Short project

Computer Vision (CV)

Design and implementation of a 3D reconstruction algorithm from stereo images.

Authors: Victor Escribano Garcia

Oriol Contreras Pérez

Teacher: Dr. Joan Aranda

Date: December 14, 2022

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Tècnica Superior d'Enginyeria
Industrial de Barcelona
ETSEIB

Contents

1. Introduction

Stereo photogrametry is a widely used technique for obtaining a three-dimensional (3D) representation of a scene from two or more images taken from different viewpoints. This approach allows for the accurate reconstruction of the geometry and layout of objects in a scene, providing valuable information for a wide range of applications, including 3D mapping, robotics, and computer vision.

In this project, we will demonstrate how to use MATLAB to perform stereo photogrametry by generating a stereo disparity map and using stereo feature triangulation to reconstruct a 3D scene. Stereo disparity refers to the difference in pixel position between corresponding points in the left and right images of a stereo pair, and can be used to calculate the depth of objects in a scene. Stereo feature triangulation, on the other hand, involves identifying corresponding points in the left and right images of a stereo pair and using their known positions to triangulate the 3D position of the points in the scene.

In addition to generating a 3D reconstruction of a scene, we will also show how to track the camera pose by means of structure from motion. This involves using the known camera parameters and the corresponding points in the current and previous image frames to estimate the 3D position and orientation of the camera as it moves through the scene. This information can be used to create a detailed 3D model of the scene, allowing for further analysis and interpretation.

Overall, this project provides a comprehensive overview of the techniques and algorithms used in stereo photogrametry, and demonstrates their practical implementation in MATLAB. It is a valuable resource for anyone interested in learning about this powerful approach for obtaining 3D information from images.

2. Camera parameters and calibration

To understand how to obtain 3D distances from 2D images, first of all the parameters that correlates this two frames has to been extracted. As the images are captured by means of two fixed cameras the intrinsic and extrinsic parameters have to be extracted through camera calibration. On this chapter this parameters will be defined and explained in order to obtain a good calibration and a final correct reconstruction.

2.1 Intrinsic parameters

The intrinsic parameters of a camera refer to the internal characteristics of the camera that affect the image formation process. These parameters are specific to each individual camera and can affect the overall quality and accuracy of the resulting images.

Internal parameters include the focal length, principal point, and skew. The focal length is a measure of the magnification of the camera lens, with a higher focal length resulting in a more magnified image. The principal point is the center of the image, and the skew is a measure of the non-orthogonality of the x and y axes in the image plane.

Distortion parameters describe the nonlinear distortions that can occur in the image formation process. These include radial distortion, which describes the curvature of lines in the image, and tangential distortion, which describes the shifting of the image plane relative to the lens.

2.2 Extrinsic parameters

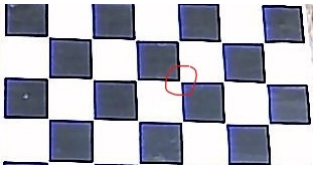
The extrinsic parameters describe the relative position and orientation of the two cameras. This information is used to align and rectify the images from each camera, enabling the creation of 3D reconstructions of the scene. The extrinsic parameters can be represented using a transformation matrix, which describes the transformation from the coordinate system of one camera to the coordinate system of the other camera. This transformation matrix can be used to map points in the image of one camera to the corresponding points in the image of the other camera, allowing for the calculation of the distance between the two cameras.

2.3 Stereo calibration

Stereo camera calibration is a technique used to determine the intrinsic and extrinsic parameters of a stereo camera system. This involves capturing images of a known calibration target, such as a checkerboard pattern, from multiple views using the two cameras, and using these images to calibrate the cameras. Once the intrinsic and extrinsic parameters have been determined, they can be used to rectify and align the images from the two cameras, enabling the creation of 3D reconstructions of the scene. This information can also be used to correct for the effects of camera motion and to perform tasks such as object tracking and pose

estimation. It is important to correctly detect the pattern during stereo camera calibration because the accuracy of the resulting intrinsic and extrinsic parameters depends on the quality of the detected corners. If the corners are not detected accurately, the calibration process will produce incorrect or suboptimal results. On figure 2.1 can be seen how this overexposed image can affect to the detection of the checkboard pattern.

(a) Check-board stereo images



(b) Error in check-board due to overexposure

Figure 2.1: Calibration problems

On figure 2.2 it can be seen the calibration results of a sequence of calibration images, resulting on the external transformation between the to cameras.

