Deformable Object Modeling with Multi-View Imaging

Kevin James Matzen
Department of Computer Science
Cornell University
Ithaca, NY 14853
kmatzen@cs.cornell.edu

Abstract

Household robots operate in unstructured environments and may encounter unknown objects. However, in order to manipulate an object safely, the robot must have some sort of internal representation of the object. For many cases, a rigid model may be assumed, but it is often the case that the object in question is best represented by a deformable model. This paper presents an algorithm for constructing a piecewise-rigid decomposition of an object given a series of images from an inexpensive color and depth camera. We demonstrate this method on a series of household objects and evaluate the performance of our algorithm at various stages in the model construction pipeline.

1 Introduction

Constructing a physical model of an object given only visual observation and no direct interaction is a difficult problem. A household robot must operate in an unstructured environment and in order to safely interact with novel objects, it must understand their physical structure. Therefore, it is interesting and useful to consider the problem where a human leaves an object in various poses over a period of time and a robot observes each of these poses in order to build a physical model.

Given several observations of an object over a period of time where the position and configuration of the object has changed, we desired to produce a piecewise-rigid decomposition of the object. This decomposition will aid planning tasks where a robot must safely pick up an object and manipulate it into some other state. Some examples include, picking up and closing books and laptops or folding clothing.

In order to accomplish this task, we have designed a system that we break down into components. The input is a series of images and depth maps from a color and depth camera. After rectifying these images and depth maps, we find salient features on the surface of the object and perform a 3D reconstruction. This process is invariant under scaling and therefore, we present a modification to bundle adjustment where we take depth data into consideration. Once we have multiple observations of the same object under different poses, we use a Markov Random Field-based inference algorithm to determine how they relate to each other. A hierarchical inference approach is designed to improve speed of the algorithm. Finally, a spectral clustering method is used to segment the data into regions that undergo similar rigid motion.

2 Previous Work

Recently, with the advent of low cost RGB-Depth cameras, merging depth data into existing vision algorithms has become a popular trend in robotics. [11] uses RGB-D data to drive articulated ICP

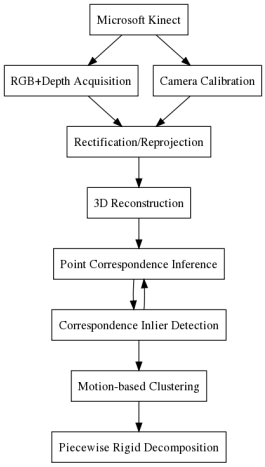

Figure 1: The flow of data for the various stages in our algorithm.

to create a rigid 3D model of an object. However, their implementation requires the manipulator to grasp and rotate the object, a potentially unsafe task given no prior physical model of the object. [19] predicts the grasping point of objects via offline training and explicitly does not build a model of the object. Furthermore, their training set is based on a series of synthetic images. Such a data set requires a human operator to define all geometric primitives.

In order to construct a sparse, feature-based 3D reconstruction of an object, one step that is almost always performed at the end is bundle adjustment. [21] and [12] provide reliable initialization and optimization, respectively, for bundle adjustment. These pieces of software provide the basis for our depth-augmented bundle adjustment stage.

There are several previous attempts to find point correspondences between two models. Mostly, these works exist in the graphics community where there is usually some template model that needs to be animated. Rather than manually posing the template model for each animation, animators can instead take a partial scan of an articulated object and deform the template to match [22], [3], [5], [18]. Such an process would be useful for filming a human actor and fitting a 3D cartoon character to the actor's pose. These formulations either assume a small number of point correspondences or they require dense and accurate data from a laser range scanner. However, [5] provides a key idea that geodesic distance is a property to preserve in our deformation model and that inference on a Markov Random Field using 1 and 2-cliques is a reasonable method. [4] extends [5] by providing details on how to take a piecewise-rigid decomposition and apply articulation constraints to derive a model that can be used for planning. In order to accelerate our desired algorithm, we look to [23] for insight on multilevel inference algorithms. Finally, we look to [10] for insight on how to prune outliers from our correspondence algorithm's output.

Motion segmentation has been explored in many ways, but for the purpose of this research, we focus on clustering methods. Some techniques focus on motion segmentation simply based on frames of video by formulating a shape interaction matrix where points either interact with each other when they are on the same body or they do not interact when they are on separate bodies. [17] shows how the shape interaction matrix can be derived using QR decomposition. 3D methods also exist. [24] shows that pixel intensity can be related to motion, but assumes that video frames are taken with a small interval between each which is not a good assumption when a large interval of time has passed.

