AR Marker and Voxel Carving

Course „3D Scanning and Motion Capture“

Group 2
Ali Canberk Anar, Ruiyun Xie, Richard Nai, Thea Kramer

June 05, 2019

computer graphics & visualization

Outline

- Related Work & Motivation
- Dataset
- Implementation
  - Camera Calibration
  - AR Marker Detection
  - Pose Estimation
  - Background Segmentation
  - Voxel Carving
  - Rendering
- Result

Team

Richard
Nai

richard.nai@tum.de

Ruiyun
Xie

ruiyun.xie@tum.de

Ali Canberk
Anar

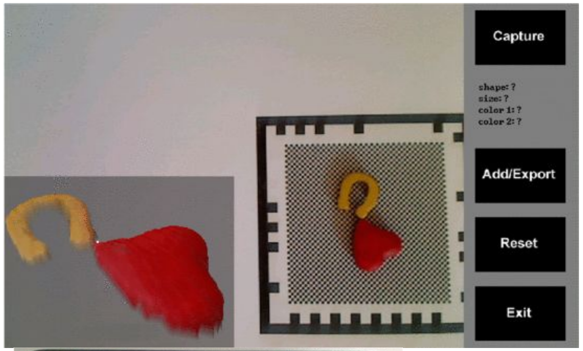
