3D Reconstruction Of Occluded Objects From Multiple Views

Cong Qiaoben Stanford University

Dai Shen Stanford University

Kaidi Yan Stanford University

Chenye Zhu Stanford University

Abstract

In this paper we present three different algorithms to approximately reconstruct certain parts of an object that can only be seen in one view due to occlusion. Single view 3D reconstruction is generally hard to carry out due to the under-determined nature. Nevertheless, we make certain observations on the general shape of the target object and make use of other parts of the object which we can accurately triangulate. We test our algorithms on different objects and conduct both qualitative and quantitative analysis on results we get.

1 Introduction

Taking a video camera and walking around an object to automatically construct a 3D model has been a long-standing dream in computer vision. What seems unreachable 20 years ago is no close to our abilities thanks to the many discoveries in 3D geometry and image matching over the years. The various 3D construction techniques have also been able to take advantage of the developments in digital cameras and the increasing resolution and quality of images they produce along with the large collections of imagery that have been established on the Internet. One of the famous application is the Photo Tourism project such as the reconstruction of the Coliseum in Rome as shown in Figure 1.

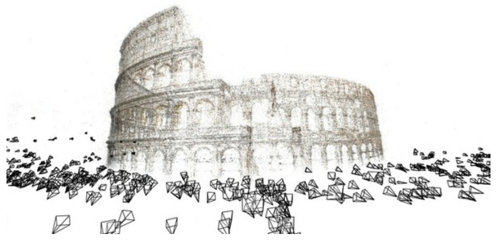

Figure 1: Reconstructed 3D model of the Coliseum

However, what currently hinders the use of more 3D information in other domains seems to be the specialization of 3D reconstruction methods to specific settings. State of the art multi-view stereo methods are able to produce accurate and dense 3D reconstructions. Unfortunately, many of them are tuned to settings like the Middlebury benchmark (Kazhdan et al., 2006), where the camera poses are known, silhouette information can easily be retrieved and the number of views are adequately large (usually more than 20). In this paper, we aim to remove these requirement of context information in the standard multi-view methods through three approximation methods - patch, symmetry, and ratio approximation.

The rest of the paper is organized as follows. Section 2 reviews the inadequacies of some of the existing approaches to the problems and our solutions. Section 3 gives a detailed explanation of our methodology as well as the underlying principles of our approximation methods. Section 4 shows the results of our methods on various objects subject to various camera settings and illuminations. Section 5 conclude our work.

2 Related Work

2.1 Existing Literature

Apart from the above mentioned works and the references within the Middlebury benchmark, there are a couple of works, which are closely related to the approaches we present in this paper.

Yezzi (Yezzi et al., 2003) proposes a method where the cameras are refined during the dense reconstruction process. However, their silhouette based approach is not well suited for general image sequences. It works best for textureless scenes where the background can be clearly separated from the object.

Furukawa (Furukawa et al., 2008) describes an iterative camera calibration and reconstruction approach. A large set of patches is used to refine the camera poses. Vice versa the refined camera poses are used to improve the reconstruction. To compute the dense surface they use the method from Kazhdan (Kazhdan et al., 2006). The resulting surface is fitted to the patches by the solution of a Poisson problem. As the spatial Poisson problem does not take photoconsistency into account, results suffer in areas not covered by sufficiently many patches.

2.2 Our Contribution

We aim to address the issues associated with the approaches mentioned above. In particular, the approximations we propose enable us to obtain an accurate object structure without the need for a background of clear contrast nor the need for a large number of photos to provide a sufficient coverage for each possible patch. In addition, our symmetry approximation method provides a simple but effective way of solving the reconstruction of symmetric objects. This method, in fact, goes beyond the problem of reconstruction, and is potentially also applicable in image recognition and even graphical reasoning.

3 Approach

3.1 General Methodology

In general triangulation, we solve the following simultaneous equations to find the 3D location of the unknown point P.

$$\begin{cases} M_1\mathbf{P} = p_1 \\ M_2\mathbf{P} = p_2 \end{cases}$$

M_1 and $M_2 \in \mathcal{R}^{3\times4}$ and the unknown point $P \in \mathcal{R}^{4\times1}$. However, in our scenario only one of the two observation p_1 and p_2 is observable. Thus we have 4 degrees of freedoms and only 3 scalar equations, resulting in an under-determined linear equation system.

Therefore, the remaining part of the paper explores different heuristics we can use in order to give one more constraint (one more scalar equation) with respect to P.

As a preliminary step, we first find matching points on different images and apply triangulation to find their 3D locations. This gives us some basic information about certain parts of the object which can be used in our methods.