3 Depth-Augmented Bundle Adjustment

Our sparse 3D reconstruction uses a standard, feature-based framework. This framework assumes a pinhole camera model and that all of the images have been undistorted.

1. For all images, detect interest points and compute descriptors.
2. For all pairs of images, match descriptors.
3. For all pairs of images (i, j), compute a Fundamental Matrix, F_{ij}. Use a robust technique such as RANSAC to discard outlier matches. The Essential Matrix, $E_{ij} = K^T F_{ij} K$, is obtained where K is our camera intrinsics matrix.
4. Coalesce matches between images into global feature tracks.
5. Add a 3D point to the scene for each independent feature track.
6. Add images one-by-one with most certain images first. Initialize the camera position and orientation based on a factorization of E_{ij} into its rotation and translation components.
7. Apply bundle adjustment to minimize global projection error.
8. Repeat by adding more images and applying bundle adjustment until all images have either been added to the scene or discarded.

The output of this process is a 3D point cloud where each point is tagged with a descriptor.

For those unfamiliar with the field of computer vision, a few of these terms warrant clarification.
Essential Matrix Given 3D points, $y_i, i \in 1, 2, ...n$ that appear in two images at normalized image coordinates x_{1i} and x_{2i}, $x_{2i} E x_{1i} = 0, i \in 1, 2, ...n$ where E is a 3×3 matrix of rank 2. Given a point's projection in one image, the essential matrix provides a line in the second image along which the point's projection must lie called the epipolar line. E may be factored into the rotation and translation between the two cameras. However, since E is rank deficient and invariant under scaling, the translation component is invariant under scaling. It is left as an exercise why the epipolar lines are invariant under translation scaling. Similarly, the final reconstruction has a degree of gauge freedom with respect to scaling. It is also left as an exercise why there are three degrees of gauge freedom with respect to scene rotation and three degrees with respect to scene translation.
Fundamental Matrix and Intrinsic camera parameters In most cases, images will not be presented in normalized image coordinates. However, the camera calibration parameters may be used to transform camera coordinates to normalized image coordinates. $E = K^T F K$.
RANSAC RANdom SAmple Consensus is a generic, robust model fitting technique that iteratively samples the minimum number of data points to fit a model and then tests that model against the entire data set. The model with the highest inlier count is assumed to be a good model for the data. Typically, a final model is fit using something like a least squares technique on just the inliers for the best model. In this instance, the problem is formulated as finding the fundamental matrix between each pair of images.

The end result of this algorithm is a scale invariant scene of 3D points. However, as we will describe in the next section, our algorithm can properly handle arbitrary scene rotation and translation, but it will not be able to properly match two deformable models that have different scales. Therefore, we propose a new algorithm that acts as a modification to the typical formulation of bundle adjustment, one that takes scale into consideration.

Let x_{ij} be the projection of the jth point into the ith image. Let a_i be the rotation and translation of the ith camera and let b_j be the position of the jth point. Let $Q(a, b)$ be a function projecting point b onto image a. Assuming the error model for projection is Gaussian, the minimization of the projection error is a least squares problem taking on the form

$$a^*, b^* = \arg\min_{a,b} \sum_{i,j} ||Q(a_i, b_j) - x_{ij}||^2 \tag{1}$$

In our formulation, we assume depth is also available for each point. We modify the cost function by including an addition depth projection error term. Let d_{ij} be the depth of point j on image i and a_{i_t} be the translation component of camera i. Our new objection function is

$$a^*, b^* = \arg\min_{a,b} \sum_{i,j} \left[||Q(a_i, b_j) - x_{ij}||^2 + \alpha |||b_j - a_{i_t}|| - d_{ij}||^2 \right] \tag{2}$$

where α is a tuning parameter that informs the algorithm how much we care about depth with respect to feature detection and matching.

Note that this formulation is not linear. In fact, the bundle adjustment problem in general is highly non-linear due to the rotation component of each camera. A non-linear optimization technique is employed. Levenberg-Marquardt is a general, non-linear optimization algorithm that computes the Jacobian at a particular point and linearizes the function. It then operates in a fashion similar to Gauss-Newton or gradient descent depending on a dampening factor that is computed at each iteration. If the reduction of the objective function is large at a certain iteration, the dampening factor should be small to avoid overshooting and large otherwise.

