3D Model Reconstruction

Kent Sommer
somme282
4530009

I. Introduction

For my portion of the project, I was in charge of recovering 3D shape of some object from a given set of images and projection matrices. The issue of recovering shape from multiple images can be tackled in many different ways with some ways working better than others for specific scenarios. Recently, there has been a surge of demand for 3D content in applications such as virtual reality, computer graphics, and video games. With products like the Oculus Rift and SteamVR headset coming onto the market soon, there will be a need for even more 3D content to be put into the virtual environments. Depending on the complexity of the object it can be beneficial to create a model from a real world object rather than creating the 3D model using a CAD tool. All of these applications help make this an interesting, and important problem to research.

II. Background and related work

Due to my lack of background in computer vision, I was unsure how best to approach the problem presented. I approached the problem thinking I could end up with an algorithm capable of running on a mobile device and allowing for real-time shape reconstruction. I very quickly realized that in order to finish the project in the time allotted, I would need to constrain my problem a bit. Although there are methods and applications available today that do accomplish the mobility and speed I had initially hoped for, my naive knowledge in computer vision and limited time frame made it necessary to wait until a later time to attempt to solve the problem in the way I had initially hoped. Related work necessary to solve the problem presented also includes silhouette extraction/foreground extraction (to determine what object is the one being modeled in an image), and camera calibration. After the model/shape is reconstructed, it would also be good to ”skin”/”texture” the model so that it looks like the object that was photographed.

III. Relevant Approaches

There are multiple different ways to approach the problem of shape reconstruction, however, I focused on silhouette based 3D shape reconstruction. Another approach that is discussed in one of my papers but I ended up not focusing on, is reconstruction using photo-consistency. Generally, this approach attempts to constrain valid points to those that appear with the same color over all the images where the point is visible. Knowing what I do know about the limitations of shape from silhouette (namely that it cannot recover concave structure) I think using a photo-consistency approach might have produced better results. The three papers I focused on were: ”Surface Model Reconstruction of 3D Objects From Multiple Views,” ”A Theory of Shape by Space Carving,” and ”Silhouette-Based 3-D model Reconstruction From Multiple Images.” All three papers use some form of space carving to solve the problem of shape reconstruction. Also, with the exception of ”A Theory of Shape by Space Carving” all the papers make use of silhouettes to perform this carving. Although there are other methods besides space carving to determine shape (stereo pairs and triangulation for instance), I did not focus on these methods in my research and as such will not be discussing them.

IV. Surface Model Reconstruction of 3D Objects From Multiple Views - First Paper

A. Problem formulation

The problem posed in this paper is the reconstruction of an object from multiple silhouettes using a robot with a camera mounted in an eye-in-hand configuration. The camera and robotic system are calibrated and the circular path of the system must be known before image acquisition. Because the path must be known, there is a limitation in what objects can be used at random (if the size of the object does not allow for the path to remain the same then the path must be adjusted to fit the new object).

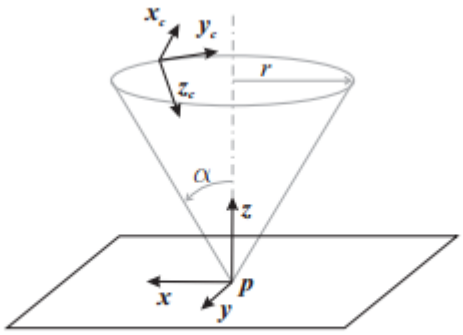


Fig. 1. Circular trajectory executed by the camera mounted on the robot hand during the image acquisition step with respect to the reference frame.

B. Preliminaries

Voxel: A value on a grid in 3D space - a mix of ”volume” and ”pixel,” and for this paper can be thought of as pixel represented in some ”space.”

Silhouette: Binary image with foreground/object pixels having a value of 1 and all other pixels (background) having a value of 0.

C. Approach

First, the circular camera trajectory described above in figure 1 is generated and assigns a rough estimation of the location of the object being modeled p, the number of images to take N, the revolution axis z, the observation angle α, and path radius r. Once the collection of images has been completed, the silhouettes are then extracted using blob analysis and image enhancement in both the frequency and spatial domains.