3.2 Technical Details

For this part we are going to elaborate on the main heuristics we adopt in approximating the 3D locations of points that can only be seen in one view.

3.2.1 Patch Approximation

In this section we employ the techniques of tessellation in computer graphics to help us recovering the shapes of our occluded object. We subdivide the surface of the object into planar polygons, called tessellators or patches, and simulate the smoothness of the surface with non-trivial amount of such patches, see figure 2. The idea is as follow: we first assign each point on the object to a planar polygon, then we calculate the position of the polygon in 3D space, and afterwards, we recover the exact location of each point by finding the corresponding 3D points in the polygons.

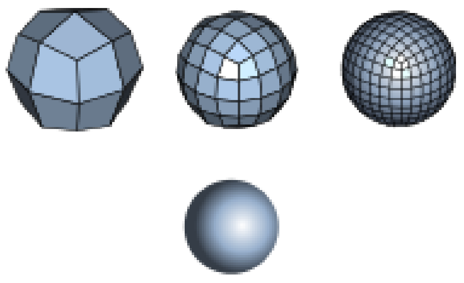

Figure 2: Tessellation of a Smooth Object

Before we proceed with our discussions we need to define some terminologies that we will use in this section and the remaining part of the paper. A polygon, or a patch, is a 3D planar tessellator. Its projection into 2D image is a region. Furthermore, if two points lie on the same polygon, their projections lie within the same region. Full view image refers to the image in which we can see the overall shape of the object, as in figure 6(b) whereas the occluded image refers to the image in which object is partially occluded by obstacles, as in figure 6(a).

1. **Edge detection**

   Consider the intersection of two patches. Since the points belong to two different regions and lie on two disparate planar polygons, we shall expect surface orientation discontinuities near the region, one of the main sources of image brightness discontinuities in the perceived images, i.e. edges. Therefore we shall be able to detect some really thin edges along the corresponding regions intersections.

Notice that these edges are delicate as the object surface is rather smooth and the changes of gradients are relatively mild. Thus we use Canny Edge Detector by drastically lowering the two thresholds to capture such minute parts. We apply the canny edge detection techniques to the full view image near the object region and obtain the preliminary result for the edge map. For an example of how it looks like, see figure 11 in Experimental Results section.

2. **Edge completion**

   Nonetheless, some of the edges are discontinued as there are nontrivial amount of noises in the images. Post-processing is warranted as to connect dots and complete the separation of planes. We employ two methods in achieving our goals: (1) a methodology similar to non-max-suppression + edge continuation in the canny edge detector but with more leniency, in that we would allow slightly more deviations from the suggested path according to the gradient directions and (2) a window-based geometric methodology that scan through each patches of the images and connect two dots if both have been labeled edges and one has a gradient direction is tangent to vector connecting these two dots. For instance in figure 3, the two blue blocks were designated as edge block by Canny detector and the one on the left bottom has a gradient direction perpendicular to the line segment connecting the centers of the two blue blocks. Thus we mark all of blocks in between as potential edge blocks.

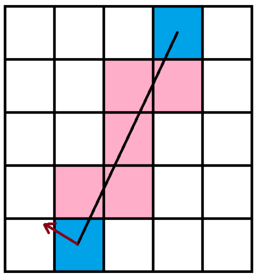

Figure 3: One possible scenario in which two dots will be connected

   Finally we combine these partial edge maps from Canny Edge Detection, figure 11 and from Edge Completion, figure 12, and produce the full view object edge map in figure 13 by restricting the total region counts in the window.

3. **Region assignment**

   With the edge map in hand, we proceed to assign each image pixel into a region. We know for sure that pixels enclosed in the same regions must belong to the same polygons. But we are not guaranteed whether two different regions correspond to disparate polygons or not. Notice that discontinuities of surface orientations are not the only reasons for edges in the images. Other artifacts such as change in material properties (color, reflectance and etc.) and changes in scene illuminations may also result in edges in the perceived images. We need to account for these artifacts. [1]

4. **Reconstructing polygons**

   We import the results from the preliminary triangulation. In particular, to account for the inaccuracies incurred due to floating point calculations and projection errors, we argue that for each triangulated point, close-by neighbors [2] should all correspond to the same 3D points. We claim that a region is properly defined if and only if we could find three or more points on the corresponding patches. For each of these proper regions, we find the plane equation of its corresponding 3D patch by interpolating the 3D locations of the points sitting in the region.

