Panorama Stitching and Point Cloud Registration

Davi G. Valério
ist1108497
Instituto Superior Técnico
Lisboa, Portugal

Francisco R. Nogueira
ist1108469
Instituto Superior Técnico
Lisboa, Portugal

Eduardo Guerra
ist1102681
Instituto Superior Técnico
Lisboa, Portugal

Ygor Acacio Maria
ist1112401
Instituto Superior Técnico
Lisboa, Portugal

1 INTRODUCTION

This project has two parts. Part 1 of this project aims to map multiple frames extracted from one or multiple videos to a single reference image. We estimate homographies using SIFT keypoints and RANSAC. Instead of sequentially composing transformations and accumulating errors, we propose a graph-based approach to lower the number of compositions and reduce errors.

Part 2 focuses on reconstructing a 3D scene. Each frame is processed for depth estimation, feature extraction, and point cloud generation. We align clouds of keypoint matches using Procrustes and ICP, but errors from depth estimates and alignment steps led to degraded results. Finally, we compare them with Meta's Visual Geometry Grounded Deep Structure From Motion (VGGSfM).

The remainder of this report is organized as follows: Section 2 provides a brief overview of SIFT, homographies, RANSAC, Procrustes, and ICP. Section 3 details the Part1 methodology and explains our proposed graph-based homography approach. Section4 covers the pipeline for 3D reconstruction, and a comparison to VGGSfM. Finally, Section 5 outlines possible future work to improve results.

2 TECHNICAL BACKGROUND

2.1 Scale Invariant Feature Transform (SIFT)

The Scale Invariant Feature Transform (SIFT) [Lowe 1999] is a computer vision algorithm used to detect and describe local features in images. It is designed to be invariant to scale and rotation, and partially invariant to affine transformations. Each feature consists of a keypoint and a descriptor: the keypoint indicates the feature's location, and the descriptor is a 128-dimensional vector used for matching. For example, to find a match of keypoints between two images, one can measure the Euclidean distance between descriptors and choose the pair with the smallest distance. Next, if this distance is below a threshold or significantly smaller than the second-best match, the keypoints are considered a match.

2.2 Homography

A Homography is a projective transformation that maps points from one plane to another. It is useful to relate two images of the same scene taken from different perspectives. A homography transforms a point $\mathbf{x}_i$ in the image plane to a new point $\mathbf{x}_i'$ through a 3×3 matrix H, where both points are in homogeneous coordinates. H can be computed from a set of corresponding points between two images, and includes translation, rotation, scaling, and perspective distortion. Given four or more point correspondences $\{x_i, x_i'\}$ the transformation can be solved by minimizing the error between the projected points and their correspondences: $\sum_{i=1}^{N} ||Hx_i - x_i'||^2$. This is typically achieved using the Direct Linear Transformation (DLT) algorithm, which is derived from the correspondences is linear in the elements of H and can be expressed as $Ah = 0$, where A is a $2N \times 9$ matrix and h has the elements of H. Once the homography is computed, it can be used to warp one image to align with another.

2.3 RANSAC

Random Sample Consensus (RANSAC) [Fischler and Bolles 1981] is a method to estimate parameters of a mathematical model from a dataset that contains outliers. It is commonly applied in computer vision tasks such as homography estimation, 3D reconstruction, and point cloud alignment, where data points may be noisy or mismatched. RANSAC iteratively selects random data subsets, fits a model, and evaluates its consistency with the full dataset to maximize the number of inliers within a tolerance.

Assume a homography estimation task where points x_i in an image A correspond to x_i' points in image B. The homography matrix relates these points via $x_i' = H \cdot x_i$, with x in homogeneous coordinates.

The RANSAC error function for homography is $d = ||x_i' - H \cdot x_i||^2$. If d is less than a specified threshold ϵ, then the point (x_i, x_i') is considered an inlier.

The objective of RANSAC is to maximize the number of inliers by finding the homography matrix that minimizes the overall error across all point pairs.

The number of iterations N required to guarantee a valid model with a probability P depends on the proportion of inliers w in the dataset and the minimum number of points s needed to fit the model. With relation to homography, for instance, s should be at least 4, the least amount required to calculate a homography matrix.