Then, for each of the N images, the centroid of the blob is evaluated giving the radius r_i of the bounding circle that the silhouette must be inside of. This allows for calculation of the radius of the bounding sphere for the object where ε is the enlargement coefficient and i runs from 1 to N.

$$r_s = max\ r_i + \epsilon$$

The bounding sphere is then reduced using dynamic steps depending on distance from the object. Once a point on the sphere intersects the visual hull it is marked as fixed and it is removed from further processing. It is important to note that this means that only the points representing the surface will be processed, there are no points ”inside the object” unlike many other common carving techniques.

They then use an interesting technique to reduce noise which works by finding a straight line z_{pi} which is a projection of z onto the ith image plane. The maximum length segment parallel to this line z_{pi} is then found.

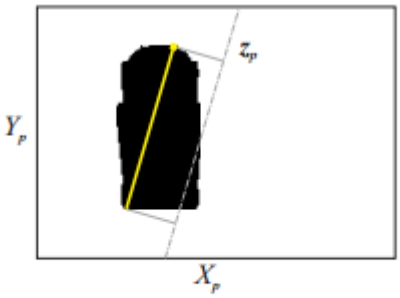


Fig. 4. The dot-dashed line represents the back-projection $\mathbf{z}_p$ of the axis $\mathbf{z}$ on an image plane; the yellow line represents the maximum chord of the silhouette parallel to $\mathbf{z}_p$.

Then the maximum segment is sampled and projected into the 3D space outside the estimated current volume and on the

opposite side of the centroid in respect to the current view. The points are then pushed towards the centroid with step η and at each step, every point is projected back onto **all** the image planes. Once a point on this segment is inside all silhouettes, it is set as fixed just like above and is removed from further processing. Once all the points have been fixed or if there are no points that belong, processing is stopped and the point s_{ij} that corresponds to the largest value of z in the positive direction is saved. This processes is repeated for all camera views and a list of maximum points is created. All of these points together represent the highest point of an object that is visible in each view. The point M can then be calculated as follows to ensure that part of the object does not get carved off due to noise (where δ_M is a weight coefficient related to step size):

$$M = min\ M_s + \delta_M$$

D. Critique

This method despite being somewhat more complicated than the methods I will discuss in the next two papers does have the benefit of being faster in theory. Because they decided not to create a ”solid” point cloud but rather only a point cloud representing the surface, they saved lots of processing on points that wouldn't need to be processed as they were inside the object model. This also meant they had to shrink a bounding sphere around the object and essentially fit the sphere to the object instead of carving away points (not possible when you don't start with a solid bounding set of points). With such a proportionally smaller data-set they managed to produce a real time reconstruction with the only ”slow” portion being image acquisition. Despite the upsides, this method ultimately cannot produce a ”perfect reconstruction” since the amount 3D understanding one can draw from silhouettes is somewhat limited (particularly in regards to concavities). They do assume a ”good number” is chosen for the density of the bounding sphere, however, without trying multiple densities it would be almost impossible to select the density that yielded the ”optimal” reconstruction without using more points than needed.

V. A Theory of Shape by Space Carving - Second Paper

A. Problem formulation

The problem posed in this paper is the reconstruction of an unknown, arbitrarily-shaped scene using multiple images taken at known but randomly distributed viewpoints. The images are acquired in such a way that the background of each image is known so that background subtraction yields a very clean foreground object (low noise throughout the image).

B. Preliminaries

Photo hull: Let υ be an arbitrary subset of R^3. If υ^* is the union of all photo-consistent shapes in υ, every point on the surface of υ^* is photo-consistent. υ^* is the photo hull.

Photo-consistency: A voxel is considered to be photo consistent when its color is similar in all camera views that can see it.

