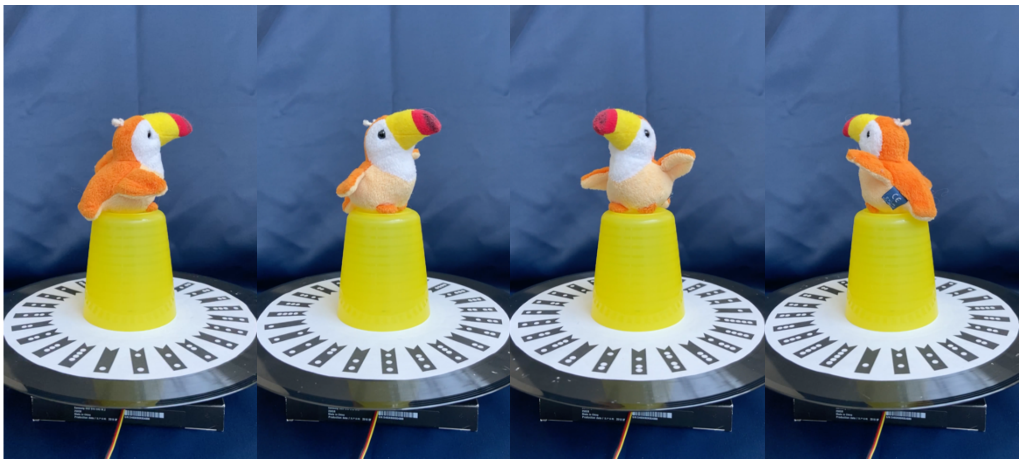
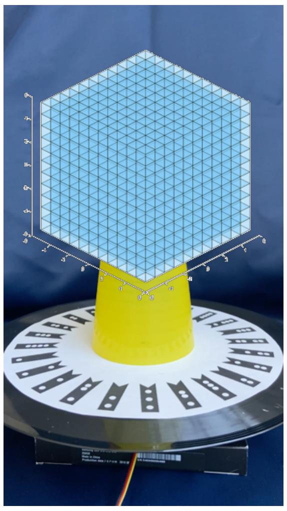
Geometric and 3D Computer Vision 2022

Final Project – “Silhouette-based space carving”

Your goal is to implement technique known as “space carving” [1] to reconstruct the shape of a 3D object from multiple photographs taken at known but arbitrarily distributed viewpoints. An object is placed on top of a rotating plate together with a custom-designed fiducal marker (see the following sections for details). The background is made of a uniform-colored material to be easily distinguished from the target object. A calibrated camera is placed in front of the object capturing the scene throughout an entire rotation, as shown in the following images:



The volume occupied by the object is represented by a discrete set of voxels distributed on a cube of size N x N x N. A voxel can be seen as the 3D extension of a “pixel” describing a property of a certain region of space. In this case, the property is being either occupied or not by our target object:



At each frame, a set of 3D rays exit the camera starting from the optical center and passing through each pixel of the image. Every ray may intersect some voxels before reaching either the background or the object itself. If a ray reaches the background without touching the object, all the intersected voxels can be "carved" (ie. removed) as they represent empty space. On the contrary, if a ray reaches the object, at least one of the intersected voxels is part of the object, so they must not be removed from the set.