5. **Recover points within a polygon**

   This is the final step of our reconstruction process. For any pixel x in the full view image, if it belongs to a region that is properly defined, i.e. we know the plane equation of the corresponding polygon D, we estimate the 3D location of x, which is denoted as $\mathbf{X}$, by solving the following minimization prob-

[1] Within the scope of this project, we decide to manually combine regions that should have been grouped together. But if time allowed we could have explored more in details and recover the information through material properties, shadowing and etc. (Bell et al., 2015)

[2] Here we impose a 5-pixel-by-5-pixel window to determine nearby neighbors

lem.

$$\begin{aligned} \underset{\mathbf{X}}{\text{minimize}} \quad & \text{distance}(\mathbf{X}, D) \\ \text{subject to} \quad & P\mathbf{X} = x \end{aligned}$$

Where distance$(\cdot, \cdot)$ is the distance function from a point to a plane, and P is the projection matrix associated with the full view image.

3.2.2 Symmetry Approximation

In this section we make the assumption that the target object is nearly symmetric. We also assume that the full view image is taken with the target object facing right in front of the camera, and the camera projection is roughly affine.

The high level idea of this approach is to find the symmetry plane, and then estimate the distance to the symmetry plane based on the distance we retrieve from the full view image. When the symmetry plane is perpendicular to the image view, the symmetry plane is projected into a symmetry axis. Therefore, the problem of finding the symmetry plane is equivalent to finding the symmetry axis in the image in this case.

Our symmetry detection algorithms consider two cases. The first algorithm tries to resolve the simplified version of symmetry detection: it assumes the image is taken parallel to the plane the object is placed. Therefore, this symmetry axis is a vertical line in this case. The second algorithm is more generic, because it will find the symmetry axis of any arbitrary near-symmetric image, given that the symmetry plane is perpendicular to the image view.

Both algorithms first use canny edge detector to find edges of the image. For convenience, the mask is denoted by M_E. canny edge detection tracks edges by suppressing all the other edges that are weak and not connected to strong edges using two thresholds (upper and lower). In our case, the lower threshold is particularly interesting because it suppresses weak edges. Here we set the lower threshold to be sufficiently large to filter away as much weak edges as possible; this also reduces the amount of computation for the symmetry detection algorithm, which is explained next.

1. **Symmetry Detection algorithm for vertical symmetry axis**

   This algorithm takes each row of M_E and applies a Gaussian filter. This generates the vector $\rho'_i(M_E)$. Then it finds the x_0 value that maximizes the sum of the convolution of each row of M_E and its "reflection" across $x = x_0$, $\rho'_{i,x_0}(M_E)$.

   $$x = argmax_{x_0} \sum_i \rho'_i(M_E) * \rho'_{i,x_0}(M_E)$$

   The time complexity of this algorithm is $O(\sigma \text{size}(M_E))$, where σ is the standard deviation of the Guassian filter. Higher σ means higher error tolerance in symmetry matching.

2. **Symmetry Detction algorithm for arbitrary symmetry axis**

   This algorithm uses Hough Transform to find the symmetry axis.The symmetry axis could be denoted as

   $$x \cos\theta + y \sin\theta = \rho$$

   . We could find the potential symmetry axis by looking at any pair of points (x_1, y_1), (x_2, y_2) and assuming they are symmetric. This would give us:

   $$\tan\theta = \frac{y_1 - y_2}{x_1 - x_2}$$

   $$\rho = \frac{x_1 + x_2}{2}\cos\theta + \frac{y_1 + y_2}{2}\sin\theta$$

   The number of bins we use is 1000 for ρ and 20 for θ. The bin with the highest frequency gives us the best estimate.

   After finding the symmetry axis, we could use the triangulation method to find the symmetry plane S, by picking SIFT keypoints close to the symmetry axis and interpolating these points. Also using SIFT, we could get correspondence between the the points we want to estimate 3D locations for and their symmetric counterparts. This is done by flipping the full view image horizontally and then perform SIFT between the original full view image and the flipped one.

   For each symmetric pair (x, y), the distances to the symmetry plane should be the same for x and y in 3D space. Therefore if we know the 3D location of y is $\mathbf{Y}$, we can calculate the distance from y to S. The 3D location of x, which is denoted as $\mathbf{X}$, can thus be estimated by solving the following minimization problem.

$$\begin{array}{ll}\underset{\mathbf{X}}{\text{minimize}} & |\text{distance}(\mathbf{X}, S) - \text{distance}(\mathbf{Y}, S)| \\ \text{subject to} & P\mathbf{X} = x\end{array}$$

Where P denotes the projection matrix of the full view image.

