Inferring 3D Scene Structure from a Single Still Image

Gabriel Yu and Jing Chen (Advised by Ashutosh Saxena)

1. Introduction

In this project, we revisit the problem of constructing 3D structures from single still images. We build upon previous work done by Saxena, Sun and Ng [1] by improving on the inference techniques used in their algorithm, with the goal of producing 3D models that are more quantitatively accurate, as well as more visually pleasing.

One area of improvement in the existing algorithm is the penalty function used during MAP inference of plane parameters. When inferring 3D models from single still images, the penalty function is used to enforce constraints such as connectedness, co-planarity, and co-linearity. Properties of the penalty function consequently determine whether the transition between two planes in the 3D model is smooth or sharp. The current penalty function, the L1 norm of the error, does not prefer either a smooth transition or a sharp transition. As a result, the resulting 3D models often have walls sloping away from the ground, rather than standing straight up. Our goal was to find a suitable penalty function that prefers a sharp transition over a smooth transition.

Another area of improvement is to make use of user-provided scribbles during inference so that the result is closer to how a human perceives the image. The idea is that these scribbles could be used to help the algorithm make immediate improvements in inferring the 3D model of a given image.

A subproblem to using scribbles during inference is finding the "correct" scribble, which is the one that would give optimal performance in the new inference algorithm. We devised an algorithm that used supervised learning to make use of the coarse scribbles provided by users to infer information about other areas in the image that have not been filled in by scribbles.

This report is divided into three portions, each describing the work that has been done in the three areas described.

2. Penalty Function

The penalty function that is currently used, the L1 norm (approximated by $\gamma(x)$), does not prefer either a smooth transition or a sharp transition. The goal of this portion of the project was to find a penalty function that prefers a sharp transition over a smooth one, in order to bring foreground objects to the front, for images where the object would slope into the background when using the original penalty function.

Current Progress

a.) **L2 norm.** We started out by implementing the L2 norm, to gain familiarity with the system. This gave a result that was not significantly different from the original, although it made more mistakes than the original in certain test images.

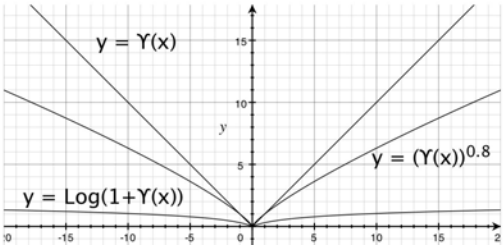

Figure 1. Graph of the penalty functions

b.) **Log(1 + γ(x)).** We saw encouraging results with this penalty function. For one particular test image, the result was clearly better than the original results (see Figure 2): rather than sloping inward to the background, a bush in the foreground stayed vertical in the foreground. This shows that the Log(1 + γ(x)) function performs better than the original in certain cases. For images where there was no clear improvement, the results from this function were at least as accurate as the original

Figure 2. **a) Original** **b) Log(1+γ(x))** **c) $(\gamma(x))^{\eta}$**
The images above show the improvement that the new penalty functions give over the original

results, although the two functions often made mistakes in different areas.

c.) **$(\gamma(x))^{\eta}$.** We saw similar results with this function: in the same test image that was mentioned above, the bush also came to the foreground. In examples without such obvious improvements, its performance was roughly the same as the Log(1 + γ(x)) function and the L1 norm.

d.) **Min(a, γ(x)).** This function is not differentiable, but we used the fact that:

$$Min(a,b) = \frac{a+b}{2} - \frac{|a-b|}{2}$$

Using gamma(x) to approximate the absolute value function again, we came up with an approximation of the Min function that was differentiable. Unfortunately, the results for this function were not usable due to numerical problems, possibly caused by a badly conditioned Hessian close to x = 0.

Numerical Stability

The new penalty functions resulted in numerical inaccuracies at lower values of β than the original penalty function, where higher values of β give more accurate approximations of the absolute value function. The initial solution to cap the number of iterations, thus capping the value of β. However, this did not work for all images, since the threshold at which the numerical inaccuracy appeared was different for each image.

