Stereo Vision 3D Object Reconstruction

Mirko Del Moro
mirko.delmoro@studio.unibo.it

***Abstract*—3D object reconstruction using stereo vision is a cutting-edge technology that allows the creation of realistic 3D models that can be used in product design, marketing materials, and more. The goal of this work is to obtain accurate 3D models using computer vision approaches. This system involves the use of two cameras to capture multiple images of an object from different viewpoints, then depth information is extracted from the images and used to reconstruct a 3D model of the object.**

I. Introduction

In this work, a stereo vision setup is built by using two iPhones which are held in place by a handcrafted wooden structure. This setup allows capturing a left and a right image of an object at the same moment and by rotating it, multiple couples of images from different viewpoints are obtained. In order to use the images captured from the two cameras, it is necessary to first perform stereo calibration and then rectify each couple of images. This allows to extraction of depth information by analyzing differences between the left and right images for each viewpoint. After computing the depth map, several background subtraction techniques are tried to isolate the pixels of the object with respect to the background pixels. The best resulting map is used to generate a corresponding 3D point cloud.
Finally, two point clouds, representing the object from successive viewpoints, are registered together by using Fast Global Registration [1] and the Point-To-Point ICP (Iterative Closest Point) algorithm [2].

II. Stereo Vision setup and Camera Calibration

In recent years, smartphones have become increasingly powerful and capable of performing complex tasks, including computer vision. One application of this technology is stereo vision. More specifically, I have built a stereo vision setup by using two iPhone 6 cameras as stereo vision cameras. Each smartphone has an 8-megapixel camera with 1.5μ pixels. However, there were some limitations due to this choice: the quality of the images captured by the phones was lower than what could be achieved with a dedicated camera, and they may limit the accuracy of the resulting models. Furthermore, was not easy to build a setup with two smartphones, which replicate a professional stereo system. I have tried to counteract this issue by using a specific wooden structure, which allows me to align the smartphones and held them in place. Figure 1 shows the stereo vision setup.

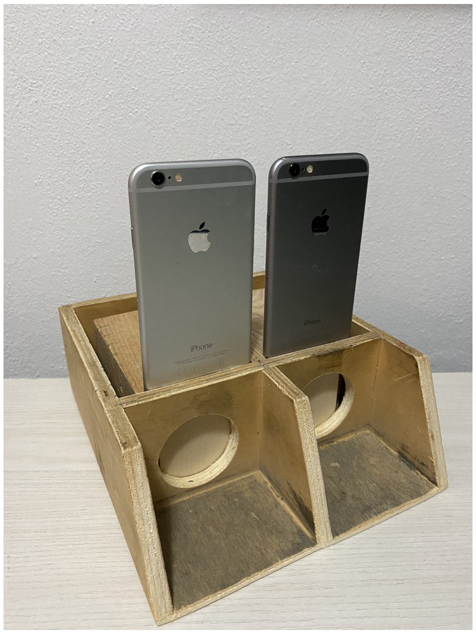
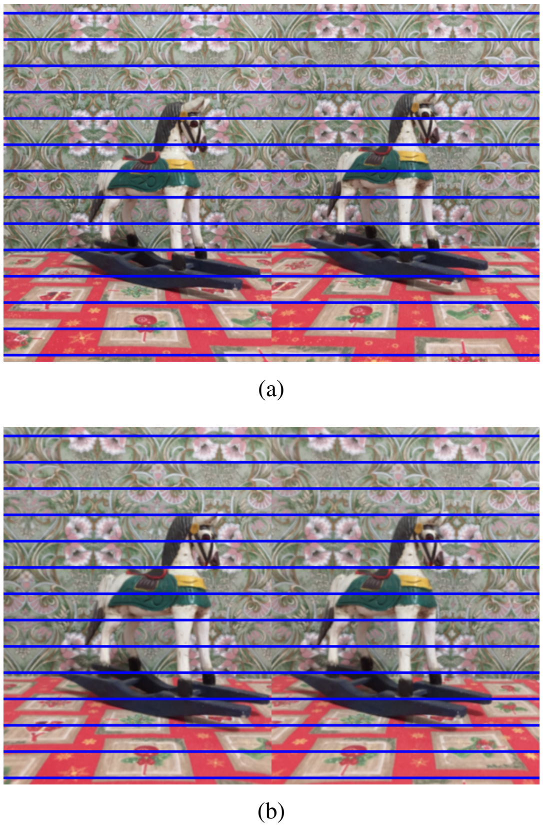