2.4 Procrustes

The Procrustes analysis is a mathematical method used to align two sets of points in 2D or 3D space. It is often applied in computer vision tasks to find the best rigid transformation, involving translation and rotation, and sometimes scaling, that minimizes the distance between two point clouds. The mathematical goal of Procrustes analysis is to minimize the error function, which is defined as the sum of squared distances between the corresponding points in the two sets. This error is expressed as:

$$E_{\text{R3D}} = \sum_i \|\mathbf{x}'_i - R\mathbf{x}_i - \mathbf{t}\|^2, \quad (1)$$

where R represents the rotation matrix and $\mathbf{t}$ is the translation vector.

The process begins with an estimate of the translation. This is done by calculating the centroids of both sets of points, denoted as $\mathbf{c}$ for the first set and $\mathbf{c}'$ for the second. The translation vector is then computed as $\mathbf{t} = \mathbf{c}' - \mathbf{c}$.

Once the translation is determined, both sets of points are centered at the origin by subtracting their respective centroids. After centering, the rotation matrix R is estimated using Singular Value Decomposition (SVD). To do this, a correlation matrix C is constructed as $C = \sum_i \mathbf{x}'_i \mathbf{x}_i^T$ and the SVD of C is then performed, resulting $C = U\Sigma V^T$. Next, the rotation matrix is obtained as $R = UV^T$. A final check ensures that R is a valid rotation matrix by verifying that its determinant equals +1; adjustments are made if necessary.

2.5 Iterative Closest Point (ICP)

Iterative Closest Point (ICP) is a method for aligning a *source* set of 3D points to a *target* set. Suppose we have source points $\{\mathbf{x}_i\}$ and an initial Procrustes transformation given by rotation R_0 and translation $\mathbf{t}_0$. Each source point is initially transformed as

$$\mathbf{x}_i^0 = R_0\,\mathbf{x}_i + \mathbf{t}_0.$$

The algorithm then finds correspondences in the target set, typically by nearest-neighbor matching:

$$\mathbf{x}_i'^0 = \text{NN}(\mathbf{x}_i^0, \{\mathbf{x}'_j\}),$$

where $\mathbf{x}'_j$ are the points in the target set. Once these correspondences are established, an updated Procrustes-like transformation $R_1, \mathbf{t}_1$ is computed via Singular Value Decomposition (SVD) to minimize:

$$E_{\text{ICP}} = \sum_i \|\mathbf{x}_i'^0 - R_1\mathbf{x}_i^0 - \mathbf{t}_1\|^2.$$

The source points are then re-transformed using $R_1, \mathbf{t}_1$, and the process repeats until convergence. By iteratively refining the transformation and correspondences, ICP can accurately register the source and target point sets, although it relies on a good initial guess to avoid local minima.

3 PART 1

3.1 Methodology

To get the transformations from each images from a video from a moving camera to a reference image, we propose the following pipeline.

(1) (Optional) Recalculate *SIFT* features for all images and the reference.
(2) (Optional) Resize images to reduce computational load.
(3) Compute the maximum descriptor distance threshold δ.
(4) Compute the minimum inliers threshold β.
(5) Find direct homographies to reference image.
(6) Find indirect homographies for the remaining images.

The goal of this part of the project is to transform a set of images extracted from a video to the coordinates of a reference image. Let x_i from 0 to N be the frame index i of a video with N frames. Define r one of the frames in this video as the reference frame. The homography between x_i and x_j will be denoted $H_{i,j}$. In this work, $H_{i,j}$ was estimated using SIFT keypoint matches.

The simple solution to this problem is to find the index of the reference frame and compose sequential transformations directly. For example, if $r = x_3$, then the homography from x_5 to r can be calculated as $H_{5,3} = H_{5,4} \cdot H_{4,3}$. However, for long sequences, the number of composition would be a major source of errors.

Instead, we would like an algorithm to find the homography for each image with low number of compositions. For this, we will treat each image as a node in a graph and aim at finding the shortest valid path from each node to the reference node, where each edge represents a valid homography. A valid homography is a homography with a number of inliers higher than β, where β is a hyperparameter and the inliers are found with the RANSAC algorithm.