To deal with this, we instead checked when the Hessian matrix of the penalty function became singular while incrementing β, then capped β at the value before the step where the matrix became singular, and reset β to that value for the next iteration. This meant that the penalty function would work for all images, while maximizing β for each image.

This method also proved to be useful for the user-provided hints portion, since the increased weights caused numerical inaccuracies to appear at lower values of β, even when using the original penalty function.

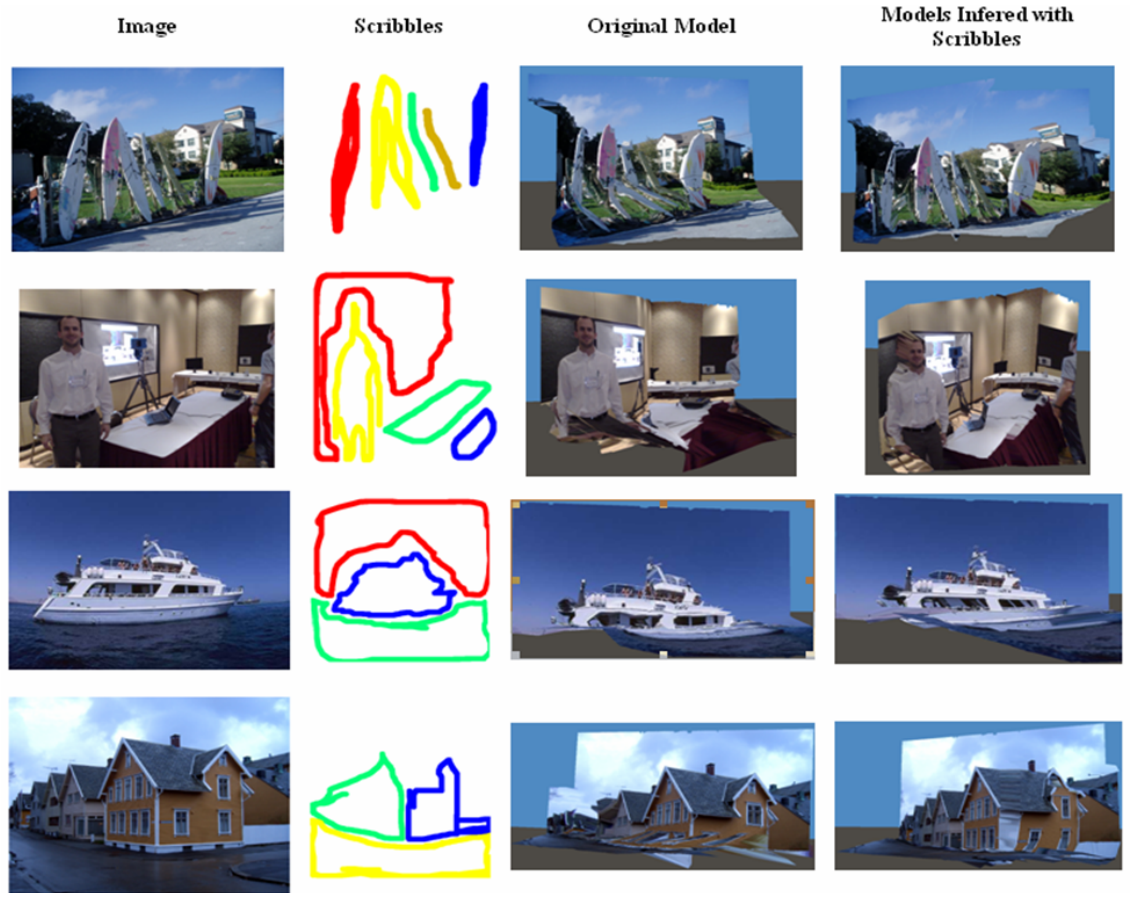

Figure 3: Typical results from our algorithm. The first three models are considered "good". For the last model, while the plane of the front of the house is separated from its side planes as expected, its orientation is incorrect.