C. Approach

The approach taken in this paper relies on a multi-view visibility ordering (visit occluders before the voxels that they occlude). Although there is no multi-view visibility ordering in a general case, they defined it as visiting voxels in order of increasing X coordinate, and for each voxel p = (X_p, Y_p, Z_p) consider only views which have an X coordinate less than X_p. Thus, if p occludes q from a camera at some location L, then p is on the line segment Lq and thus $X_p < X_q$ and p gets evaluated before q. Then, the algorithm to obtain

the model through space carving can be done using a multi-sweep volumetric algorithm. A solid block of voxels is carved away by sweeping over a single plane along pre-defined sweep directions. For each position of the plane, voxels are evaluated using photo-consistency checks. To make sure that all voxels are consistent with the entire set of input images it is necessary to perform multiple plane sweeps as described above. Once this has been done, every voxel that has had its consistency evaluated in multiple plane sweeps is checked for consistency across all camera views that participated in its consistency check across the multiple plane sweeps. This approach ensures that a voxel is consistent with all views that can see it, not just the subset of camera views on the side of a given plane sweep.

D. Critique

This approach solves a lot of the issues inherent with the approach from the first paper above, but does so at a large sacrifice to speed. For instance, on a set of 16 images of a gargoyle sculpture, this implementation took 250 minutes to compute the model. Granted the speed of the workstation at the time was not great, however, the number of voxels processed was close to 51 million which today would still take a large chunk of time on most systems. This approach does solve the issue of concavities though, and is the only way (currently known) to accurately reconstruct non-smooth, free-form shapes from camera views that are randomly positioned and/or oriented. They did make the assumption that during image capture, the background (without the object) would be captured first to allow for perfect background subtraction which would significantly reduce noise. In a real world setup this is possible, however, it limits the feasibility of doing automated image acquisition. It also does not allow for the reconstruction of very large objects (impossible to take an image of the Colosseum background without the Colosseum being in the image). In general, this approach works well if speed is not an issue and the object being modeled can be moved in/out of an image frame (to allow for an image of just the background).

VI. Silhouette-Based 3-D Model Reconstruction From Multiple Images - Third Paper

A. Problem formulation

The problem posed in this paper is the reconstruction of a complex 3D object from a set of 2D images. The setup requires a pre-calibrated fixed camera. The multiple images are taken by rotating the object about its center on a rotary/turn table. Lighting and background are also controlled/known before hand so as to reduce noise and allow for easier foreground and background segmentation. Each new image angle (turn of the table between each image capture) also must be known before-hand so that the projection matrices for each view are available to the algorithm (internal and external parameters are extracted from the projection matrix P = K[R — T] where K contains the internal parameters of the camera and R and T represent the external rotation and translation parameters respectively).

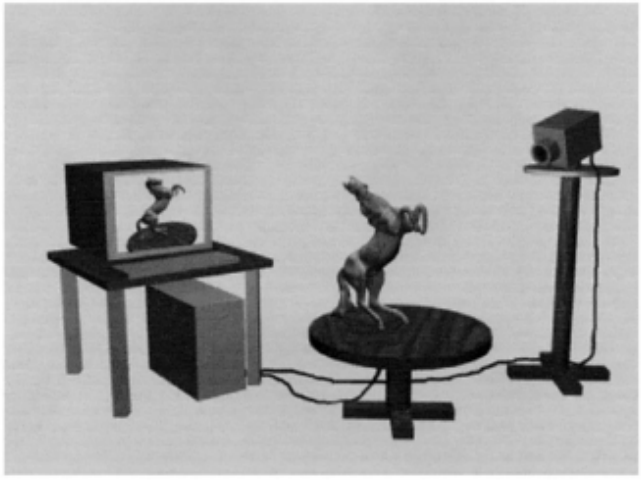


Fig. 1. System setup for image acquisition.

B. Preliminaries

Voxel: A value on a grid in 3D space - a mix of ”volume” and ”pixel,” and for this paper can be thought of as pixel represented in some ”space.”

Silhouette: Binary image with foreground/object pixels having a value of 1 and all other pixels (background) having a value of 0.

Photo-consistency: A voxel is considered to be photo consistent when its color is similar in all camera views that can see it.