canberk.anar@tum.de

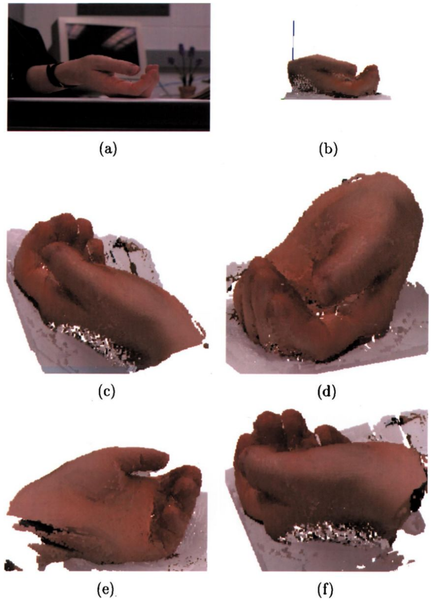
Thea
Kramer

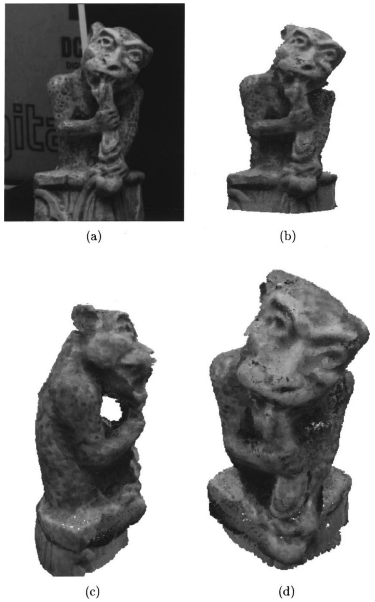
thea.kramer@tum.de

Related Work

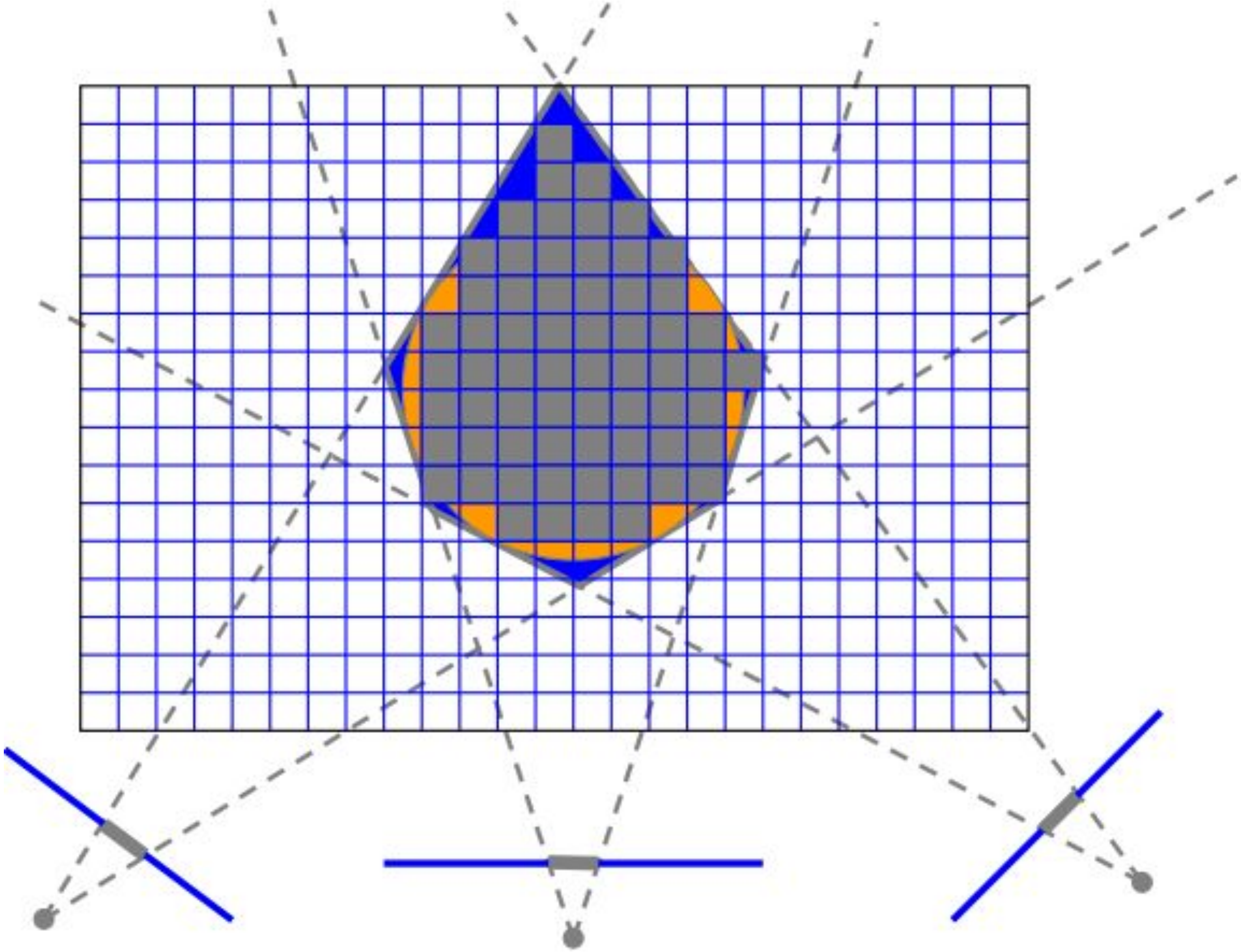
- Kiriakos N. Kutulakos and Steven M. Seitz. A theory of shape by space carving.International Journalof Computer Vision, 3(38):199–218, 2000.
- Andreas Hartl, Lukas Gruber, Clemens Arth, Stefan Hauswiesner, and Dieter Schmalstieg. Rapid reconstruction of small objects on mobile phones.Proceedings of CVPR 2011, pages 20–27, 2006.