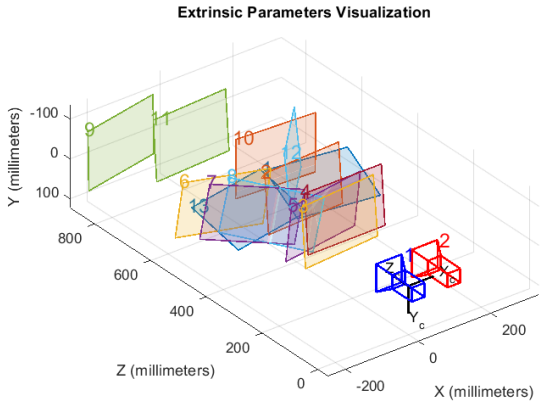


Figure 2.2: Pose between the 2 stereo cameras(left camera as the main on on the origin).

Notice that when calibrating the cameras two main issues can appear:

- Free hand image capturing: When capturing the images holding the cameras on free hand, some error was induced due to hand tremoring, not representing correctly the true transform between cameras.
- Camera synchronization: Similarly the cameras where not shooting the images with the same timestamp, if that issue is added with the previous one, the images obtained are not representative enough of the actual setup of the system.

To solve the second problem some solutions can be implemented such as synchronize the shutter of the cameras, some cameras can accept software triggering or some PWM hardware triggering to capture the images with the same timestamp, but in this project this solution will not be implemented but a more simple solution is applied, explained on section 2.4.

2.4 Support

To solve the previous problem regarding to free hand shake when capturing images, a camera support was specially designed to capture the images on a static surface, in this case a tripod. This tripod support ensures a stable image pair extraction, and allow us to rotate the cameras giving us yaw and pitch angle to position the camera views. Also if the scene to reconstruct is static (it does not have moving objects) the timestamp problem can be ignored. On figure 2.3 the designed support can be seen along with the stereo camera pairs attached.

Figure 2.3: 3D printed camera support mounted on a goPro tripod.

3. 3D reconstruction

On this chapter the 3D stereo reconstruction will be performed by means of 2 different methods:

1. Disparity map
2. Triangulate keypoints

Once the 3d reconstruction on a single view the structure from motion algorithm will be implemented on a consecutive set of images to estimate the pose of the camera during those frames.

3.1 Disparity map

A disparity map is a visual representation of the difference in distances between corresponding points in two images of the same scene taken from slightly different perspectives. This difference, known as the disparity, can be used to calculate the depth of objects in the scene.

To create a disparity map, the two images of the same scene are first aligned so that corresponding points in the images line up. Then, the difference in positions of corresponding points is calculated and used to determine the depth of the objects in the scene. This process is typically performed using specialized algorithms which are designed to accurately and efficiently compute disparities even in complex and cluttered scenes.

The resulting disparity map is typically a grayscale image where the intensity of each pixel represents the magnitude of the disparity at that location. This magnitude can be represented in a color bar, in this way, the disparity map provides a compact and intuitive representation of the depth information in the scene.

In this projects the steps done to perform the disparity map computation and fill 3D poincloud representation were the following:

1. Image rectification
2. Disparity computation
3. Pointcloud computation

3.1.1 Image rectification

Image rectification is an important preprocessing step for disparity map computation. It is a technique that aligns the images from a stereo pair so that corresponding points are on the same horizontal scanline. This is necessary because the stereo matching algorithm that computes the disparity map relies on the assumption that corresponding points are horizontally aligned.

Without rectification, the stereo matching algorithm would not be able to accurately determine the disparities between corresponding points in the stereo images, which would result in an inaccurate disparity map. In

other words, image rectification is necessary to ensure that the disparities computed by the stereo matching algorithm are correct and can be used to accurately reconstruct the 3D structure of the scene.

Rectification also implies not only align the image pairs, but also undistort the images and removing the fish eye effect that can appear depending on the lenses that the camera is using. On figure 3.1 it can be seen how the images are not aligned and looking at the shelf on the right it can be seen slightly displaced, also the lens applies some distortion to the image, showing a fish eye effect. With the camera intrinsic parameters along with the Matlab function *rectifyStereoImages* the new rectified images are computed (3.2) and a new re-projection matrix is computed.

