Project Final Report

’Voxel Carving for 3D Human Reconstruction’

Maximilian Anzinger Alexander Fuchs Georg Henning Nils Keunecke

1 Abstract

Voxel carving is a known 3D reconstruction method for object reconstruction from a set of RGB images. We present our implementation of this method with the goal of achieving a good estimation of a human model placed in a scene. For pre-processing, we used pose estimation using the ChArUco markers set up in our scene, as well as different image segmentation methods. The voxel carving method is then used to extract a voxel representation of the mesh. Finally, post-processing is applied to both color our model according to the input, as well as improve the topology to achieve a convincing mesh. For results, we compare the performance across our chosen approaches and perform a qualitative comparison of the final model against a Structure from Motion approach. The project’s source is available on Github.

2 Introduction and Related Work

Voxel carving or space carving is a widely known method in computer vision, presented in the work by K. N. Kutulakos and S. M. Seitz [1]. RGB images taken from different camera perspectives with known intrinsic and extrinsic parameters are used for 3D reconstruction. This means it is an alternative to Structure from Motion (SfM) approaches used for scene reconstruction from RGB data, such as COLMAP [2]. These methods find correspondences across input images to project points into 3D space.
Voxel carving differs from this approach in that a predefined voxel representation of space is reduced until only the object of interest remains. We restrict ourselves to a static scene with a single camera setup, however, the method is applicable to real-time as shown in the work by Schick, A. and Stiefelhagen, R. [3].
In figure 1, the pipeline of our implementation is shown. Initially, pre-processing is done on the input images in order to obtain camera extrinsics as well as extract a mask of the object. The pose estimation is performed using ChArUco Markers placed in our scene. For the image segmentation, we used methods including RGB thresholding, k-means clustering as well as a pre-trained segmentation RNN. The obtained data is then used in our voxel implementation. Post-Processing in the form of color reconstruction and morphological closing is then performed before the voxel grid is converted to a triangle mesh using the marching cubes algorithm.

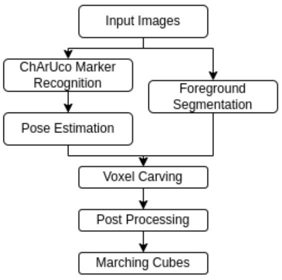


Figure 1: Final Pipeline

We provide benchmarking results for our implementation's runtime, as well as a qualitative comparison of the final human reconstruction to results from COLMAP.

3 Methodology

In the following section, we lay down the setup and design choices for our voxel carving implementation.

3.1 Environment setup

Estimating the projection matrix of the camera is essential for all steps of the presented 3D reconstruction process. This was achieved using a checkerboard pattern of known size using the implementation in OpenCV [4].
All image segmentation approaches profit from noise-free environments which were achieved with consumer-grade green screens, and consistent indirect illumination.

3.2 Pose Estimation

Pose estimation is performed using ChArUco boards[5]. ChArUco boards combine the advantage of traditionally used checkerboards and ArUco markers. They give a more precise position estimation and can sustain occlusions of a subset of markers and checkerboard tiles to a certain degree. This method works independently of the size of markers. For household items, an A4-sized paper ChArUco board is used, while the *human* data set relies on a board of size $1.5m$x$1m$. Given a calibrated camera and known size of checkerboard tiles and ChArUco markers, a 3D pose relative to the ChArUco board can be computed using projective geometry.

3.3 Image segmentation

Due to the importance of accurate segmentation maps, two traditional segmentation methods namely segmentation by RGB color value and k-means clustering of color values have been implemented as well as a Deep Learning based approach using Mask R-CNN[6].

3.3.1 RGB Color Masking

The idea is very straightforward. One or more masks are defined by setting a range for the red, blue, and green channels of the image respectively. Pixels are evaluated as valid if they fall within the defined range for all channels. Masks can be stacked by applying the *max* operation for each pixel over all masks. Due to the manual labor required, the objects of interest should ideally be monochrome and equally illuminated.

3.3.2 K-Means Clustering

K-means clustering attempts to divide the image into regions of similar RGB color values to find areas in the image with high similarity. In contrast to the aforementioned color segmentation method, no user input is required to define scene-specific color values in this case. However, after segmentation, the user has to select which of the clusters represents the object of interest. Like color segmentation, this method requires simple, ideally equally illuminated, monochrome objects.

3.3.3 Mask R-CNN

Because color consistency of the entire scene is hard to ensure without a professional green-screen environment, we use a pre-trained neural network to segment images in the *human* data set. Mask R-CNN [6] is a two-stage instance segmentation model consisting of a regional proposal network, recommending potential bounding boxes, and CNN which performs classification and bounding box regression. Our adapted model takes an RGB image as input and returns a mask highlighting the detected person.