Fig. 1: Stereo Vision setup

Stereo calibration is the process of estimating all the parameters defining the specific stereo vision system. Firstly, I have performed camera calibration on each of the two cameras separately by following the method introduced by Zhang [3]. To do the calibration process I have captured 20 couple of images of a chessboard from different angles and with several rotations of the planar calibration pattern. Due to the coarseness of the stereo vision system build, I have decided to downsample images and work with images having a size equal to 306×408, which is $\frac{1}{8}$ of the original size. First of all, I have detected the corners of the chessboard images, I have refined the resulting 2D positions of corners, and I have found the correspondences between the 2D corner position in the image and 3D world coordinates. Finally, I have estimated all the parameters of each camera. I have obtained a reprojection error in pixel equal to 0.082 and 0.083 for the left and right camera calibration respectively.
Once each camera has been calibrated and its intrinsic parameters have been estimated, I have found the rigid motion between the left and right smartphone's camera. To do that, a stereo calibration process has been performed on the stereo vision system. For what regards the intrinsic parameters, I have decided to pass to the stereo calibration function the intrinsic found previously so that it can optimize them. The extrinsic parameters are initialized to the median value of the pattern views. From the stereo calibration process, I have obtained a reprojection error in pixels equal to 0.088.



(a)

(b)

Fig. 2: An example of left and right image (a) before and (b) after rectification process.

III. Stereo Rectification

The stereo calibration process provides the intrinsic parameters of the stereo system as well as the roto-translation between the two cameras. To simplify the stereo search of corresponding points in the left and right images it is more convenient to have a standard stereo geometry. To have it, I have computed the rectification transformations, which are calculated based on the parameters obtained from stereo calibration. More specifically, I have first defined two new cameras by virtually rotating the original ones, so that they are related one to another only by a translation of the x-axis and they have the same intrinsic parameters. Once the rectification transformations are computed, it is possible to undistort and rectify each couple of images acquired by the smartphone's cameras by applying these transformations to the original images. The left and right have been warped as if they were captured through a standard stereo geometry and their epipolar lines become horizontal and collinear. Figure 2 shows a couple of images (left and right) of the object from a specific viewpoint before and after the rectification process. As we can see, after the images have been rectified, corresponding points lie at the same height in the left and right images.

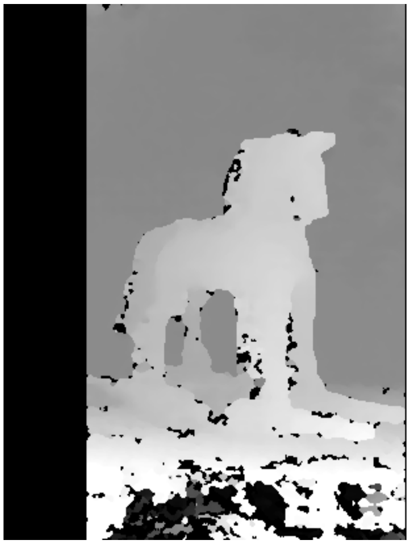


Fig. 3: Disparity map with SGBM stereo algorithm

IV. Depth Information Extraction

Rectification is a necessary process to transform an arbitrary stereo setup into a standard one to simplify the stereo correspondence problem. With rectification transformations, given a point in one image, the search for the corresponding point in the other image occurs along a horizontal line placed at the same height as the given point. Hence, once the left and right images have been rectified, a disparity map has been computed. The disparity map represents the horizontal displacement of two pixels representing the same image point in the two images. To compute disparity I have compared two different approaches. The first one is based on a traditional stereo correspondence algorithm. In particular, I have used the SGBM algorithm [4], which computes the disparity map by matching small image patches between the left and right images. Figure 3 shows the disparity map obtained by using the SGBM stereo algorithm on the rectified couple of images shown previously. The second approach I have used to compute the disparity map is based on deep learning. More specifically, I have used the Raft-Stereo algorithm, introduced by Lipson et al. [5], which is a deep learning-based stereo algorithm that uses a convolutional neural network (CNN) to extract features from the stereo images and a 3DCNN to aggregate the feature-matching costs into a cost volume. In particular, I have used the model which was pretrained on the Sceneflow dataset [6] and fine-tuned on the 2014 Middlebury stereo dataset [7]. Figure 4 shows the disparity map obtained by using the Raft-Stereo algorithm on the same rectified stereo images. As we can see from the above figures, compared to SGBM, Raft-Stereo achieves better accuracy and robustness in computing the disparity map. Indeed, while the first disparity map is affected by noise, especially on the object edges and on the floor, in the second one, noise is almost absent and the silhouette of the object is more accurate.