The objective function at hand is not only non-linear, but has many local minima. Therefore, initialization must be done very carefully. Typical techniques, like described above, try to add images to the scene one by one, first choosing images that are most likely to fit the other images already in the set. This greedy approach produces good results in general.

Since bundle adjustment is typically performed with a non-linear optimization technique, depth is a reasonable variable to consider given that we can take existing bundle adjustment software and make simple modifications to include this cost term. Furthermore, an analytical derivation of the Jacobian is not required since a finite differences-based Jacobian works just as well in practice.

4 Hierarchical Correspondence Inference

The primary assumption that we make is that the deformations are approximately isometric. That is, given two points on the surface of the object, geodesic distance is preserved under the deformation. This is a property that we believe many deformable household objects possess.

We formulate the correspondence problem as a Markov Random Field composed of 1-cliques and 2-cliques. Let one point cloud be the source and let the other point cloud be the target. The nodes in our MRF are the points in the source and we model them with a multinomial distribution where each state is a point in the target that will form a correspondence. Let S_{si} be the descriptor for point i in the source and S_{ti} likewise for the target. The 1-clique potential for each node is formulated as

$$\phi_i(x_j) = exp(-\gamma||S_{si} - S_{tj}||^2) \tag{3}$$

This encodes that we favor correspondences where the feature descriptors of the two points are similar. Let $GD_s(i,j)$ be the geodesic distance between points i an j in the source and likewise for the target. The 2-clique potentials are formulated as

$$\phi_{ij}(x_k, x_l) = \begin{cases} 1 - \frac{1}{\bar{d}^2}\frac{GD_s(i,j)-GD_t(k,l)}{GD_s(i,j)GD_t(k,l)}^2 & \left| \frac{GD_s(i,j)-GD_t(k,l)}{GD_s(i,j)GD_t(k,l)} < \bar{d} \right. \\ 0 & otherwise \end{cases} \tag{4}$$

The goal behind the quadratic 2-clique model was to sparsify the MRF. An inverse exponential would have a gradual decay leaving several 2-clique potentials with near-zero potential. This leads to a situation where it is not easy to determine the cutoff where we can disregard the edges and since this MRF is highly connected, potentially complete. We must sparsify it in some sense if we expect to perform inference on it. We have exact methods for inference on graphs with loops such as the junction tree algorithm, but large cliques are not desired. [5] uses a threshold model and has found success, but this was done on very small models. Therefore, the quadratic model seems to provide a good balance between simplicity and a strict cutoff.

It is not applicable in this problem to apply convex optimization techniques since the problem is discrete rather than continuous. Furthermore, the state space is so large that a continuous relaxation of the discrete problem is not feasible. This is why we look towards graph-based inference techniques. In practice, loopy belief propagation failed to converge on these graphs. Since there are no general guarantees with respect to loopy belief propagation when it fails to converge, junction tree was explored. However, due to the size and connectivity of these graphs, junction tree required far too much memory to operate on the full graph. In order to resolve this problem, we propose a hierarchical correspondence inference algorithm.

Rather than try to explicitly match a source point to a target point, we first try to assign a region of source points to a region of target points and then recursively refine them. This coarse-to-fine approach is similar to multiresolution or multigrid approaches except the boundaries between regions at a particular level are not entirely obvious. What we require is a way to determine how to find these regions.

One way to consider the problem is to think of each node in the MRF has having an importance associated with it. We desire to sample equally spaced regions in order to get a reasonable, coarse representation of the MRF. If we were to look at the messages being passed around in belief propagation, it gives some insight that important nodes are nodes that given a random walk on the graph will be hit the most often. The problem then becomes finding the stationary distribution of a random walk on this graph. To find this stationary distribution, a Markov Chain Monte Carlo simulation may be performed. Transition probabilities are defined based on the distance between k-nearest neighbors. Finally, a small transition probability between all pairs of points is added to ensure ergodicity. The second largest eigenvalue of the transition probability matrix, P, is computed to determine the mixing time of the Markov chain where [8]

$$\mu(P) = max\{\lambda_2, -\lambda_n\} \tag{5}$$

$$\text{Mixing rate} = log(1/\mu(P)) \tag{6}$$

$$\text{Mixing time} = \tau = \frac{1}{log(1/\mu(P))} \tag{7}$$

After τ steps, the L_1 norm of the stationary distribution minus the sampled distribution will be less than $exp(-1)$. Therefore, we have a rough idea of how many iterations to run the MCMC and get a good idea of which nodes are the most important.