3.4 Voxel Carving

For the carving process, we provide two different procedures. The standard approach iteratively carves each input image. Therefore, each voxel is projected from world to camera coordinates. The previously discussed segmentation can then be used to differentiate between back- and foreground. This approach immediately proves to be inefficient ($\Theta(nxyz)$ time complexity, with x, y, z being the dimensions of the voxel model and n the number of images) and not feasible for high-resolution models or a large number of input images.

Our second approach is influenced by the work of Kutulakos and Seitz [7] and Gaillard et al. [8]. This greedy algorithm uses a queue to keep track of voxels that are yet to be checked. For each voxel of the queue, the projection into camera-space is performed for a corresponding image until either an image was found that carves the voxel or the voxel is determined to be a point on the object's surface. Any voxel revealed by the removal of the current voxel is added to the queue. Hence, only voxels that are carved and those on the surface of the object have to be processed. While the worst-case running time of this procedure remains in $\mathcal{O}(nxyz)$, any real-world data set will result in significantly better performance since voxels encapsulated by the surface of the model may never be processed. Overall this alternative method provides a solution for processing larger inputs with only minimal increased complexity compared to the original implementation.

To convert our voxel model into a triangular mesh representation compatible with the ".OFF"-format we use an adapted version of the marching cubes algorithm [9] that can be applied at every point of the process.

3.5 Post-processing

3.5.1 Color Reconstruction

For color reconstruction, a sweep through all voxels and images is performed to obtain the RGB values of the individual pixels. Additionally, information about the distance between the camera and the voxel is stored. Based on this data it is possible to either choose the RGB value of the closest observer or compute the average of multiple observers. Further extensions, e.g. weighted average (by distance) could be added easily.

3.5.2 Morphological Closing

Morphological closing[**?**] is performed using a 3x3x3 kernel which is used to apply dilution on the voxel grid, followed by erosion. This results in a smooth surface that might otherwise contain holes or protrusions due to inconsistencies in the segmentation masks. This ensures a clean triangle mesh without artifacts after applying marching cubes.

4 Results

Two data sets were used to test the presented pipeline. The *box*-data set contains 8 images with an A4 paper-sized ChArUco board. The *human*-data set contains 23 images with a $1.5m * 1m$ ChArUco board. We use the *human*-data set to show the performance of the Mask R-CNN image segmentation in combination with average coloring and morphological closing at a resolution of $5mm$ edge length per voxel. The results are compared with those of COLMAP[2], a popular SfM framework.

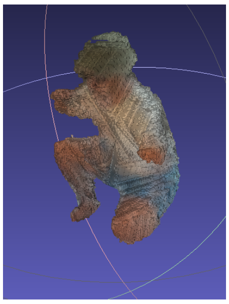


Figure 2: The result of the carving.

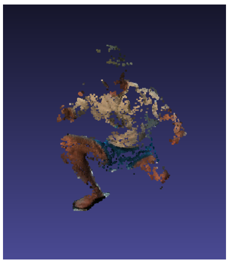
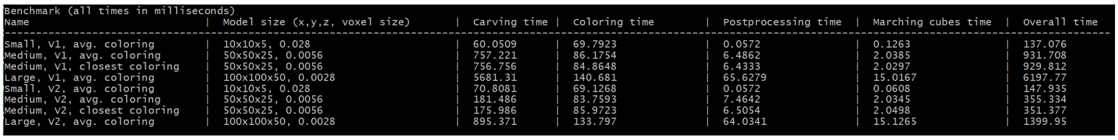


Figure 3: The result from COLMAP for comparison.