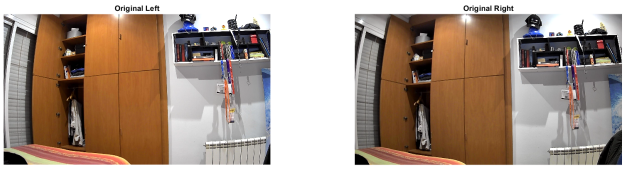


Figure 3.1: Distorted original image pairs.

Figure 3.2: Rectified image pairs.

This new re-projection matrix is needed because once the image is rectified, the size of it also changes due to some perimeter cutout and image translation, this new matrix return is of the form:

$$\begin{bmatrix} 1 & 0 & 0 & -cx \\ 0 & 1 & 0 & -cy \\ 0 & 0 & 0 & f \\ 0 & 0 & \frac{1}{b} & 0 \end{bmatrix}$$

Where f and [cx,cy] are the focal length and principal point of rectified camera 1, respectively. b is the baseline of the virtual rectified stereo camera.

3.1.2 Disparity computation

In the provided Matlab code, the left and right frames of a stereo image to grayscale using the rgb2gray function. This is done because the disparity calculation only requires intensity information and not color information. The next two lines apply histogram equalization to the grayscale left and right frames using the histeq function. This is done to enhance the contrast in the images, which can improve the accuracy of the disparity calculation. The disparitySGM function is then used to calculate the disparity map from the enhanced left and right frames. This function implements the semi-global matching (SGM) algorithm, which is a popular method for calculating disparity maps.

Finally, the disparity map is thresholded to remove disparities that are less than 20 or greater than 80. This is done to remove noise and outliers in the disparity map that may be caused by occlusions or reflections in the scene. The thresholded disparity map is then returned by the code.

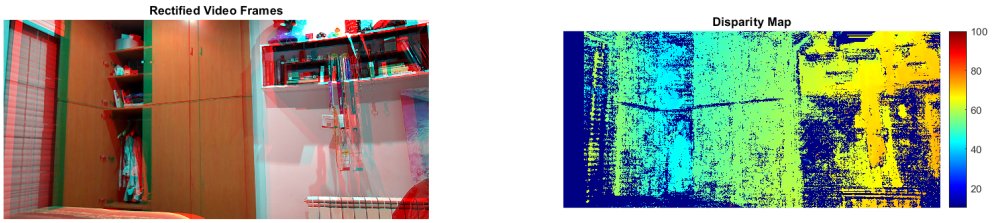


Figure 3.3: Disparity map.

3.1.3 Pointcloud computation

To compute the 3D poincloud the reconstructScene function uses the disparity map and the reprojection matrix to calculate the 3D coordinates of the points in the stereo image.The resulting 3D points are then converted from millimeters to meters by dividing by 1000. This is done for convenience, as it is more common to express 3D coordinates in meters rather than millimeters. Finally, the pcdenoise function is applied to the point cloud to remove noise and outliers. This can improve the overall quality and accuracy of the point cloud. The denoised point cloud is then returned by the code.

Figure 3.4: Colored poincloud from disparity map.

3.2 Feature triangulation

The feature triangulation method uses similar methods as the ones described on the previous one, such as image rectification, in order to avoid distortion on the images. This method extract features from both images and matches them using a matching algorithm. Then with the same features detected on both images the point is triangulated using the extrinsic correlation between cameras and their intrinsic parameters. To find the best descriptors for this method a study has been made applying the state of the art computer vision

descriptors as seen on image 3.5

On the study we observed that the ORB (Oriented FAST and Rotated BRIEF) and SURF (Speeded Up Robust Features) descriptors were the ones with more keypoints. One of the main differences between SURF and ORB is the type of local feature descriptor used by each algorithm. SURF uses a scale-invariant feature transform (SIFT) descriptor, which is a type of descriptor that is resistant to changes in scale and orientation. ORB, on the other hand, uses a BRIEF (Binary Robust Independent Elementary Features) descriptor, which is a type of binary descriptor that is designed to be faster and more efficient than other types of descriptors. As in our application the images have no scale differences as the cameras are one next to the other, the ORB descriptor was selected, but SURF also can work, giving more robust results but more slowly. On figure 3.5 can be seen the figures with the ORB descriptors and the fast matching.