3.2.3 Ratio Approximation

The last heuristic is based on the fact that affine projections preserve the ratio of lengths for parallel line segments. Given an affine transformation $A(\mathbf{p}) = M\mathbf{p}+\mathbf{b}$, the ratio of lengths between parallel line segments is invariant. This is true since two parallel line segments $\mathbf{l_1}$ and $\mathbf{l_2}$ has the relationship $\mathbf{l_1} = k\mathbf{l_2}$, and after affine transformation the ratio of lengths is

$$\frac{\|M\mathbf{l_1}\|}{\|M\mathbf{l_2}\|} = \frac{\|kM\mathbf{l_2}\|}{\|M\mathbf{l_2}\|} = k = \frac{\|\mathbf{l_1}\|}{\|\mathbf{l_2}\|}$$

In reality the camera projection is not affine - however for the sake of approximation we assume the camera is close to affine. If we can find two parallel line segments from the image which contains the entire object, then we can compute the ratio of lengths of these two segments. Assuming an affine transformation, the ratio of lengths of these two line segments should remain the same after we project them back into 3D world space. If we happen to know the length of one line segment in the 3D world space, then we can calculate the length of another one, thus potentially giving us more new constraints on the unknown point location.

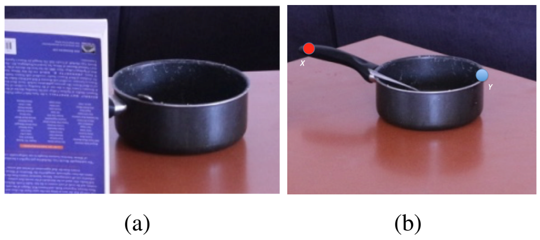

Figure 4: Example of pot. Figure 4(a) shows only part of the pot while figure 4(b) shows the entire pot. Our goal is to reconstruct the part that is occluded by the book.

Here is an illustration using the example of a pot. Suppose we want to estimate the 3D location of point x, which is indicated as a red spot in Figure 4(b). Clearly the pot is asymmetric, and thus we cannot apply symmetry approximation to this object; patch approximation also faces obstacles as we need to know at least 3 more points on the pot handle, which in this case is impossible to know given the occlusion in Figure 4(a).

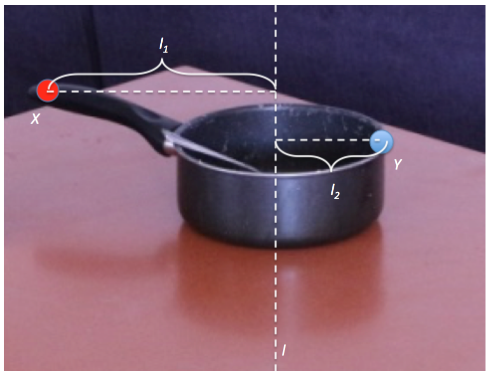

Figure 5: Graphic illustration of applying ratio approximation

In order to apply ratio approximation, we first find a vertical line l in figure 4(b) using the same way as we find the vertical symmetric axis in symmetry approximation. Notice that in this case we use the same method but the line we find no longer carries the meaning of a symmetric plane - it only serves as a reference line. After that, we choose a SIFT point y that is located on the opposite side of l from x. From here we can calculate the distance from x to l and from y to l. Denote them as l_1 and l_2 respectively.

The rest part is similar to what we do in symmetry approximation. Again, we assume that we can find at least 3 SIFT points on line l which we already know the 3D locations - thus when l is projected back into 3D space we can approximate this plane S by interpolating these 3 SIFT points. We also find the 3D location of point y, and hence we can calculate the distance from Y to plane S in the 3D space. Let it be l_2'. If the distance from the 3D location of point x to plane S is l_1', then we have the following ratio of length equality:

$$\frac{l_1'}{l_2'} = \frac{l_1}{l_2}$$

l_1, l_2 and l_2' are all known. Thus we can compute l_1'. Now we know the distance from the 3D location of x into plane S, and therefore we have one more constraint which allows us to solve for actual X.

4 Experiments

4.1 Experimental Setup

For the experimental setup, we need to take two images of the same target object. The full view image contains the entire front face of the target object unoccluded; while another image only contains the target object occluded by other objects. Our goal is to use these two images and reconstruct the part that is occluded by other objects.

