertices

In our attempt to learn to identify various vertex configurations we consider only vertices formed by three adjacent superpixels. Though there are many cases of four intersecting edges and some vertices with even more, we address the case of three superpixels because it is simpler and because it is not clear that the predictive benefit of considering such vertices outweighs the computational costs of introducing larger cliques in the CRF. We also attempt to classify each edge into one of the following categories: convex, concave, A obscures B, B obscures A, and phantom (where A and B indicate order as one proceeds clockwise around a vertex). We will number these vertex labels for notational convenience so that the set of possible edge labels is $\{1, 2, 3, 4, 5\}$Thus each each vertex can be characterized by a combination of vertex types. With this view in mind we try to learn how to classify each edge into one of the 125 3-permutations of edge types.

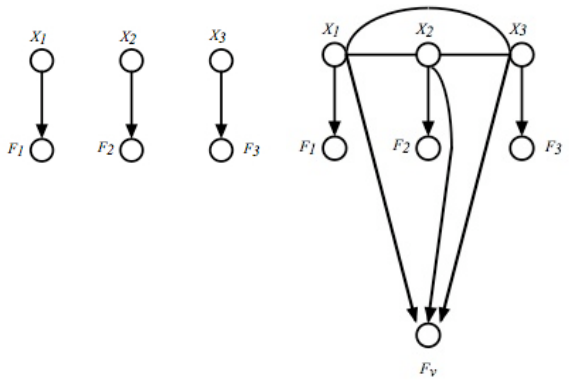
This yields a simple graphical model for classifying vertices. A simple model assumes that each edge label is chosen independently which then generates a set of edge-specific features with some probability distribution, $P(F_i|X_i)$. Thus, the most likely assignment is just the set of labels which maximize each $P(F_i|X_i)$ or equivalently, the argmax of:

$$P(\vec{X}|\vec{F}) \propto \exp(\phi_1 + \phi_2 + \phi_3) \tag{1}$$

where $\phi_i = \ln P(X_i|F_i)$. However, assuming that the edge features alone dictate the vertex classification, and that any combination of edges is equally possible, as long as the edge features are explained by the labels, misses the inherent constraints on which types of edges may join in a vertex. We know that the edges do in fact constrain one another, as Waltz's algorithm demonstrated. To capture the constraint of consistency between edge labels, we will introduce a set of vertex-wide features, F_v which is also generated by $\vec{X}$. This yields the graphical model (b). Thus to classify a new vertex we calculate the posterior probabilities, $P(F_i, X_i)$ for each edge i and each possible label $X_i \in \{1, 2, 3, 4, 5\}$ as well as the posterior probability $P(F_v|\vec{X})$ for all values of $\vec{X}$ (i.e. combinations of edge type) and find the value of $\vec{X}$ which maximizes their product. Or equivalently, the argmax of:

$$P(\vec{X}|\vec{F}, F_v) \propto \exp(\phi_1 + \phi_2 + \phi_3 + \psi) \tag{2}$$

where $\psi = \lg P(\vec{X}|F_v)$. However, because it is not clear how much each of these ψ_is and ψ should account for our final vertex classification (i.e. how important vertex wide features are, or how much more useful one edge estimate is over another), we need to learn weights for these terms. To simplify the problem, and because of the way in which edge indices get assigned, we have no reason to believe that edges should be weighted differently and thus we only need to find the weight for ψ, which we will call θ. To do this, we use 5-fold cross-validation. To compute $\phi_i = \ln P(X_i|F_i)$ by Bayes' rule, we find



(a) Naive Model (b) Consistent Edge Model

$\ln P(F_i|X_i)P(X_i)$ which are simply the result of tallies of our training set. And to compute $\psi = \lg P(\vec{X}|F_v)$ we find $\ln P(F_v|\vec{X})P(\vec{X})$ which is also derived from tallies from our training set. Specifically, $P(F_i|X_i)$ and $P(F_v|\vec{X})$ are given by a set of gaussian distributions for each labeling X_i or $\vec{X}$ fit to the training vertices whose ground truth labeling matched X_i or $\vec{X}$. One issue arises due to the large number of vertex classes. We use smoothing to deal with the sparsity of our

data set and the fact that we do not have observed vertices for all of the possible edge combinations, and nor should we, as some are geometrically impossible.

3 Features

By the above model, we have two different feature sets: those corresponding to each edge, F_is, and those corresponding to the vertices F_vs. For the F_vs we just used coordinates of the vertex and estimates of the angles and lengths (in number of edge pixels) of each edge. The edge angles were computed by taking a weighted average of the angle that each edgel (pixel closest to the edge) made with the vertical. For the F_is we used the edge angle and length as above, an estimator of edge linearity, an estimator of edge strength, and the differences and respective values for basic statistics of the adjacent superpixels. To estimate linearity we used the absolute value of the covariance of the edgel coordinates. To estimate the edge strength we took the mean color difference between adjacent pixels which belonged to different superpixels. We had initially used a large set of randomly sampled templates with which we took the cross correlation for various points associated with the edge, but these cross-correlations were extremely time-consuming to compute and we do not have sufficient labelled data to test the features and have been left our in our initial model.

4 Results

With respect to the vertex classification we do not have meaningful results for their effect on the global depth estimation. Nor do we have valuable results for just the classification of vertices we are finding. The data set must be labelled by hand and moreover must be reasonably well balanced across different edge and vertex classes which has made getting enough data to meaningfully predict a 125 class classifier exceedingly difficult. We are still largely predicting the most frequently occurring vertex type across all actual vertex classes.

5 Future Work

For one, it has become apparent that using a data set with ground truth depth maps might be valuable in allowing automatic labeling of much larger training sets. Additionally, the current data set is largely for outdoor images, which makes certain vertex types very rare (such as convex-convex-convex). Moreover, once we can evaluate the effect of vertex classification on the global depth estimates, we can experiment with various segmentations will change the classification accuracy of vertices but which may improve the overall depth reconstructions. Also, there is still a fair amount of work to be done in exactly how to build potentials given information about vertex types. This will amount to implementing some of the actual constraints used in Waltz's algorithm.