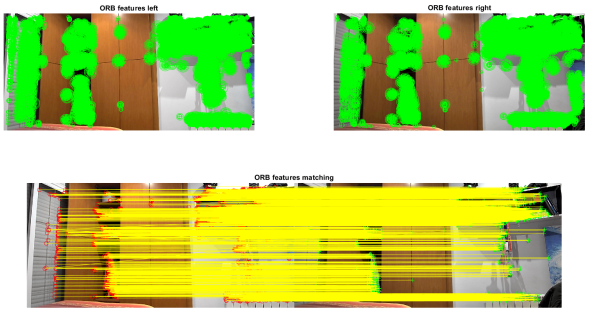


Figure 3.5: ORB descriptors and matching.

Once all the points are matched, it can be triangulated with the Matlab *triangulate* function returning the poincloud of the triangulated features ()figure 3.6.

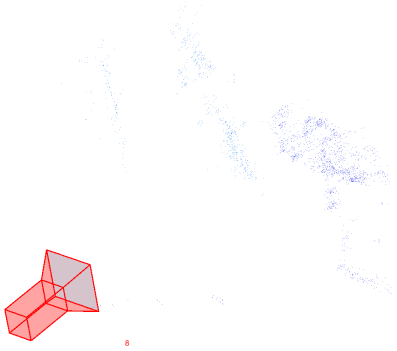


Figure 3.6: Triangulated poincloud.

As it can be seen the poincloud is not as dense as on the disparity map because this method only takes into account the most relevant features against the disparity map that computes the disparity of all the pixels.

3.3 Structure from motion

Structure from motion (SfM) is a technique used in computer vision to estimate the 3D structure of a scene from a set of 2D images. This is done by capturing a series of images of the scene from different viewpoints and then using the motion of the camera and the correspondences between the images to compute the 3D structure of the scene.

The SfM algorithm operate in two main stages: feature detection and matching, and 3D reconstruction. In the feature detection and matching stage, the algorithms use local feature detectors and descriptors to find and match corresponding points in the images.

Once the corresponding points have been found, the algorithms use the camera motion and the correspondences between the points to estimate the 3D position of the cameras on the scene. Once all the camera positions are computed on the scene the 3D reconstruction is done for each frame and finally transforming the poincloud on the computed camera pose. On the figures 3.7 and 3.8 it can be seen a sequence of consecutive frames captured by the stereo camera setup and the corresponding 3D camera poses for each consecutive frame correspondingly.

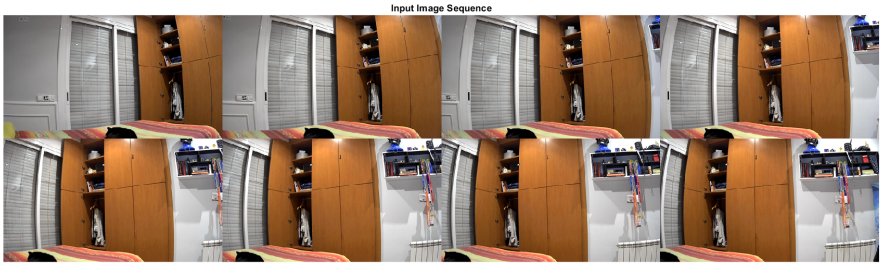


Figure 3.7: Image sequence.

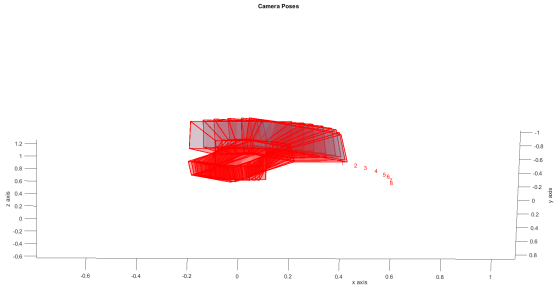


Figure 3.8: 3d camera poses from the image sequence.

Applying the explained method to the consecutive reconstructions we obtain the 3D dense poinclouds shown

on figures 3.9 and 3.10.