After we sample these important nodes, we cluster all other nodes based on nearest neighbor geodesic distance. It works well to simply perform a weighted average of the geodesic distances between all points in a cluster since they were all clustered based on geodesic distance in the first place. However, performing a weighted average on the feature descriptors isn't very robust. A local region of an object does not necessarily have similar features and averaging them loses this detail. Therefore, the 1-clique potential is reworked to be the number of matches between the two clusters that have some L_2 norm within a designated threshold normalized by the number of points in the cluster.

This hierarchical, top-down approach is good for getting initial estimates for the positions of the correspondences, but it does lose global information at each level. For example, if at a particular level, two clusters are formed, the refinement in each cluster will happen independently of the other cluster. To resolve this issue, we take a further iterative approach. Give a single top to bottom run of the inference algorithm, we change the 1-clique probabilities to emphasize the MAP estimate that we have found so far rather than the feature descriptor. There are some nodes where we have yet to make a decision with high certainty. We run the algorithm again to determine how they should be assigned. The algorithm terminates once we reach an iteration that makes no further assignments. The algorithm is guaranteed to terminate since each iteration either increases the number of assignments in which case we will eventually reach all nodes being assigned or they remain the same in which case we terminate.

5 Correspondence Outlier Rejection

For the sake of completeness, we include a formulation of an outlier rejection algorithm that we designed. In many cases, a model for the data is simple and a threshold-based rejection algorithm will suffice. However, the correspondence problem of deformable models does not have a simple model and therefore more flexible techniques must be explored. We present the algorithm and its formulation, however, after implementation, it was found that a faster solver must be found before it could be included in the experiments that we present.

We formulate the outlier rejection problem as follows. Given a set of elements S and a function $D : S \times S \to \mathcal{R}$ which represents the disagreement between two elements of S, how can we choose elements of S such that we have a large number of elements, but a low total level of disagreement between them. If we express this as a graph where the vertices are the elements of the set and the weighted edges are the disagreements between all pairs of elements, then this optimization problem may be expressed as

$$\begin{array}{rl} \max & \alpha \sum_{v \in V} x_v - \sum_{e \in E} W_e \min\{x_{e_u}, x_{e_v}\} \\ \text{s.t.} & x_v = \{0, 1\}, \forall v \in V \end{array} \tag{8}$$

Clearly, this encodes the problem exactly. For every vertex we select in the graph, we count and weight according to the tuning parameter. For all edges, if both of its vertices are selected, then we count the level of disagreement between them. Solving this problem can be done in worst-case exponential time and we don't attempt to place a tighter bound on the running time. Instead, we consider the fractional relaxation of this problem.

$$\begin{array}{rl} \max & \alpha \sum_{v \in V} x_v - \sum_{e \in E} W_e \min\{x_{e_u}, x_{e_v}\} \\ \text{s.t.} & x_v \leq 1, \forall v \in V \\ & x_v \geq 0, \forall v \in V \end{array} \tag{9}$$

Finally, by adding some slack variables, we can turn this relaxation into a linear program relaxation of the original problem.

$$\begin{array}{rl} \max & \alpha \sum_{v \in V} x_v - \sum_{e \in E} W_e x_e \\ \text{s.t.} & x_e \leq x_{e_u}, \forall e \in E \\ & x_e \leq x_{e_v}, \forall e \in E \\ & x_v \leq 1, \forall v \in V \\ & x_e \geq 0, \forall e \in E \end{array} \tag{10}$$

The solution of a linear program may be found in polynomial time, therefore, a polynomial time algorithm exists to solve our outlier rejection relaxation problem. As long as a model for the disagreement between elements in the model exists, such as difference in geodesic distance for correspondences in our problem, this algorithm will find the desired level of balance. The fractional solution may be quantized and used directly, or the value of the objective function may be used in reducing the search space for a branch and bound algorithm over the integer problem's solution space.

6 Motion-Based Clustering

After we have two point clouds representing an object in two different poses along with the correspondences between the two clouds, we desire to cluster sets of points into regions that undergo similar rigid motion. We propose a formulation of normalized cuts that encodes this desired clustering behavior.