Edge Maps

Checking whether a change to the algorithm worked was initially done by visually inspecting the resulting 3D model for each image. However, this method is inaccurate and subjective, and it becomes easy to miss minor differences. To attempt to make testing new methods more objective, we generated edge maps for each result.

These edge maps were generated from the residuals that resulted from the penalty function. The higher the residual for a given pair of neighboring superpixels, the more likely it was that algorithm was classifying the edge between the pair as a boundary between two separate planes. The resulting edge maps from trial runs could be compared much more easily and objectively than visually inspecting the 3D model.

Mixed Penalty Functions

We also tried using different penalty functions for different properties (co-linearity, co-planarity, connectivity), with the idea that a penalty function may work better for one property than another. However, the results showed no significant improvement over using either the Log function or the η-norm. There were minor differences in the resulting 3D structure, but the differences were not clearly better or worse than the results from using non-mixed penalty functions.

3. Inference with Scribbles

For this portion of the project, an online drawing tool was provided to let users scribble colored lines on top of their images – neighboring superpixels with the same scribble color are interpreted as being on the same plane, while

neighboring superpixels with different color should be on different planes.

Since the scribbles drawn by users are often coarse and imprecise, we have to first pre-process the scribble image. We do so by first filling in any holes encompassed by the scribble lines. We then apply a non-linear filter to the scribble image, in which each pixel takes on the most frequent color in its 5x5 neighborhood. This essentially expands the user-provided scribbles with inferred auxiliary colors.

We then hand-tuned the weights of the co-planarity and connectivity constraint functions used in Saxena, Sun and Ng. In order to enforce co-planarity between two superpixels, the weights of the co-planarity constraints are set to high values. To enforce non-co-planarity, weights to both the co-planarity and connectivity constraints are set to low values, because connectivity often indirectly enforces co-planarity. In addition, a lower weight is given to neighbors with auxiliary colors than to those with actual scribble colors.

Results and Discussions

We randomly picked 20 images uploaded to the website and subjectively selected those that are not visually pleasing. We then added scribbled lines to the images, and used the algorithm to re-infer the 3D models.

To get a fair qualitative evaluation of the result, we should have collected ratings from users in the websites, but we were unable to publish the latest algorithm to the web in time to collect statistics. However, by visually comparing the original model with the new models, the modified algorithm improves the visual quality of the model in more than half of the cases.

In all cases, the algorithm is successful in making planes co-planar, while separating planes that should not be co-planar. However, since the scribble does not explicitly provide hints about the orientation of the planes, the algorithm sometimes makes certain planes that are correct in the original model co-planar attached to a plane with an incorrect orientation, thus magnifying the original problem (see the last sample in Figure 3). In other words, the algorithm works well when the majority of the scribbles cover planes whose orientation was correct in the original model.

For complete results, please visit:
http://ai.stanford.edu/~gbrlhkyu/images

4. On-the-Fly Learning of Scribbles

The 3D models re-inferred using the scribbles are only as good as the scribbles provided by users. Therefore, as described in the previous section, we tried using various tricks to modify the scribbles. However, the resulting scribble image is still highly dependent on original scribble.

Therefore, to make better use of the scribbles, we experimented using supervised learning to intelligently infer the scribble on-the-fly.

Learning Algorithm

The learning problem is formulated as follows: for each pixel on the image not scribbled by users, we want to use the information given by the user's scribble to infer a scribble color that should be assigned to this pixel. This is a multi-class classification problem, and we reduce it to multiple binary classification problems.

Specifically, for each scribble color *c*, the input features are the RGB value of a given pixel, and the class labels are 1 if color c is scribbled on the pixel and 0 if otherwise. The training data are all the pixels in the images that are currently filled in with scribbles by users. We then model the relationship between the input features and class label using logistic regression with gradient descent. The resulting learned hypothesis will then output the probability that the given pixel belongs to color *c*.

After learning the hypothesis for each scribble color, we then use it to model the probability that a given pixel should be assigned a particular color, for each pixel that was not filled in by the user. These pixels are then assigned the color with the highest probability for that pixel.