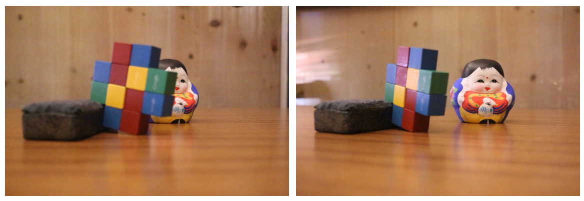
(a) (b)

Figure 6: Experiment 1. The target object is the Chinese doll.

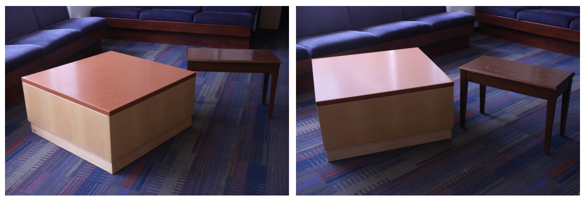
(a) (b)

Figure 7: Experiment 2. The target object is the chair.

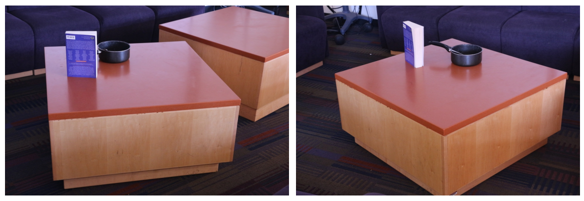
(a) (b)

Figure 8: Experiment 3. The target object is the pot sitting on the table.

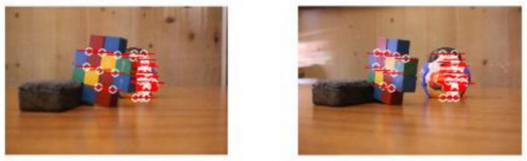

Figure 9: SIFT matching points we find for experiment 1. The red lines are the epipolar lines at each point.

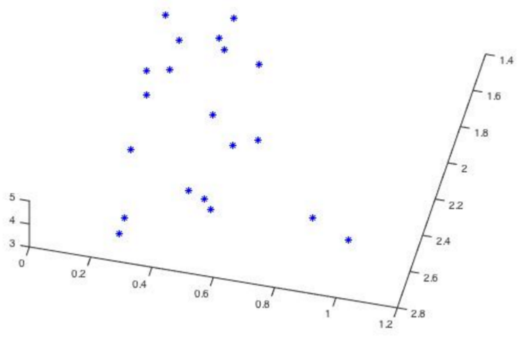

Figure 10: Triangulation result for SIFT matching points in experiment 1

In order to evaluate our reconstruction result, for each experiment we also take a third image, which also contains the entire front face of the object.

By including the third image, we are able to use triangulation to accurately find 3D locations of all points that we see on the front face of the target object. This is the baseline result for our experiments - when implementing our heuristics we are only going to use the images shown in Figure 6, 7 and 8. For points we have estimated, we are going to compare their estimated locations with the actual locations found from triangulation.

4.2 Experiment Results

4.2.1 Triangulation Result

The first part of the experiment is to find 3D locations of points that appear in both images. This provides some basic information about certain parts of the object and is needed for each of the heuristic we propose.

We calculate the intrinsic camera matrices using single view metrology. After that, we use SIFT to find matching points in the two images. For these matching SIFT points, we apply projective triangulation in structure from motion to estimate the rotation and translation matrices R and T of cameras, and therefore find the 3D locations of those matching SIFT points (For more details, refer to CS231A HW2 Question 4). Figure 9 and 10 show the resulting 3D points we find for experiment 1.

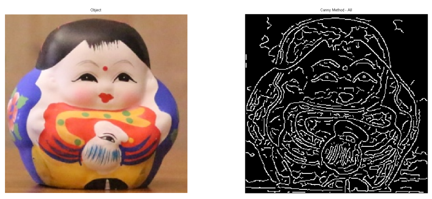

Figure 11: Pure Canny Edge Detection Result with Canny Bound as 10^{-4} and 0.05

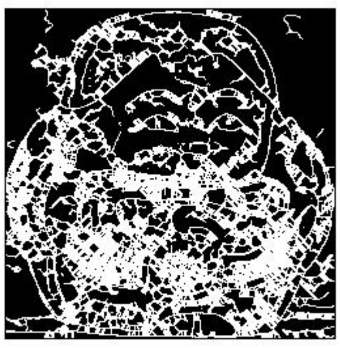

Figure 12: Edge Completion Result

4.2.2 Patch Approximation Result

Figure 13 shows the resultant full edge map we get for experiment 1, using edge detection and edge completion technique we mentioned in Technical Details section.

Using Figure 13, we are able to assign regions for the target object. Figure 14 shows the region map we get experiment 1.