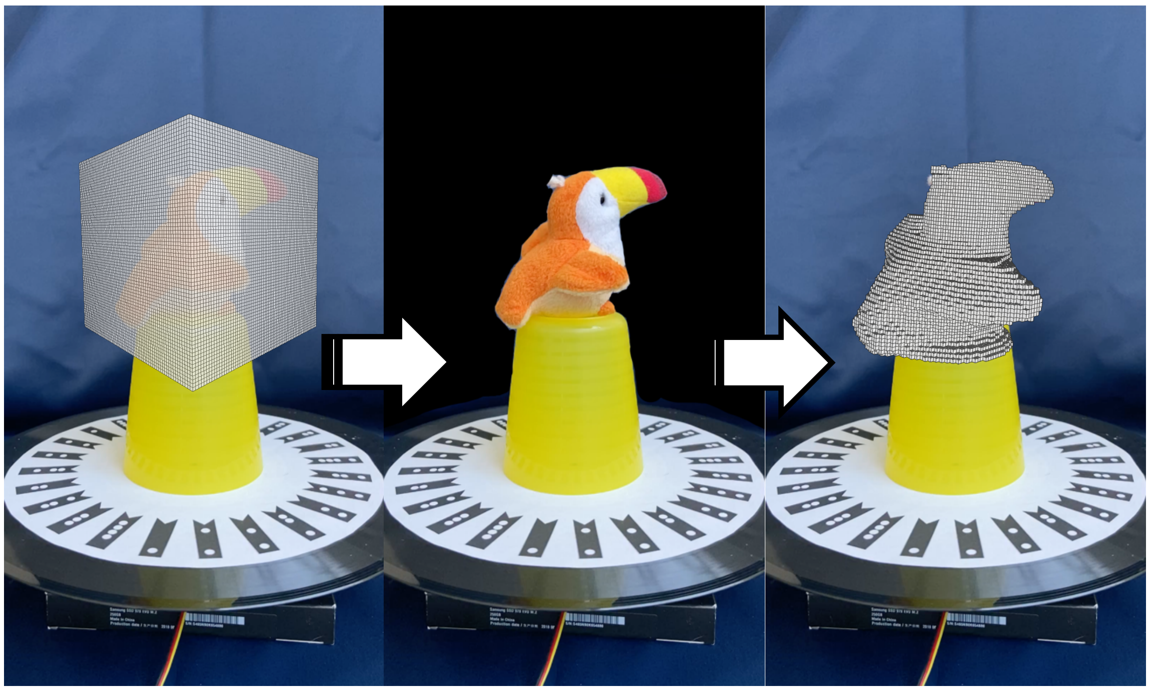
For the space-carving technique being effective, two sub-problems must be solved at each frame i:

1. Estimate the position of the camera with respect to the object (ie. the camera pose R_i, T_i)
2. Find the "silhouette" of the object by clustering the pixels as part of the background or foreground.

The complete algorithm works as follows:

- Let V be an initial set of voxels distributed in a N x N x N cube
- For each image:
    - Compute the camera Projection matrix $P = K [R_i \ T_i]$
    - Project all the voxels onto the image
    - Check if the projection of the voxel is inside or outside the silhouette. In the latter case, remove the voxel from V

When all the frames have been processed, the remaining set of voxels define the volume occupied by the object:



Calibration

The camera must be calibrated to obtain the projection matrixes for each frame. Together with the space carving program, you are also asked to create a program for calibrating a camera using multiple views of a given chessboard.

The calibration software should work **without any user intervention**. In other words, it should load the provided calibration video file, detect the chessboard for at least 20 different frames (possibly evenly distributed among the sequence), and estimate the camera intrinsic parameters using the OpenCV function `calibrateCamera`.
After the calibration, the camera parameters should be written to a file and used for the subsequent operations. I suggest using a 5-parameters polynomial distortion model.

For additional information on how to use OpenCV to calibrate the cameras refer to the following page:

https://docs.opencv.org/master/dc/dbb/tutorial_py_calibration.html

Video data

A package containing 4 different video sequences (plus the calibration) can be downloaded at the following URL:

https://www.dais.unive.it/~bergamasco/teachingfiles/G3DCV2022/data.7z

Content:

- `calibration.mp4` the video sequence of the calibration chessboard
- `obj1.mp4 … obj4.mp4` video sequences with 4 different rotating objects
- `marker.svg` the marker used in the turntable. You can open this file with Inkscape to inspect its dimensions

The developed project should fulfill all the requirements described below considering the provided data. In other words, parameters and algorithms can be tuned to “work well” with the given samples.

Requirements

The project must contain 2 different programs:

1. A *“camera calibrator”* that loads the provided calibration video and computes the intrinsic parameters of the camera (the intrinsic matrix K and the lens distortion vector assuming a 5-parameters model) **without any user intervention**.

2. The space carving program taking in input a video sequence and producing in output the list of voxels contained in the interior volume of the reconstructed object. The whole analysis must be performed on all the frames of the selected video sequence **without any user intervention.** However, you can modify algorithm parameters (if needed) for each video sequence.