Let W be the dissimilarity where W_{ij} encodes how dissimilar two points are with respect to their motions and let S be the similarity matrix with where S_{ij} encodes how similar two points are with respect to their motions. Our previous assumption for this problem was that geodesic distance is preserved under deformation. However, what does change is Euclidean distance. Therefore, we encode two points to be similar if they have a low difference in Euclidean distance.

$$W_{ij} = ||ED_s(i,j) - ED_t(i',j')|| \tag{11}$$

$$S_{ij} = exp\left(-\gamma W_{ij}^2\right) \tag{12}$$

This formulation has the parameter γ that requires tuning. This was found experimentally, but results indicate that some automatic adjustment is required on a per-instance basis for it to be reliable.

Additionally, we implement an existing motion segmentation affinity matrix called the shape interaction matrix [7]. Let P be a $3F \times N$ matrix where N 3D trajectories over F point clouds are stacked into columns. We find an orthonormal basis of $\mathcal{R}^N$ using the SVD of P. Taking the sum of the outer products of the first rank(P) rows of this orthonormal basis gives us Q, the shape interaction matrix of P. Elements of Q capture the angle between sub-bases and therefore, a 1 indicates that two elements share the same basis and a 0 indicates they do not. Since this is a basis for trajectory, a 1 indicates two points are on the same object and 0 indicates they are on separate objects. The problem becomes permuting the matrix into block diagonal form which can be approximated with normalized cuts.

Normalized cuts is an NP-Complete problem [20], so we use spectral methods to find a good approximation. Let D be a diagonal matrix such that

$$D_{ii} = \sum_{j=1}^{n} S_{ij} \tag{13}$$

Then the normalized Laplacian matrix is

$$L = D^{-\frac{1}{2}} S D^{-\frac{1}{2}} \tag{14}$$

We compute the eigenvalues and eigenvectors of L. Let k equal the eigengap of L. We take the k dominant eigenvectors and apply k-means clustering. This produces our final clustering and our piecewise-rigid decomposition. In practice, both formulations did not produce a correct eigengap in presence of noise. Therefore, our experiments fix it to 2.

7 Experiments

7.1 Data Acquisition

The input to this algorithm is a series of video frames acquired from a color and depth camera. For the purposes of this experiment, the camera used is a Microsoft Kinect, a low cost color and depth camera sold for the home entertainment industry. Usage of the Microsoft Kinect is not officially supported for use other than with the Microsoft Xbox 360, but the open source community has reverse engineered a significant portion of the USB protocol [2]. It is assumed that data acquired from this unit is accurate. Manual verification of all data was performed to make sure that no corrupt data was captured.

Members of the open source Kinect community have put together a series of tools in addition to the driver itself. One such tool that we used was a color and depth camera calibration tool [6]. Color camera calibration works as normal. However, it is more difficult to get an accurate calibration of the depth camera. The reason being, it's not as simple to just image a standard chessboard calibration pattern. Instead, we used a cardboard cutout with depth discontinuities and manually annotated coordinates. This provided enough data to accurately find the intrinsic calibration parameters for the depth camera as well as the extrinsic calibration parameters of the two cameras. It should also be noted that while the Kinect serves as a low cost platform, the cameras are low quality themselves and this calibration process must be done on a per unit basis since the parameters vary greatly from unit to unit. The depth map to meter units calibration was performed by placing objects at specific intervals from the camera. The regression model was found to be linear with respect to inverse distance.

In order to fuse the depth and color data into a single view, we had to re-project the data from the depth image onto the color image. This was achieved via a utility that we wrote using OpenGL and will be made available as an open source, general purpose RGB+Depth re-projection utility. Using OpenGL, we simulate a pinhole camera model and in the frame of the depth camera, we unproject the data into the scene. Then we move the camera over to the frame of the color camera and record the depth at each pixel. The performance of this implementation is fast enough so that this re-projection may happen without limiting the framerate of the Kinect assuming a modern GPU is present in the system. Furthermore, it is general enough that the camera calibration parameters may be specified independent of the executable.