C. Approach

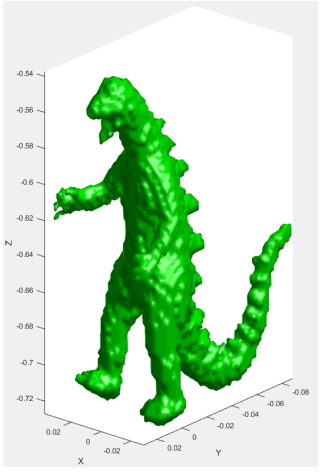
Since I did not focus on calibration in my portion of the project, I will leave out their approach to calibration from this section. Once the camera has been calibrated and projection matrices have been found for each view, the bounding box can be calculated from the silhouettes of the object in each view. Since the rotation axis of the turn table is known, taking the intersection of the back-projected silhouette bounding rectangles yields the bounding box of the object. Then the bounding box is divided into n voxels of equal size. Each of these n voxels is then projected back onto the images using the projection matrices obtained through calibration. Any voxel that is not inside the silhouette region of the view is removed and those voxels that are either on or inside the silhouette are marked as such and kept. The result of this process is a set of voxels that are guaranteed to enclose the object since what is left over is the maximal shape of the object for all views. To carve away the excess voxels and recover concave structure, photo-consistency constraints are applied to the silhouette based voxel model. To start, all voxels are assigned a high photoconsitency value. For each view i, the ray from the camera center C_i through the voxels visible in that view are traversed. Each voxel on the ray is projected onto the images i - 1, i, i + 1, and its photoconsistency value is calculated between the views. After traversing the whole ray, the voxel with the highest value is selected and all voxels between this voxel and the camera center have their photoconsistency values reduced as they move closer to the camera center. This is repeated for all rays in view i. Thus, after completion, most of the excess voxels loose their high photoconsitency value. Then, any voxel whose photoconsitency value is below a set threshold is removed/carved away. In this way a more accurate model is formed from the silhouette model estimation.

D. Critique

This approach is, in a way, a combination of the two approaches above except unlike the first approach this one does not fit a bounding surface sphere to the object but rather carves away a solid bounding box similar to the second method. The benefit of this approach is that most of the carving is done very efficiently and quickly using silhouettes, and only after the bounding model is found is the more costly photoconsitency carving done. This greatly reduces the run time, but approaches the quality possible with a pure photoconsitency based approach. This approach is not without its drawbacks though. Due to the requirement of a pre-calibrated camera, the variety of objects that can be reconstructed is someone limited (the size of the object must be such that it fits completely inside the camera view, otherwise the camera must be moved and re-calibrated). The controlled environment also limits the flexibility of the solution (known background and controlled lighting to reduce shadows and noise). The assumptions made are justified but limiting in this approach; however, the quality of reconstruction and lowered computation time make this a very useful solution.

VII. My Approach

My approach to the problem is mostly based off of the third paper. That is, I have a controlled environment and use a turntable and calibrated camera. The input to my algorithm is a set of images and their corresponding projection matrices. I then setup a structure to hold both of these things as well as the decomposed projection matrix parts K, R, T, and the corresponding silhouette. The silhouette is extracted using green-screening (known background). K,R, and T for each view can easily be extracted using QR decomposition of the projection matrix that is passed in. I then calculate the limits of the object using the translation T matrices of all of the views. Once I have the rough (**at least** containing the object) limits I do a quick carving of voxels. If any voxel is outside of the silhouette after being projected to the camera views, it is removed. This yields a very quick and rough estimation of the object and gives better bounding limits for a finer reconstruction. Once I have the refined limits of the object, I do a much more detailed carving using many more voxels. This yields a fast and relatively good looking model of the object (overall run-time is 7 seconds using a pointcloud of 100,000,000 voxels as the starting block to be carved). The run-time varies directly in relation to the number of points/increased resolution of the starting block (that is, a higher resolution yields a better result with a slower run-time).