The output list of voxels can be expressed either as a list of points placed at the voxel center or by explicitly defining the geometry of each cube. In both the cases, they must be saved on a PLY file (see http://paulbourke.net/dataformats/ply/) that can be opened by the freely available Meshlab visualizer (https://www.meshlab.net/).

The 2 programs **must be written in Python3** and can only use the NumPy and OpenCV libraries. Additional libraries like Open3D are forbidden.

Additional notes:

There are no constraints on the number of function/classes/py files produced. You are highly encouraged to give visual feedbacks to better understand each algorithm involved. For example, the analysis program could show the detected marker points, the projected voxels, etc. Any "visual" debug information useful to explain the operations involved are welcome and will contribute to the final evaluation of the project during the oral examination.

A reasonable processing speed will be evaluated as a positive feature of your work, although not mandatory to pass the exam.

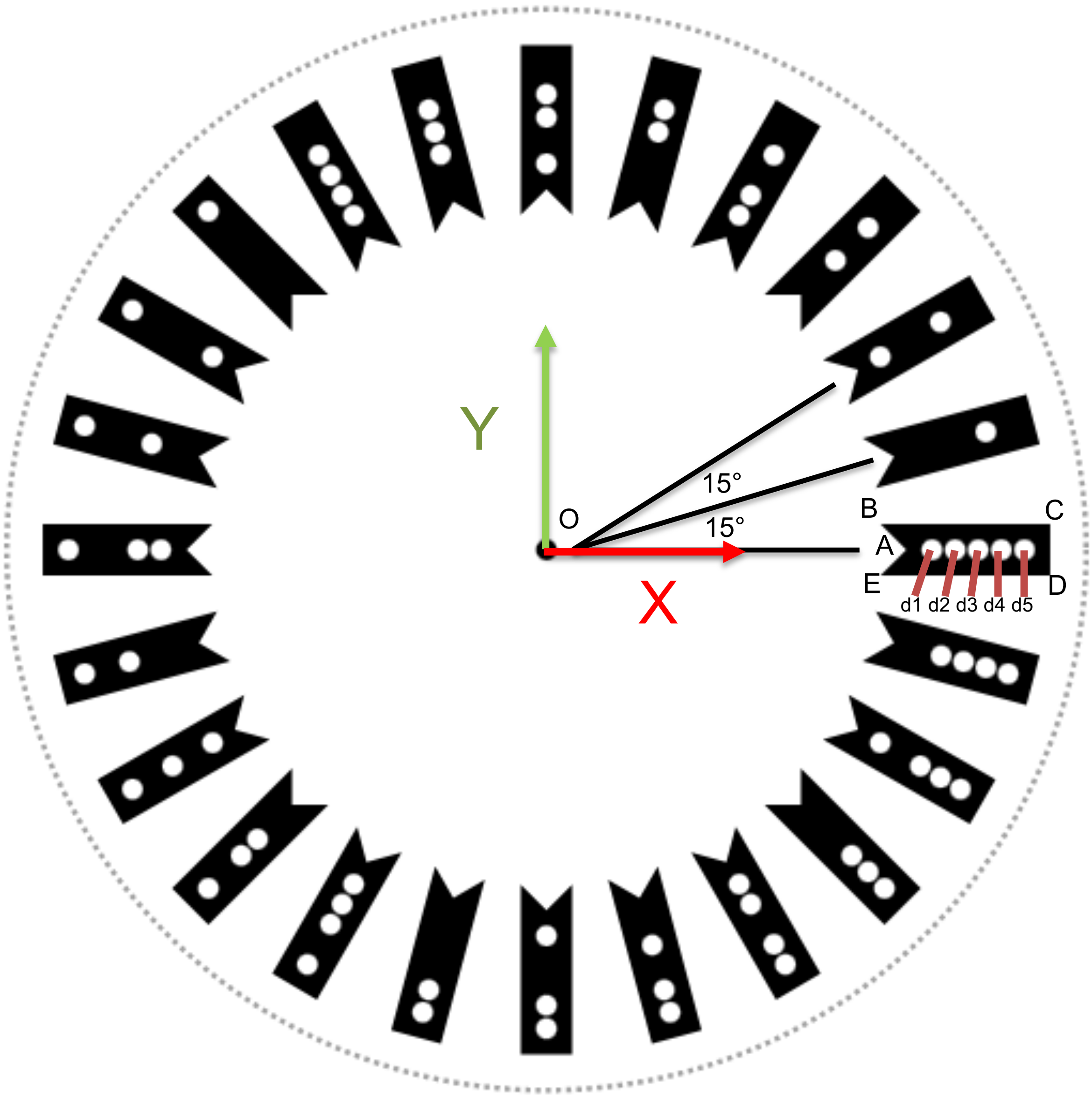
Comment your code whenever possible. Since no additional report is required, comments are a good way to clarify what your code is supposed to do.