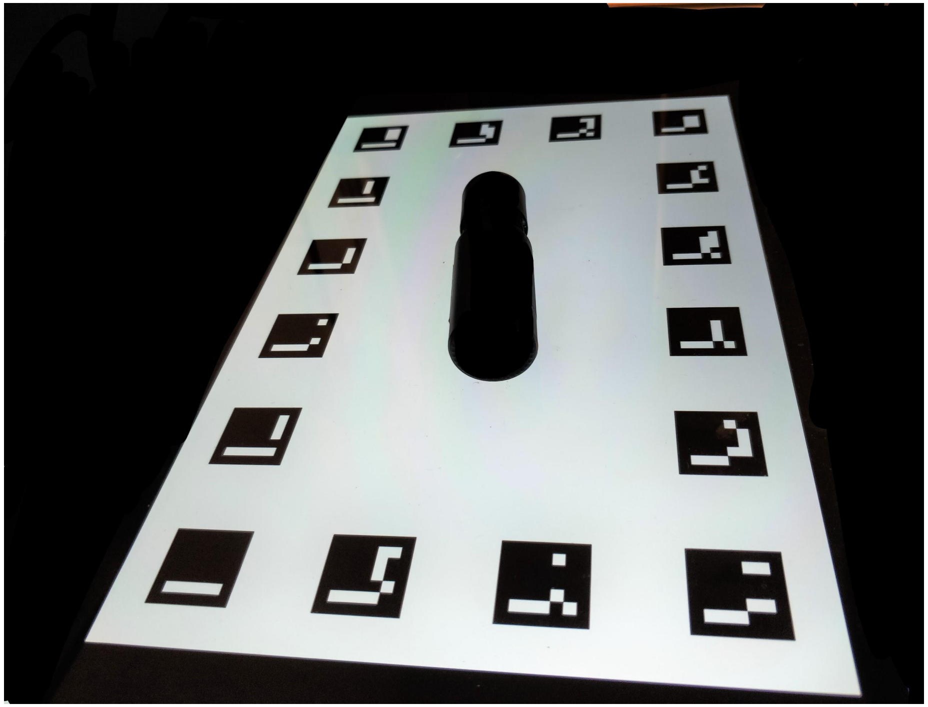


Motivation



Dataset

- Two different datasets created with a smartphone.
- Both datasets had the object sitting on a predefined marker background



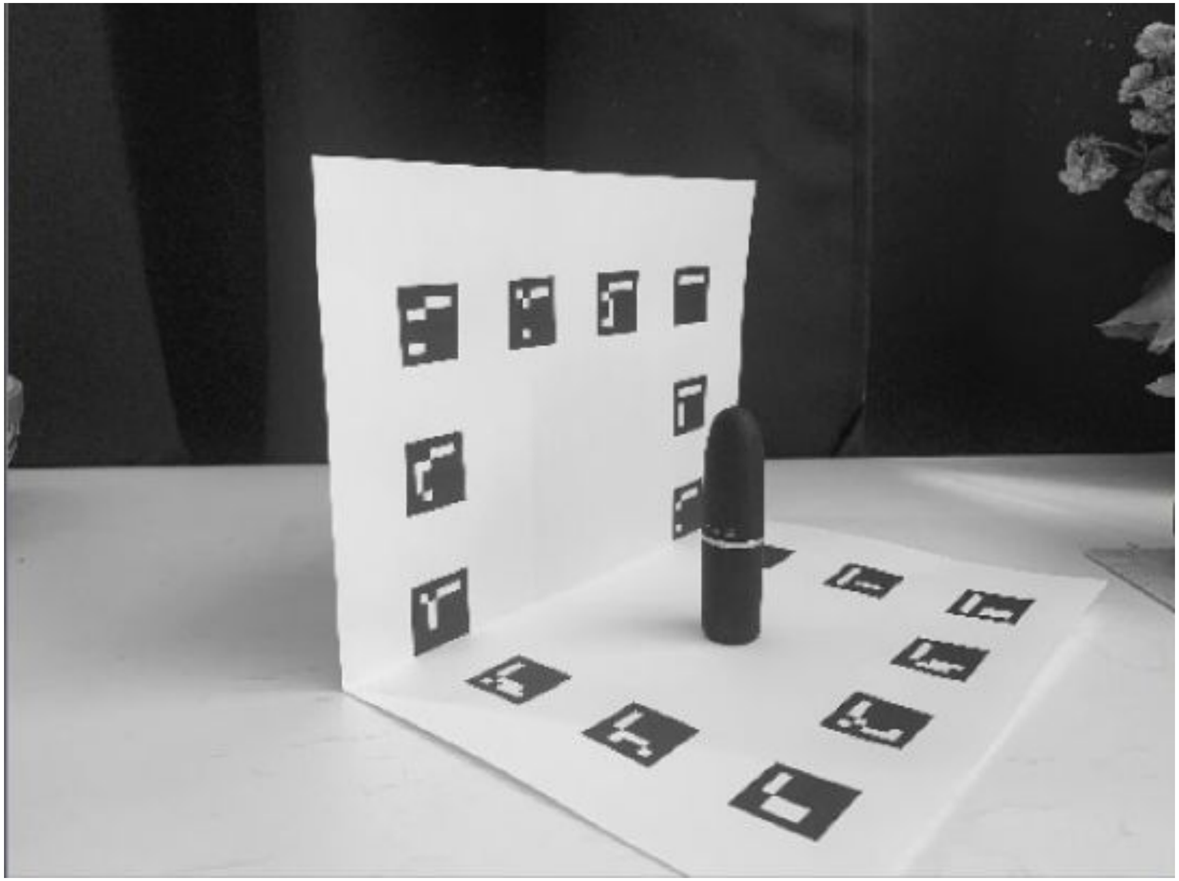
Example from First Dataset

Final Dataset

- Marker background paper instead of a tablet
- Camera enhancement with “pro mode”:
  - fixed focus
  - fixed exposure
  - fixed white balancing