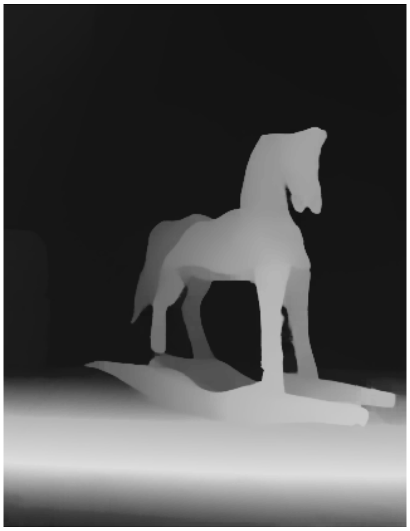


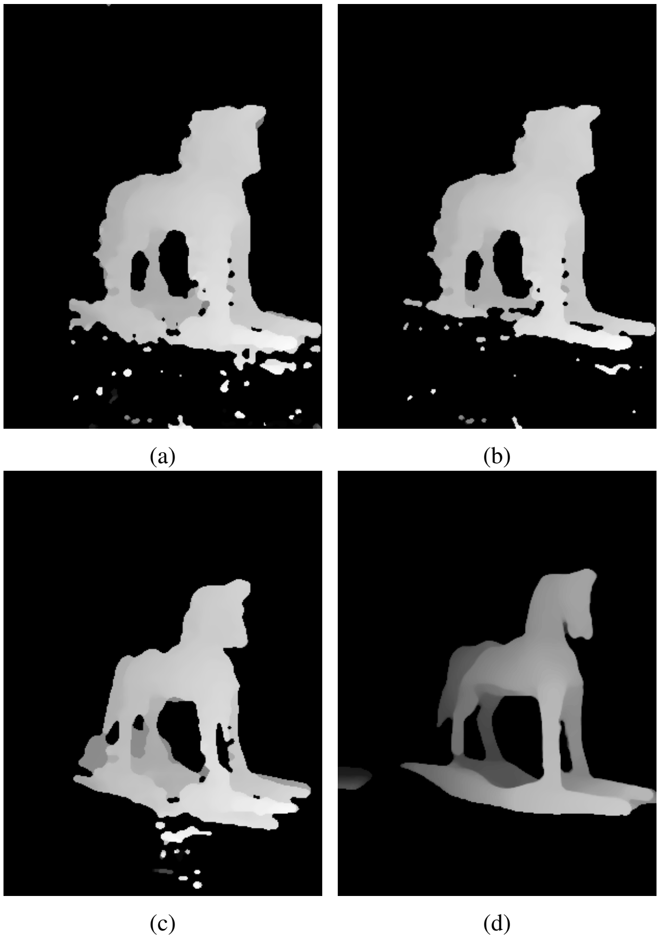
Fig. 4: Disparity map with Raft-Stereo algorithm

V. Disparity map refinement

The goal of this work is to reconstruct the 3D model of the object as accurately as possible. Hence, I have decided to try to mask all the background pixels in the disparity map, to isolate as better as possible the points which belong to the object. To do that, I have applied different techniques:

- in the first approach I have considered a couple of images (left and right) representing the background without the object, then I have computed the corresponding disparity map and finally I have obtained the refined disparity map by computing the absolute difference between the disparity map image representing the object and the one representing only the background. Figure 5a shows the refined disparity map by using this approach.
- the second one is based on the usage of MOG2 background subtractor [8], which is an algorithm used for foreground detection based on the mixture of Gaussian models. More specifically, I have used the MOG2 algorithm to compute the foreground mask. I have passed to the algorithm the disparity map representing the object and the background one, then I have applied the output mask to the disparity map representing the object. Figure 5b shows the results by using this approach.
- the latter approach is similar to the second one but changes what I have passed to the background subtractor algorithm. Indeed, i have first considered 5 left images representing the object from the same viewpoint and 5 left images representing just the background, then i have computed the mean for the object images and for the background ones and i have passed the two resulting mean images to the algorithm. The purpose of computing the mean of 5 images is to make the resulting image more robust with respect to light conditions. Figure 5c shows the resulting disparity map.