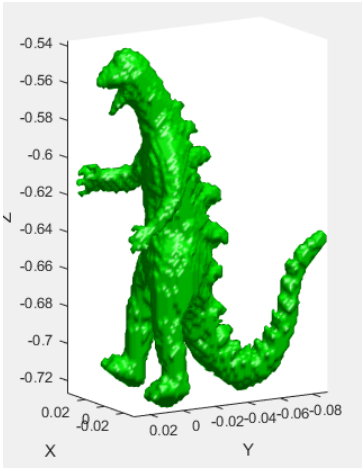
I initially struggled with the carving away of points since after even a very loose carve based on one image view, all of my points would disappear. I finally realized that the K, R, and T that I was calculating from the projection matrices were wrong. To obtain the proper decomposition, K and R needed to be calculated by performing QR decomposition on the inverse of the first 3x3 sub-matrix of the projection matrix.

```
function [K, R, T] = decompose_projection(P)
[q,r] = qr(inv(P(1:3,1:3)));
k = r(1:3,1:3);
R = inv(q);
% Since QR does not guarentee unitary determinant, correct for sign if
% needed.
if(det(R) < 0)
    R = -R;
    k = -k;
end
K = inv(k);
T = k*P(:,4);
% Get homogeneous K (normalize so that last element is equal to 1)
K = K/K(3,3);
% Get translation matrix
T = -R'*T;
end
```

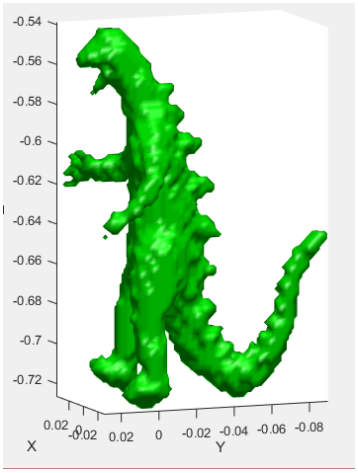
A struggle and point of issue initially with the project was how the foreground objects/silhouettes were going to be extracted. We had originally planned on using a camera that traveled on a fixed path around an object but with no prior knowledge of the background. Without having an image of just the background by itself to allow for background subtraction, this problem quickly become much more work than I had anticipated. To make sure that I could finish on time and work on the more relevant carving algorithm I decided to constrain the problem and assume a known background (in my case a greenscreen). This allowed for very clean extraction of silhouettes; however, had I had more time I would have liked to have been able to implement a more robust method without prior knowledge of the background of the object view.

I initially did not refine the bounds of the model before carving a block with higher resolution and number of points. As you can see in the two images below, the model that did not have refined limits before hand had a much poorer

reconstruction than the model that did (both used the same number of points and resolution for the carving, the only difference was the refinement of the bounds). Notice the large difference in detail in the left hand and face.



Model built using refined bounds after a quick carving



Model built without using refined bounds

I got the idea to refine the bounds before running a finer quality carving from the third paper which did a rough silhouette based model before running a higher quality photoconsistency check to refine it. Although I did not have the time to implement a photoconsistency refinement algorithm, doing two passes with tighter bounds the second time through proved to be an effective method of obtaining a higher quality model.

One glaring problem with this approach is memory usage. When calculating the model based off of a block of 150,000,000 voxels, about 8.5GB of memory is used. Unfortunately, I have not yet found a way around this issue. Since the system my testing was done on had 32GB of memory, this was not an issue; however, speed loss would likely occur on mobile systems (laptops) with less memory.

Overall, I really enjoyed researching the different methods of solving the task of 3D object reconstruction and only wish I had more time to implement a more robust silhouette extraction and photoconsistency refinement algorithm.

As an aside, all test images and projection matrices used as input to my implementation are from the Visual Geometry Group at the University of Oxford and can be found here:

http://www.robots.ox.ac.uk/ vgg/data/data-mview.html

REFERENCES

[1] V. Lippiello and F. Ruggiero, "Surface model reconstruction of 3D objects from multiple view," in *Proc. IEEE Int. Conf. Robot. Autom.*, Kobe, Japan, 2009, pp. 2400-2405

[2] K. N. Kutulakos and S. M. Seitz, "A theory of shape by space carving," *International Journal of Computer Vision.*, vol. 38, no. 3, pp. 199-218, 2000.

[3] A. Y. Mulayim, U. Yilmaz, and V. Atalay, "Silhouette-Based 3-D Model Reconstruction From Multiple Images," *IEEE Trans. Syst., Man, Cybern. B*, Vol. 33, No. 4, pp. 582-591, Aug. 2003.