  → Little improvement on the result
- 8 images (fast computation and reasonable reconstruction quality)

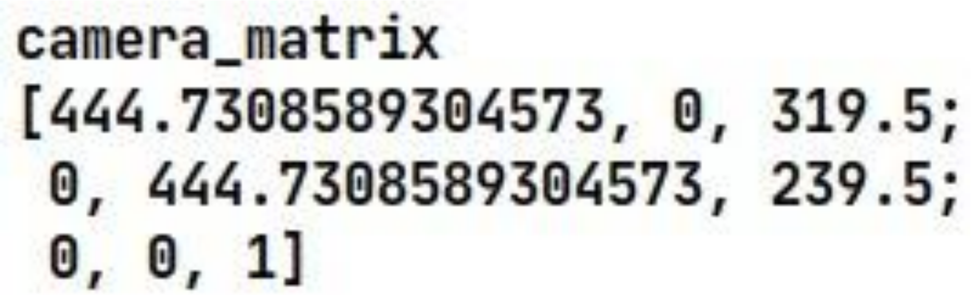
Final Dataset



Example from Second Dataset

Camera Calibration

- Camera pose estimation done by using camera intrinsics matrix of the capturing device.
- Camera calibration with the OpenCV Library and the chessboard model as the calibrating object.

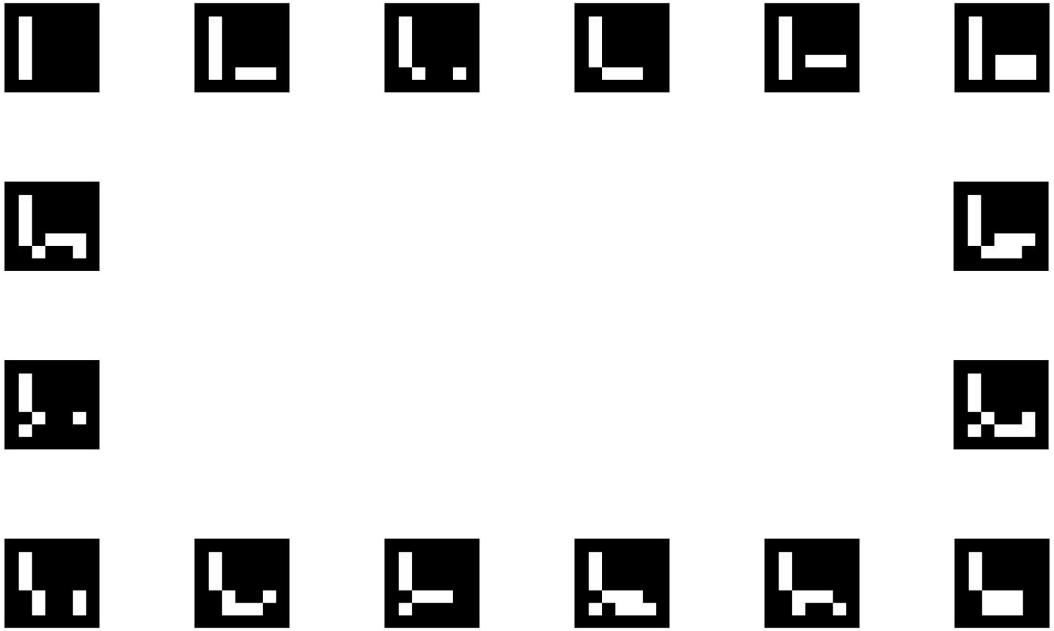


```
camera_matrix
[444.7308589304573, 0, 319.5;
 0, 444.7308589304573, 239.5;
 0, 0, 1]
```