With region map, we are able to approximate the plane equation of some of the regions if we already have three or more SIFT points with known

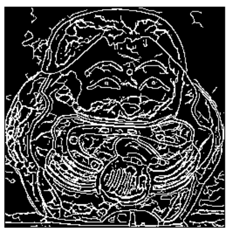

Figure 13: Combine Edge Detection Results and Edge Completion Results

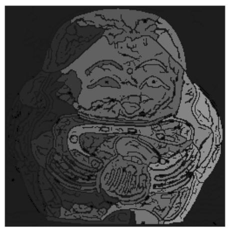

Figure 14: Region assignment for experiment 1

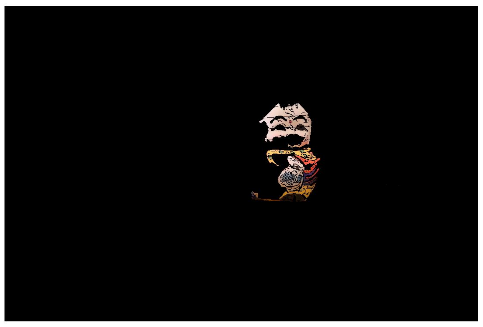

Figure 15: Reconstruction result with patch approximation on experiment 1

3D locations sit on the regions. Thus we are able to recover the entire polygon by estimating the 3D locations of each pixel in that region. The reconstruction result is shown in Figure 15 and 16.

Note that certain part of the object is missing, since the region at those parts do not have sufficient number of SIFT points for us to approximate the plane equations. We also try patch approximation for experiment 2 and 3 - however the results are mostly unsatisfactory because the number of patches on the target object which we can approx-

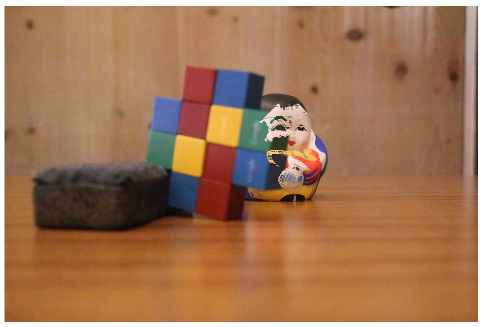

Figure 16: Reconstruction result with patch approximation on experiment 1 projected onto the occluded image

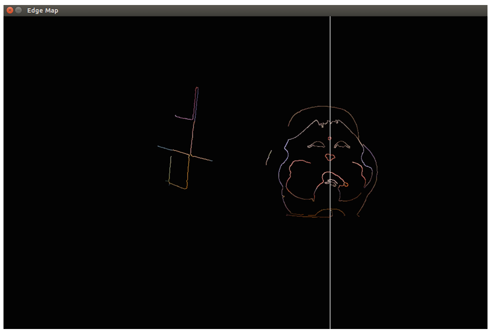

Figure 17: Symmetric line generated for experiment 1

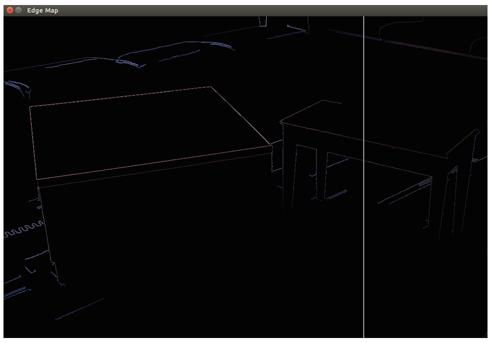

Figure 18: Symmetric line generated for experiment 2

imate is limited (fewer than 3 in experiment 2 and 3). And the occluded part are mostly not sitting on those patches. Therefore, patch approximation cannot achieve much in such scenarios.

4.2.3 Symmetry Approximation Result

We first use canny edge detector to generate the vertical symmetric line. Figure 17 and 18 show the resulting symmetric lines we get for experiment 1 and experiment 2.

With the symmetric line, we are able to calculate the equation of the symmetric plane. The next part is find symmetric point correspondences. As described in Technical Details section, we flip the image horizontally and then use SIFT matching again to find matching between the original image and the flipped image. The resulting symmetric point correspondences we find for experiment 1 is shown in Figure 19.

After we find the symmetric plane S as well as all these symmetric point correspondences, we apply symmetry approximation to each pair of correspondences we find. Figure 20 and 21 show the estimated 3D locations of the SIFT points where we can only see from the full view image for experiment 1 and 2.