A series of deformable household objects such as toys, books, boxes, etc. where gathered. The set of objects was restricted to objects with patterns and textures that could lead to a correct 3D reconstruction. Therefore, objects of a single color without any patterns or sharp corners were not included. In addition, transparent objects where an accurate depth can not be found with our depth camera were excluded. We believe that this does not limit the application of our algorithm too much since many transparent objects in a home, such as dishes, are rigid. Each one was placed on a table in a particular pose and imaged from multiple views. The data set includes 9 objects, each with at least 2 poses, for a total of 49 poses. Some objects have more ways in which they could deform and so the number of poses for a particular object reflects the number of ways we could think of deforming the object. It was found that our hierarchical correspondence inference formulation improved results, but not enough that we could segment the non-articulated objects well enough to make a reasonable performance analysis on them. Therefore, we limited our results to three of the nine objects that performed well enough to be reasonable and display them here.

7.2 Depth-Augmented Bundle Adjustment

Feature extraction and matching was done using SIFT features [13]. In order to implement our depth-augmented bundle adjustment algorithm, we used an existing bundle adjustment package called Bundler [21]. Bundler performs the necessary initialization for Levenberg-Marquardt to find a good local minimum. Then it delegates the bundle adjustment step to a library called Sparse Bundle Adjustment or sba [12]. Our modification extracts the depth for each feature point from our reprojected depth maps, and modifies the calls to the relevant sba functions. sba considers the problem in terms of the parameter space and the projections along with a function to compute the projection from the parameters. Then it uses Euclidean distance to compute the projection error. Simply reworking the depth-augmented bundle adjustment as projection with three coordinates, x, y, and depth, and then redefining the projection function to set the last coordinate to be the Euclidean distance between the camera and the point gets us the desired functionality. The depth-trust tuning parameter can be tuned via the command line. In order to validate the functionality of our implementation, we take the real object, measure the distance between a representative series of points on the surface of the object and compare the distance to our reconstruction. A laser scanner was not available and so we simply have this qualitative assessment of our algorithm. Nonetheless, we report this qualitative performance compared to a classical BA reconstruction where we manually find the correct scaling factor.

7.3 Hierarchical Correspondence Inference

Our hierarchical correspondence inference algorithm intends to find a valid assignment of all points such that the assignments fit our model of geodesic distance preservation well. However, this is a combinatorial optimization problem and we developed the hierarchical inference algorithm as a fast approximation. In order to validate the formulation of this approximation, we compare the results of this algorithm against an implementation of nearest neighbors that simply matches feature descriptors. Junction tree inference is performed using libDAI [15], an open source discrete approximate inference library. Geodesic distance was computed using FLANN [16] for fast approximate nearest neighbors and MatlabBGL [9] as a wrapper to the Boost Graph Library [1] for all pairs shortest paths.

7.4 Motion-based Clustering

The motion-based clustering task is an unsupervised task. Therefore, we developed a program that allows a user to view a point cloud in 3D and highlight points that they believe should be in the same cluster. For example, if the object is a plush children's toy of an animal and the deformation is that we twist the head around, then we expect the user to label the head as one cluster and the rest of the body as a second cluster. Labels for all instances were done by a single user and then verified by a second. The effectiveness of the algorithm is reported in terms of the Rand index.

8 Results

8.1 Depth-Augmented Bundle Adjustment

Results of the modified bundle adjustment algorithm were qualitatively evaluated against the unmodified bundle adjustment results to make sure the depth optimization provided reasonable correction. Objects were measured and the reconstructions were approximately 1 or 2 centimeters smaller than the real object when the imaged object was 1 meter from the camera. We assume this is either due to our calibration of the camera, due to the inaccuracy of the depth cameras algorithm itself, or due to the sparsity of the final point cloud. For our goal, all we desire is a canonical scale, not that it be extremely accurate.

8.2 Hierarchical Correspondence Inference

We also qualitatively evaluated the performance of our correspondence inference algorithm against naive nearest neighbors SIFT matching. The results were reasonable for two component objects with low texture symmetry, but unreasonable for the rest of our data set.

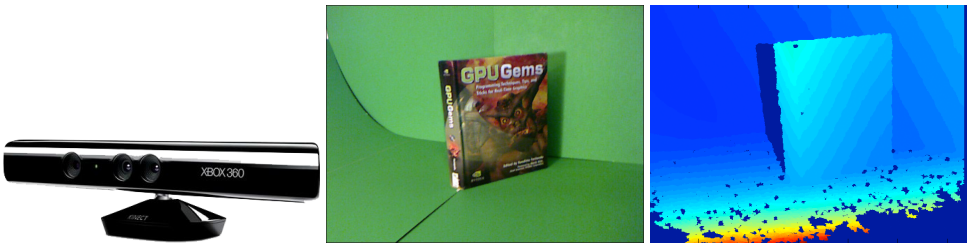

Figure 2: A Microsoft Kinect and an image with its depth map.