Our Camera Intrinsics Matrix

AR Marker Detection

- ArUco markers were placed on an RGB image
- Top left corner of top left marker is (0,0,0)
- Each corner is then defined to be 10 apart from the previous one



Pose Estimation

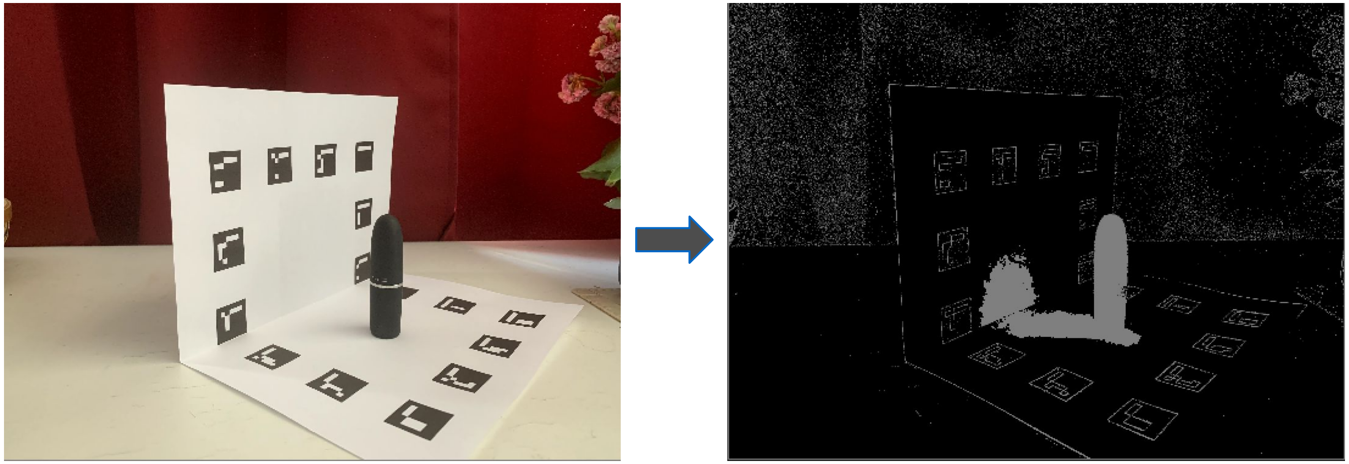
- Pose Estimation determines the translation and rotation matrices (transformation camera to world frame)
- OpenCV’s PNP algorithm:
  - Marker corners were matched from the image to a predefined world reference grid
  - Translation from camera to world origin Tc was computed
  - Inverse was taken to determine translation from world to camera

$$T_c = \begin{bmatrix} & & & T_x \\ & R_{3\times3} & & T_y \\ & & & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Pose Estimation

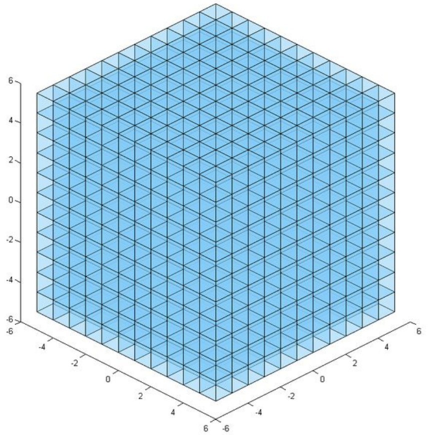
Background Segmentation

- Background subtraction with OpenCV’s Mixture of Gaussians 2 (MOG2 algorithm)
- Comparison of the background with and without the object
- Transform into silhouette:
  - foreground (object) to white pixels
  - background to black pixels



Voxel Carving

- 300x300x300 voxel grid starting at (0,0,0)
- Calculation of signed distance values of silhouette (*dist*)
- Mapping of each voxel to image pixel (→ pose estimation matrix)
  - Corresponding pixel white (object) → set voxel to *dist*
  - Corresponding pixel black (background) → set voxel to 0
  - Corresponding pixel not within image coordinates → set voxel to -1
- Voxel grid array input for rendering



Rendering

- VTK library
- Structured points with signed distance values
- Surface representation using Marching Cubes Algorithm
- Mesh topology recreation
- Interactive render window for visualization

Result