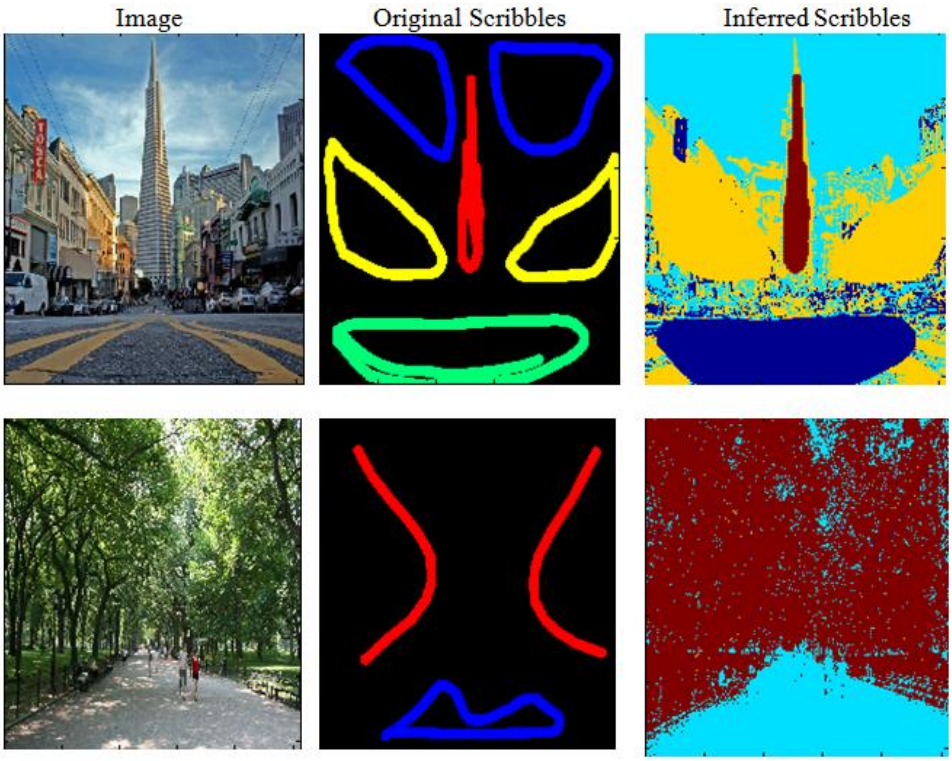

Figure 4. Selected results from our algorithm. As seen in the second results, the algorithm performs wells if planes contain the same color.

Results and Discussions

We randomly picked 5 sets of images and scribbles from the websites and use them to test the learning algorithm. We then visually evaluate whether the inferred scribble covers regions that should or should not be co-planar in the human eyes.

As expected by the simplicity of the features, the algorithm works well only if each plane has the same colors. However, it will not work well if the planes contain different colors (See Figure 3).

There are many way to improve the current algorithm. We can use HGV or YCbCr values instead of RGB values, which are known to give better performance for image processing. We could also increase the number of features by looking at properties of the neighboring pixels.

5. Conclusion

The goal for this project was to improve the performance of the original system, especially the mistakes that appeared frequently for typical images. We succeeded in finding a penalty function that gave a noticeable improvement in performance for samples where a foreground object appeared attached to the background. We also succeeded in using user-provided scribbles to improve the quality of the result. To improve the results even further for cases where the provided scribble is very coarse or inaccurate, we implemented a method for inferring more comprehensive information from the given scribble. The end result is a system that gives better results on several samples, without degradation in quality for samples with no clear improvement.

6. Acknowledgements

We would like to thank Ashutosh Saxena for the many helpful ideas and thoughtful advice that he provided throughout the course of this project.

7. References

[1] **Learning 3-D Scene Structure from a Single Still Image**, Ashutosh Saxena, Min Sun, Andrew Y. Ng, In *ICCV workshop on 3D Representation for Recognition (3dRR-07)*, 2007