All the resulting disparity maps have been further refined by applying a median filter with a 5x5 aperture twice. To perform the different background subtraction techniques, just the disparity maps obtained with the SGBM algorithm have been considered. Then I have compared them and applied the best technique to the disparity map obtained with the deep learning approach. The third approach seems to be the best in terms of the resulting silhouette of the object and noise. Indeed, the results of the first two approaches are very similar and more coarse about the object's shape. Hence, I have decided to use the third approach to compute the refined disparity maps and I have applied this technique to the deep learning disparity map. Figure 5d shows the result. Figure 5 shows a summary of all the background subtraction methods used and the resulting disparity maps, also for the case of deep learning stereo algorithm.



(a) (b) (c) (d)

Fig. 5: Examples of refined disparity maps obtained with different background subtraction techniques and after median filter smoothing.

VI. Point cloud

To reconstruct the 3D model of the object, it is necessary to estimate exactly the 3D coordinates of each pixel. The representation of a set of points in 3D space is called Point Cloud and it can be generated from a depth map, which

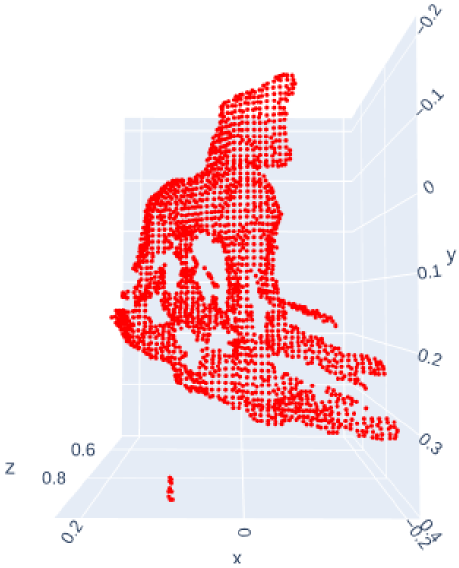


Fig. 6: Example of point cloud generated from the best result obtained with SGBM algorithm

represents the distance between the camera and each point in the scene. Therefore, it is necessary to know the intrinsic parameters of the camera to get the 3D coordinates of each pixel. To generate the point cloud I have used *Open3D* library and the intrinsic parameters obtained with the rectification process. I have considered the best result obtained with the SGBM algorithm and the deep learning algorithm result, Figure 5c and Figure 5d respectively, to generate two point clouds relative to the same scene. However, since the point cloud corresponding to the Raft-Stereo algorithm shows up a trail of points from the object to the background, I have decided to consider just the point cloud generated from the best result achieved with the SGBM algorithm for the successive steps. Figure 6 shows an example of a point cloud generated from the result obtained with the SGBM algorithm and the third approach of background subtraction.

VII. Point Cloud Registration

Point cloud registration is the process of aligning two or more point clouds in a common coordinate system. The goal of registration is to find the transformation that best maps one point cloud onto another while minimizing the distance between corresponding points.

As we can see from the figure above, the point cloud generated shows up a bit of noise, more specifically there are some points that do not belong to the object such as the points on the lower left or the ones behind the rocking horse. So I have decided to first refine the point cloud by removing all the points that have neighbors less of a specific number of points in a sphere of a given radius. To perform registration I have decided to use the ICP algorithm and more specifically its point-to-point variation. ICP is a popular algorithm for rigid point cloud registration which iteratively finds the optimal rigid transformation that aligns two point clouds. More specifically it updates the transformation T by minimizing an objective function, which, in the point-to-point variant, is the following:

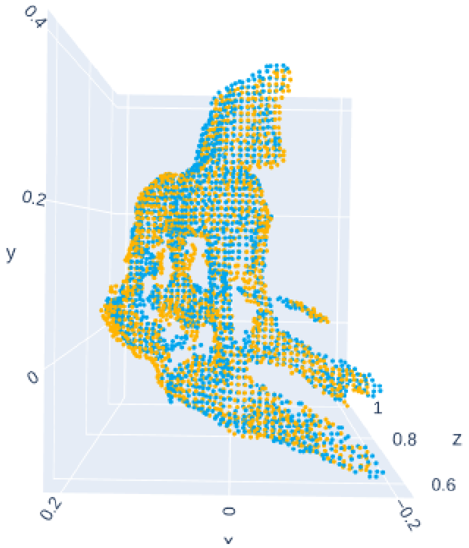


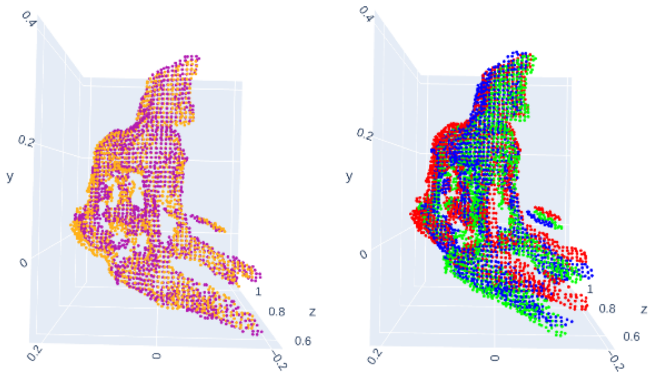
Fig. 7: Registration of two successive point clouds