Suggestions

- To recover the camera pose from the marker, you can either factorize the Homography matrix mapping points from the marker reference frame to the camera or **use the OpenCV function solvePnP** (https://docs.opencv.org/4.x/d9/d0c/group__calib3d.html#ga549c2075fac14829ff4a58bc931c033d). This latter method tends to be more stable in case of noise, especially if using the method SOLVEPNP_IPPE.
- Once the camera pose has been recovered, try to project back the marker points onto the image plane to check the error between the projected position and the expected marker point location. You should aim for RMS error below 1px for good results when carving the voxel grid.
- It is a reasonable simplification to consider each voxel as a single 3D point instead of a cube. In other words, to check if a voxel is visible or not, you can just consider if the voxel center projects inside or outside the object silhouette.

Fiducial marker

The marker placed at the turntable is designed to provide a good estimation of the camera pose even if partially occluded by the object. It is composed by 24 “marks” distributed radially around the center with angle increments of 15°. Each mark is a black polygon with 5 vertexes, 1 concave and 4 convex. In the internal area, 5 black&white dots are evenly distributed to binary encode the mark index (black dot=1, white dot=0).
Coordinates of each mark vertex can be recovered from the `marker.svg`, and are briefly summarized here:



Point	Coordinates (x,y,z)
O	0, 0, 0
A	70, 0, 0
B	65, 5, 0
C	98, 5, 0
D	98,-5, 0
E	65,-5, 0
D1	75, 0, 0
D2	79.5, 0, 0
D3	84, 0, 0
D4	88.5, 0, 0
D5	93, 0, 0

References

[1] KUTULAKOS, Kiriakos N.; SEITZ, Steven M. A theory of shape by space carving. *International journal of computer vision*, 2000, 38.3: 199-218.
https://www.cs.toronto.edu/~kyros/pubs/00.ijcv.carve.pdf

Exam Information

Final course exam consists in an **oral discussion about the project.**

When ready to take your final exam, package the source code in a zip file named `<name>_<surname>_exam.zip` and submit it via Moodle. Then, notify me via mail (filippo.bergamasco@unive.it) so that we can arrange a date (usually within a couple of weeks from the submission). Please, **do not submit any data related to the project (video files, images, calibration data, etc.)**.

During the exam, I will ask you to explain all the interesting parts of the source code, focusing on the implementation details and the algorithms involved. You are supposed to discuss strengths and limitations of any choice made in your implementation. During the discussion, you will also have to show a demo of your project.

I'll consider the exam passed with full marks if you demonstrate proficiency on the computer vision techniques we have seen during the course in relation to the project you have developed. In other words, my goal is to evaluate your ability to combine different vision/image processing algorithms to fulfill the required task

Your project will be evaluated as follows:

- Camera calibration: 5 points
- Pose estimation: 10 points
- Silhouette extraction (background/object segmentation): 6 points
- Voxel carving: 9 points
- All the previous + voxel coloring: “Lode” (honors)

In any case, the final score finally depends on the oral discussion. The point distribution is meant to show the difficulty of each part.

Important note:

If you are not able to fulfill some of the tasks, you can download and use the pre-processed data in this package:

https://www.dais.unive.it/~bergamasco/teachingfiles/G3DCV2022/processed.7z

In that case, all the parts that are missing or not fully implemented will badly affect your final score. For example, if you implement camera calibration, silhouette extraction and voxel carving but not the pose estimation, the maximum score you can obtain is 20 (5+6+9).

For any question feel free to mail me at filippo.bergamasco@unive.it

20/04/2022
Filippo Bergamasco
Geometric and 3D Computer Vision a.a. 2021/2022