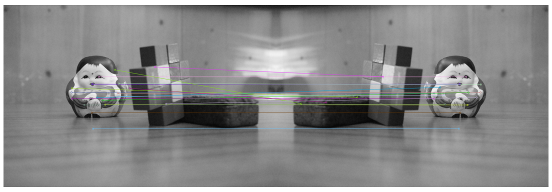

Figure 19: Symmetric point correspondences found by flipping images and SIFT matching for experiment 1

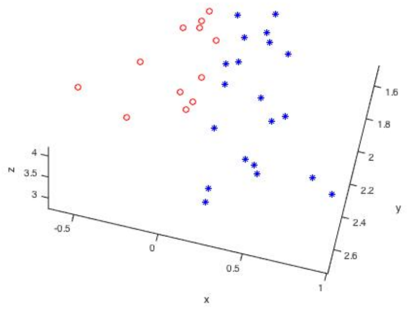

Figure 20: Reconstruction result for experiment 1. Blue points indicate SIFT points where we already know the 3D locations from triangulation; red points are the ones we estimate from symmetry.

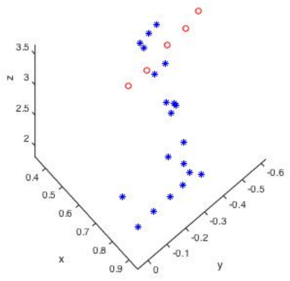

Figure 21: Reconstruction result for experiment 2. Blue points indicate SIFT points where we already know the 3D locations from triangulation; red points are the ones we estimate from symmetry.

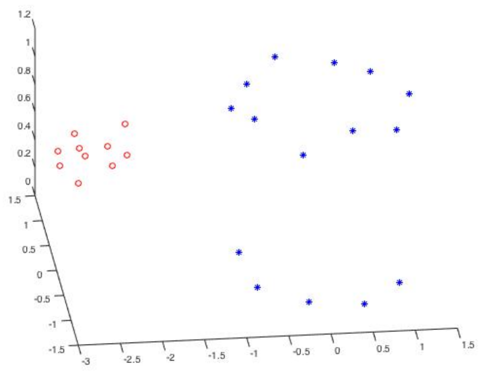

Figure 22: Reconstruction result for experiment 3. Blue points indicate SIFT points where we already know the 3D locations from triangulation; red points are the handle part which we estimate using ratio approximation.

4.2.4 Ratio Approximation Result

Since ratio approximation is essentially an extension of symmetry approximation, we expect similar results to be achieved for experiment 1 and 2. Thus, our main focus here is the result for experiment 3. We pick 10 SIFT points on the handle part of the pot from the full view image and try to estimate their 3D locations. Figure 22 is the reconstruction result for the handle of the pot.

4.3 Result Evaluation

As mentioned previously, we also have a 'hidden' image for each experiment so that we can perform triangulation on the points we estimate during experiments in order to achieve a baseline result for their 3D locations.

For patch approximation, we are able to recover all pixels sit in the same region if we can calculate the plane equation for the region. The obstacle is that it is extremely difficult for us to triangulate all these points and compare the results - thus we choose to visually verify the result and based on Figure 15, the reconstruction result is satisfactory for experiment 1. As mentioned before, this approximation only works well for experiment 1 since we are unable to locate many useful patches for experiment 2 and 3.

For symmetry approximation and ratio approximation, we only reconstruct those SIFT points that can just be seen in one view; therefore we are able to use the hidden image to actually perform triangulation and find their 3D locations (subject to projective ambiguity).

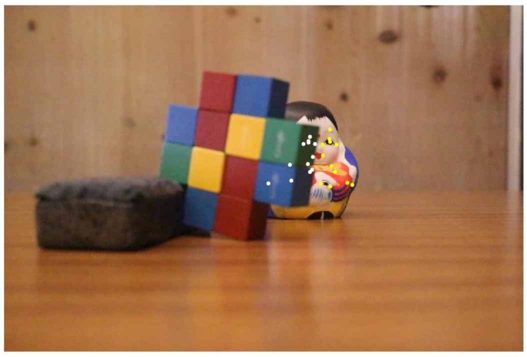

Figure 23: Re-projection result for experiment 1; White points are SIFT points that are re-projected back; yellows points are SIFT points for symmetric correspondences.