Figure 3: A 3D point cloud generated from depth-augmented bundle adjustment. Left: Correct size is 0.22 meters, Right: 0.30 meters.

8.3 Motion-based Clustering

Unfortunately, due to our simplification of the MRF where we try to sparsify the graph, geodesic distance still works well locally, but does not work well globally on objects with symmetry. Furthermore, since we cannot determine the correct number of clusters automatically using the eigen-gap, we simplified the problem to two clusters. We quantitatively evaluated the final performance against a few simple examples of objects such as books that have two primary rigid components by using Rand index based on hand labeled data.

Object	NN + Shape Interaction	MRF + Shape Interaction	MRF + Diff Euclidean Distance
1	0.5684	0.8950	0.9518
2	0.5136	0.5856	0.5055
3	0.5198	0.5584	0.5349

9 Conclusion

Given several color and depth images of a simple object in different poses, we are able to improve the decomposition into two rigid components over a naive nearest neighbors correspondence method. While this type of decomposition has been possible for a few years in the field of computer graphics where an ideal reference model is available and must be articulated to fit a laser scan, our implementation is successful in some attempts to do this with sparse SIFT keypoints and no prior reference model. Unfortunately, the success rate is low.

The availability of low-cost color and depth cameras is significant given that they are now sold for home entertainment. The use of the depth information in this work was used only for the recovery of scale during bundle adjustment due to the until recent unavailability of such a low-cost camera.

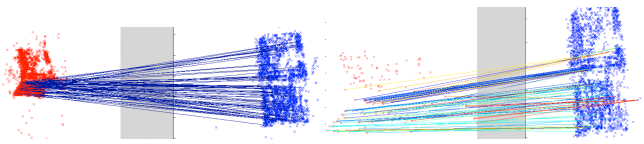

Figure 4: A subset of point correspondences found between two poses. Left: SIFT nearest neighbors. Right: MRF SIFT + geodesic distance. Not rendered to scale.

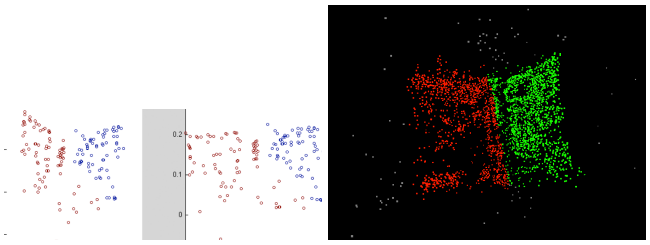

Figure 5: Left: Detected rigid components. Right: Hand-labeled components.

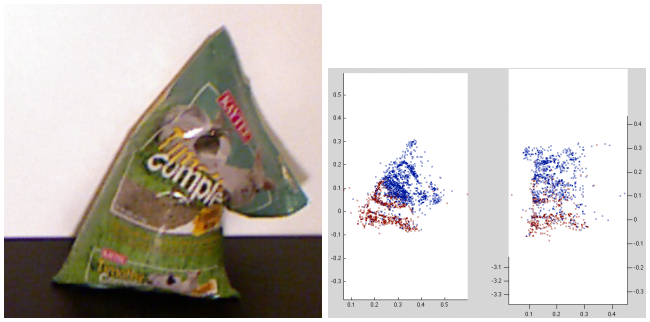

Figure 6: Object 2 from our evaluation. This example violates the assumption that it is piecewise-rigid, but a good two part approximation can be found.

Future work using more depth information includes using dense features such as spin images on a Poisson surface reconstruction of the object.

Additionally, our attempt only captures rigid motion between two poses of an object. Given more observations it is possible for the model to increase in complexity and accuracy. Perhaps a hierarchical decomposition or a decomposition fusion using another MRF may prove useful. A primary difficulty in this problem is performing this decomposition with only two poses. Video of an object makes the task much simpler since the feature points may be more robustly tracked. However, our problem was posed in such a way that a robot would only see the objects after the human has moved them. Observation of a human manipulating the object might be a feasible alternative goal. Finally, rather than trying to learn rigid components through unsupervised learning, it might be more fruitful to use supervised learning for all sub-shapes that describe common rigid components of household items. It would be less general, but more feasible.