To evaluate our theoretical analysis of the different voxel carving implementations we provide a consistent benchmark compatible with every data set. For each approach four test cases are executed: small ($10x10x5$), medium ($50x50x25$) with both coloring-methods, and large ($100x100x50$) model size. The time needed for each part of the process (carving, coloring, postprocessing, model conversion) is measured separately. The results of a benchmark run on the *box*-data set can be seen in , where ”V1” is the original voxel carving approach while ”V2” is our greedy implementation. The output shows, that there are huge runtime differences between both carving methods that become larger with increasing model size. Tests with fewer images show similar results, again supporting the theoretical expectations. Runtimes apart from the carving process show only very small differences likely depending only on variations in CPU clock boost.



```
Benchmark (all times in milliseconds)
Name                             |  Model size (x,y,z, voxel size)        |  Carving time |  Coloring time          |  Postprocessing time  |  Marching cubes time  |  Overall time
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Small, V1, avg. coloring         |  10x10x5, 0.028                        |  60.0509      |  69.7923                |  0.0572               |  0.1263               |  137.076
Medium, V1, avg. coloring        |  50x50x25, 0.0056                      |  757.221      |  86.1754                |  6.4862               |  2.0385               |  931.708
Medium, V1, closest coloring     |  50x50x25, 0.0056                      |  756.756      |  84.8648                |  6.4333               |  2.0297               |  929.812
Large, V1, avg. coloring         |  100x100x50, 0.0028                    |  5681.31      |  140.681                |  65.6279              |  15.0167              |  6197.77
Small, V2, avg. coloring         |  10x10x5, 0.028                        |  70.8081      |  69.1268                |  0.0572               |  0.0608               |  147.935
Medium, V2, avg. coloring        |  50x50x25, 0.0056                      |  181.486      |  83.7593                |  7.4642               |  2.0345               |  355.334
Medium, V2, closest coloring     |  50x50x25, 0.0056                      |  175.986      |  85.9723                |  6.5054               |  2.0498               |  351.377
Large, V2, avg. coloring         |  100x100x50, 0.0028                    |  895.371      |  133.797                |  64.0341              |  15.1265              |  1399.95
```

Figure 4: Benchmark output for *box*-data set run on an Intel Core i7 CPU @4.5GHz

5 Discussion & Evaluation

The results show for both the *box*-data set as well as the *human*-data set, that the objects are reconstructed sufficiently to recognize them and are relatively detailed. As we do not have any metric available to analyze our results quantitatively, we limit ourselves to a qualitative evaluation. A synthetic data set could solve this in the future, as it provides a ground truth and allows testing the quality of each component of the pipeline individually. The addition of post-processing shows clear improvements with significantly more realistic surfaces. The color reconstruction shows some shortcomings, especially in areas of large color differences, inferior to contemporary SfM or Optical Flow methods[10]. The implemented method for voxel carving has multiple serious issues. In contrast, to other reconstruction methods, voxel carving can inherently only reconstruct convex shapes due to the nature of carving lines out of the initial voxel block. Additionally, in its current implementation, the model carves a voxel if it is not recognized as part of the foreground in a single image. This highly penalizes inconsistencies in the segmentation maps as can be seen with the missing arms in the *human*-data set. While capturing the data sets, it was noted that it is near impossible to take images orthogonal to the ChArUco-board as the markers are not captured correctly. This leads to a lack of details, on top of the reconstructed box.
Finally, the current implementation of the greedy-carving approach has still inconvenient restrictions: (1) at least one image of the data set must depict the origin of the ChArUco board and, (2) the object must not stand on the origin. Both issues may be resolved by either allowing the user to input the initial voxel to start the process or adding sufficiently many voxels chosen at random to the processing queue.

5.1 Future Work

In the future, it could prove valuable to implement a statistical approach to voxel carving to preserve more features of the object of interest[11]. The same goal could be achieved with more consistent segmentation maps or a professional green-screen environment. One could use GPUs to make the approach real-time capable[3]. Finally, non-rigid and deformable meshes would make it possible (in combination with the other proposed improvements) to capture human video sequences and turn them into 3D models.

References

[1] Kiriakos N Kutulakos and Steven M Seitz. A theory of shape by space carving. *International journal of computer vision*, 38(3):199–218, 2000.

[2] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] Alexander Schick and Rainer Stiefelhagen. Real-time gpu-based voxel carving with systematic occlusion handling. In Joachim Denzler, Gunther Notni, and Herbert Süße, editors, *Pattern Recognition*, pages 372–381, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[4] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[5] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99)*, pages 85–94, 1999.

[6] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[7] Kiriakos N. Kutulakos and Steven M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision*, 38(3):199–218, 2000.

[8] Mathieu Gaillard, Chenyong Miao, James C Schnable, and Bedrich Benes. Voxel carving-based 3d reconstruction of sorghum identifies genetic determinants of light interception efficiency. *Plant direct*, 4(10):e00255, 2020.

[9] William E. Lorensen and Harvey E. Cline. Marching cubes: a high resolution 3d surface construction algorithm. *Seminal graphics*, 1996.

[10] Felix Wimbauer, Nan Yang, Lukas Von Stumberg, Niclas Zeller, and Daniel Cremers. Monorec: Semi-supervised dense reconstruction in dynamic environments from a single moving camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6112–6122, 2021.

[11] Adrian Broadhurst, Tom W Drummond, and Roberto Cipolla. A probabilistic framework for space carving. In *Proceedings eighth IEEE international conference on computer vision. ICCV 2001*, volume 1, pages 388–393. IEEE, 2001.