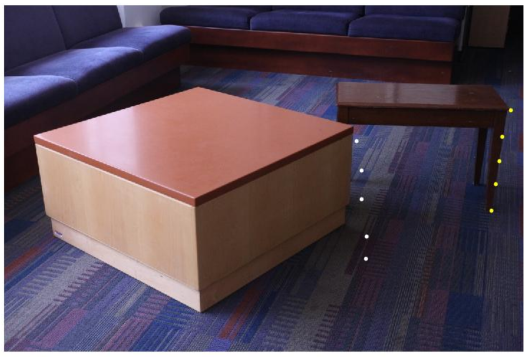

Figure 24: Re-projection result for experiment 2; White points are SIFT points that are re-projected back; yellows points are SIFT points for symmetric correspondences.

In order to compare our estimation with the triangulated locations, we project both our estimated 3D points and the triangulated 3D points back onto another image, where the target object is occluded by another object. In ideal situation, these points should sit in regions which are occupied by the occlusion when projected back onto the image. We can visually verify that to determine if our estimations are good or not. Furthermore, we can compute the average Euclidean distances between the 2D points projected back from our estimations and the ones projected back from triangulation. We use this as our reconstruction error.

Figure 23, 24 and 25 show the re-projection results when we project our estimated points back onto the image with occlusion for experiment 1, 2 and 3. Table 1 shows the reconstruction error for each of the experiment and the heuristic we adopt.

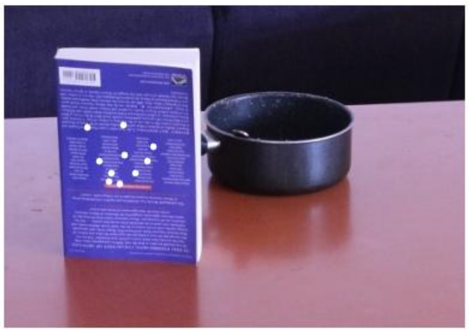

Figure 25: Re-projection result for experiment 3; White points are SIFT points that are re-projected back. There are no yellow points on this graph as we use ratio instead of symmetry approximation.

Experiment	Heuristic	Reconstruction Error
1	Symmetry	1.11
2	Symmetry	32.40
3	Ratio	5.70

Table 1: Reconstruction error for different experiments

We also have estimation error during triangulation. The average triangulation error when using non-linear estimate (refer to CS231A HW2 Question 4 for more details) for each of the 3 experiments is listed in Table 2.

Experiment	Triangulation Error
1	0.97
2	1.50
3	2.00

Table 2: Triangulation error for different experiments

Comparing results in Table 1 and 2, we observe that the reconstruction error is close to the triangulation error when we use symmetry approximation on the Chinese doll. This also gets visually verified by Figure 23, where the white points sit roughly at the expected locations.

Meanwhile, the reconstruction error on chairs when using symmetry approximation is about 20 times as large as the triangulation error. We can verify this by observing that most white points in Figure 24 are located in ground, implying that many the 3D estimates are deviated significantly away from the actual 3D locations. A potential explanation for this is the dark light condition when conducting experiments which causes trouble for finding symmetric correspondences using SIFT. This shows that symmetry approximation is very sensitive to the environment light as it makes heavy use of SIFT.

The reconstruction performance for experiment 3 is somewhat in between the performance for experiment 1 and the performance for experiment 2. From Figure 25 we see there are a few points that are off the expected locations; but most of the white points are still around the handle area (which is covered by the book).

5 Conclusion

We show that using patch, symmetry and ratio approximation, we are able to reconstruct certain occluded parts of the target object given that occluded part is only seen in one view. Different approximations do suffer from different drawbacks - patch approximation requires a good amount of patches to be approximated; symmetry approximation requires the target object to be nearly symmetric; and lastly both symmetry and ratio approximation make the assumption that the camera is close to affine. Therefore, a potential future work for our project is to combine different heuristics together and compare the overall reconstruction result with the one we achieve when using individual approximation.

Our code can be found at `https://www.dropbox.com/sh/3a67yvxpe0a9osn/AAB3N4jJ5Hpvq0E8LHNZpMTea?dl=0`.

References

Kazhdan, M., Bolitho, M., Hoppe, H. *Poisson surface reconstruction* Symposium on Geometry Processing. Eurographics Association (2006)

Yezzi, A.J., Soatto, S. *Structure from motion for scenes without features* Proc. CVPR (2003)

Furukawa, Y., Ponce, J. *Accurate camera calibration from multi-view stereo and bundle adjustment* Proc. CVPR (2008)

Kazhdan, M., Bolitho, M., Hoppe, H. *Poisson surface reconstruction* Symposium on Geometry Processing. Eurographics Association (2006)

Sean Bell, Paul Upchurch, Noah Snavely, Kavita Bala *Material Recognition in the Wild with the Materials in Context Database*