References

[1] *Boost Graph Library,* `http://www.boost.org/doc/libs/1_42_0/libs/graph/doc/index.html`.

[2] *libfreenect,*`https://github.com/OpenKinect/libfreenect`.

[3] Brett Allen, Brian Curless, and Zoran Popovic, *Articulated body deformation from range scan data*, 2002.

[4] Dragomir Anguelov, Daphne Koller, Hoi cheung Pang, Praveen Srinivasan, and Sebastian Thrun, *Recovering articulated object models from 3D range data*, In Proc. UAI, AUAI Press, 2004, pp. 18–26.

[5] Dragomir Anguelov, Praveen Srinivasan, Hoi cheung Pang, and Daphne Koller, *The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces*, In TR-SAIL-2004-100, at http://robotics.stanford.edu/drago/cc/tr100.pdf, 2004, pp. 33–40.

[6] Nicolas Burrus, *Kinect Calibration,* `http://nicolas.burrus.name/index.php/Research/KinectRgbDemo`.

[7] J. Costeira and T. Kanade, *A multi-body factorization method for motion analysis*, Computer Vision, IEEE International Conference on **0** (1995), 1071.

[8] Garren, Steven T. and Smith, Richard L., *Estimating the second largest eigenvalue of a markov transition matrix*, Bernoulli **6** (2000), no. 2, 215–242.

[9] David Gleich, *MatlabBGL,* `http://www.stanford.edu/~dgleich/programs/matlab_bgl/`.

[10] Qi-Xing Huang, Bart Adams, Martin Wicke, and Leonidas J. Guibas, *Non-rigid registration under isometric deformations*, Proceedings of the Symposium on Geometry Processing (Aire-la-Ville, Switzerland, Switzerland), SGP '08, Eurographics Association, 2008, pp. 1449–1457.

[11] Michael Krainin, Peter Henry, Xiaofeng Ren, and Dieter Fox, *Manipulator and object tracking for in hand 3D object modeling*, Tech. Report UW-CSE-10-09-01, University of Washington, September 2010.

[12] M.I. A. Lourakis and A.A. Argyros, *SBA: A Software Package for Generic Sparse Bundle Adjustment*, ACM Trans. Math. Software **36** (2009), no. 1, 1–30.

[13] David G. Lowe, *Distinctive image features from scale-invariant keypoints*, 2003.

[14] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze, *Introduction to Information Retrieval*, 1 ed., Cambridge University Press, July 2008.

[15] Joris M. Mooij, *libDAI: A free and open source C++ library for discrete approximate inference in graphical models*, Journal of Machine Learning Research **11** (2010), 2169–2173.

[16] Marius Muja, *FLANN, fast library for approximate nearest neighbors*, 2009, `http://mloss.org/software/view/143/`.

[17] JinHyeong Park, Hongyuan Zha, and Rangachar Kasturi, *Spectral clustering for robust motion segmentation*, Computer Vision - ECCV 2004 (Tomás Pajdla and Jirí Matas, eds.), Lecture Notes in Computer Science, vol. 3024, Springer Berlin / Heidelberg, 2004, pp. 390–401.

[18] Y. Sahillioglu and Y. Yemez, *3D shape correspondence by isometry-driven greedy optimization*, Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010, pp. 453 –458.

[19] Ashutosh Saxena, Justin Driemeyer, and Andrew Y. Ng, *Robotic grasping of novel objects using vision*, Int. J. Rob. Res. **27** (2008), 157–173.

[20] Jianbo Shi and J. Malik, *Normalized cuts and image segmentation*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **22** (2000), no. 8, 888 –905.

[21] Noah Snavely, Steven M. Seitz, and Richard Szeliski, *Photo tourism: Exploring photo collections in 3D*, ACM TRANSACTIONS ON GRAPHICS, Press, 2006, pp. 835–846.

[22] C. Stoll, Z. Karni, C. Rossl, H. Yamauchi, and H.P. Seidel, *Template deformation for point cloud fitting*, Symposium on Point-Based Graphics, 2006, pp. 27–35.

[23] Liang Xiong, Fei Wang, and Changshui Zhang, *Multilevel belief propagation for fast inference on markov random fields*, Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on, 2007, pp. 371 –380.

[24] Hua Yang, Greg Welch, Jan michael Frahm, Marc Pollefeys, and Kitware Inc, *3D motion segmentation using intensity trajectory*.