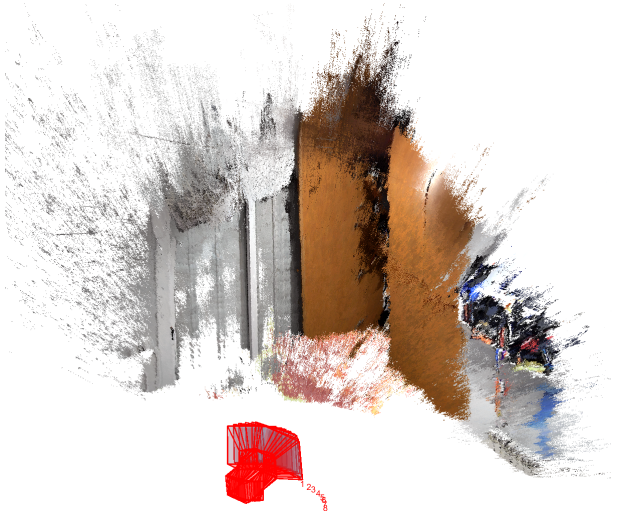


Figure 3.9: Multi-view 3D poincloud reconstruction from disparity map

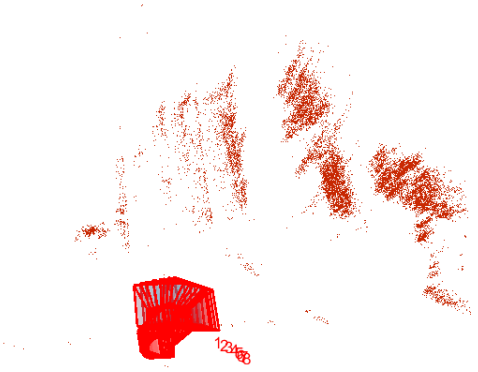


Figure 3.10: Multi-view 3D poincloud reconstruction from feature triangulation

3.3.1 3D poincloud fitting

Also as an extra step on this project some geometry fitting has been done, in this case a wall fitting has been done, using the MATLAB function *pcfitplane*, and obtaining the results on page 3.11. More precisely, the fitted plane has a normal vector of $(0.55, 0, 0.24)$ with and angular tolerance of only 3 degrees, obtaining very promising results.

The actual plane fitting is performed using the M-estimator SAmple Consensus (MSAC) algorithm to find the plane, a variant of the RANdom SAmple Consensus (RANSAC) algorithm.

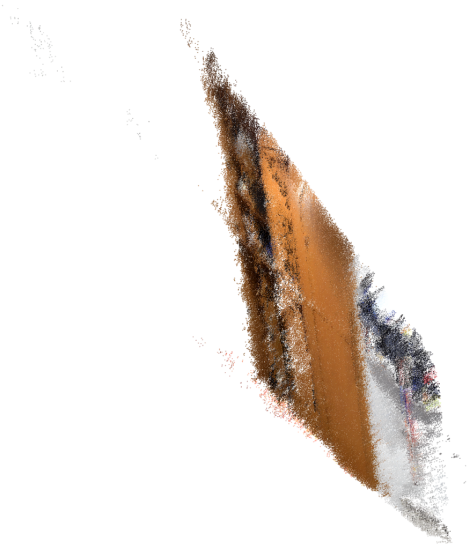


Figure 3.11: Wall fitting on 3D poincloud with MSAC algorithm

Plane reconstruction has a wide variety of applications from ground removal algorithms to reconstruct object for autonomous vehicles to hall fitting to autonomous vehicles such as drones that navigate on interiors.

4. Conclusions

As it can be seen the poincloud reconstruction from triangulate features on both images is less dense, that is because the features are triangulated on the most significant part of the images in order to reduce the error in the matching. In this case the regions with more reliable features was the window, the clothes from the closet and the books from the shelf. If the objective is to reconstruct large scenarios with a dense poincloud the disparity map reconstruction is the right choose. On the other hand if only a small object wants to be reconstructed with great precision maybe triangulate the poinclouds from different points of view could work if there are enough reliable features to track and triangulate. On the other hand photogrametry algorithms like this one does not handle reflections, this can be seen on figure 3.9, where the window's glass is not well tracked and some points on the poincloud are spread behind, this is because the features of one image does not correspond with the ones on the other one, causing problems on reconstructing reflecting surfaces. To solve this it is recommended the use of more complex state of the art tools like NeRF (Neural Radial Fields), this neural networks can make very dense and light responsive reconstructions. Also it some problems with the SfM algorithm was found causing in some cases bad pose reconstruction of the image due to bad feature tracking, causing in a bad reconstruction, it is recommended to use some external sensors to make sensor fusion and estimate the real camera pose with less error.

Overall, photogrammetry has a wide range of applications in fields such as surveying, engineering, and geospatial mapping. It is a powerful tool for creating accurate 3D models of objects and environments. (This project has been published on github and can be seen on the following github page: Github page of the code)