$$E(T) = \sum_{(p,q)\in K} \|p - Tq\|^2 \tag{1}$$

where $K = \{(p, q)\}$ is the set of correspondences from target point cloud P and source point cloud Q transformed with the current transformation matrix T. The ICP algorithm is a local registration method, which means that it relies on a rough alignment as initialization. So if the initial transformation is far from the true solution, the ICP algorithm can get stuck in local minima. To mitigate this, I have decided to use a global registration algorithm, which does not require alignment for initialization. It produces less tight alignment results but they are used as initialization for the Point-To-Point ICP method. In particular, I have used the fast global registration proposed by Zhou et al., which is based on a probabilistic model that estimates the transformation parameters by maximizing the likelihood of correspondences between the two point clouds. Figure 7 shows the result after the registration of two successive point clouds. As can be seen, the registration has aligned the two point clouds to produce a more complete and accurate representation of the object. However, the resulting point cloud shows up a bit of noise, specifically where the noise of the two point clouds accumulates, for instance at the pedestal. I have used Open3D implementation of Fast Global Registration and ICP algorithm and I have set the maximum number of iterations of the latter to 2000 so that it runs until convergence or reaches this number of iterations. With Point-To-Point ICP refinement the fitness score, which measures the overlapping area, increases from 0.112622 to 0.7564563 and the inlier_rmse, which measures the Root Mean Square Error on inlier correspondences, reduces from 0.003702 to 0.005024 with respect to the values obtained with global registration.

VIII. Results

For simplicity, due to the noise that builds up as new point clouds are registered, I have decided to register just three successive point clouds and not completely reconstruct the object at 360 degrees. Figure 8 shows an example of results obtained by registering three successive point clouds by using two different approaches. In the first one, I have first registered the first and second point clouds and then I have concatenated them to obtain a new point cloud. Finally, I have registered the new point cloud with the third to obtain the result. In the second approach, I have registered the first and second point clouds, the second and the third, and displayed all the three point clouds with respective transformations. As we can see from Figure 8a, the first approach seems to be better in terms of registration transformation and accumulated noise; indeed in Figure 8b there are some points, especially in the pedestal, which are not perfectly registered with respect to the three different point clouds.



(a)

(b)

Fig. 8: Different registration approaches of three successive point clouds.

IX. Conclusion

Based on the research and analysis conducted in this report, it can be concluded that despite the usage of a non-professional stereo vision setup, it is still possible to create detailed and accurate 3D models of objects by utilizing the principles of triangulation. Certainly, the accuracy of the reconstructed model is highly dependent on the quality of the images captured by the cameras, as well as their calibration. Therefore, to improve the results with this stereo setup, some modifications could be adopted. For instance, I could follow the path of deep learning methods to create the disparity map and then the point cloud to get a more detailed and accurate silhouette of the object. Furthermore, I could try more accurate techniques of background subtraction and try to reduce the noise problem by, for instance, removing the pedestal points of the rocking horse in the 3D model.

References

[1] Q.-Y. Zhou, J. Park, and V. Koltun, "Fast global registration," vol. 9906, 10 2016.

[2] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, 2001, pp. 145–152.

[3] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, 2000.

[4] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, 2008.

[5] L. Lipson, Z. Teed, and J. Deng, "Raft-stereo: Multilevel recurrent field transforms for stereo matching," *CoRR*, vol. abs/2109.07547, 2021.

[6] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," *CoRR*, vol. abs/1512.02134, 2015.

[7] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nesic, X. Wang, and P. Westling, "High-resolution stereo datasets with subpixel-accurate ground truth," in *GCPR*, ser. Lecture Notes in Computer Science, vol. 8753. Springer, 2014, pp. 31–42.

[8] Z. Zivkovic and F. van der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognit. Lett.*, vol. 27, no. 7, pp. 773–780, 2006.