One useful heuristic for β comes from the assumption that the set of images is ordered and comes from a video with continuous movement. When this is the case, it is likely that there are good matches of keypoints between each pair of images (x_i, x_{i+1}). Thus, to set β, one could calculate the homography between each sequential pair (x_i, x_{i+1}) and define β as the minimum number of inliers between these homographies.

Another relevant hyperparameter in the algorithm is δ: the maximum descriptor distance. This is defined to ensure only high quality matches are used in the homography. In our experiments, we computed the 90% percentile distance between the descriptors for all pairwise sequential matches (x_i, x_{i+1}) and set δ to the maximum among these matches. This value was then used to filter each SIFT keypoint match by the distance between its descriptors.

With δ, β and valid homographies defined, the algorithm consists of two steps. First, find the direct homographies for each frame to the reference image. Note that matches were filtered by δ and that a homography is only valid if the number of inliers is higher than β. Figure 1 illustrates a scenario where x_1 and x_2 have valid homographies to the reference.

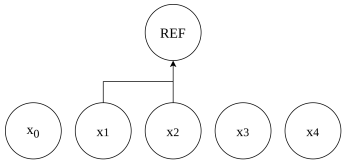

Figure 1: Illustration of direct homographies.

Next, the second step consists of finding the indirect homographies. For this, iterate over the remaining nodes and verify if there is a valid homography between it and the nearest connected node. In the example, suppose there exists the valid homographies $H_{0,1}$, $H_{3,2}$ and $H_{4,2}$. As a result, the path from x_4 avoids x_3, as it can be seen in Figure 2.

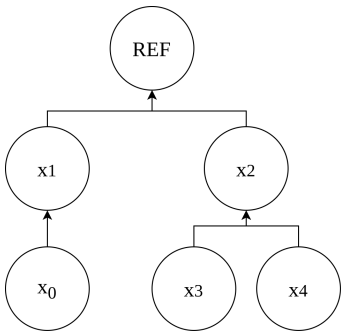

Figure 2: Illustration of indirect homographies.

3.2 Experiments and results

In this section, we present the experiments conducted to evaluate the proposed pipeline across three different datasets: *Isr_wall*, *Power Plant* and *Volley*, all provided by the faculty. The performance was assessed based on computation time, maximum descriptor distance threshold, minimum number of inliers, and the resulting transformed images.

For the *Isr_wall* dataset, the total computation time was 26 seconds, achieving a maximum descriptor distance of 0.2609 and a minimum inliers count of 544. The homography graph created from this dataset (Figure 3) showed a mixture of direct and indirect transformations, indicating robust performance in aligning consecutive images as can be seen in image 4.

For the *Power Plant* and *Volley* datasets, we obtained the stitched images shown in Figures 5 and 6, respectively. Initially, the transformations generated using the original SIFT descriptors, which consisted of 64-element vectors, exhibited a substantial amount of error. To address this, we decided to recalculate the SIFT features using 128-element descriptors. This recalculation significantly improved the results. For the *Power Plant* dataset, the processing took 22 minutes and 18.8 seconds, with a maximum descriptor distance of 284.1335 and a minimum number of inliers of 644. Similarly, for the *Volley* dataset, the processing took 32 minutes and 19.5 seconds, resulting in a maximum descriptor distance of 299.0066 and a minimum number of inliers of 770.

It is important to take note that the proposed algorithIm works best if the images come from a sequence of frames with continuous motion. That is, if there is a sudden change in a video or if two distinct videos are put in sequence, the algorithm may not yield good results. This is because there will exist a pair of frames x_i, x_{i+1} that does not have good matches. Despite this, the heuristic to determine β will take this pair into account, which will lead to transformations with low quality.

4 PART 2

4.1 Methodology

To reconstruct a 3D scene from 2D images, we propose the following pipeline.

(1) Estimate monocular depth with *depth-pro* [Bochkovskii et al. 2024].
(2) Extract feature keypoints using *SIFT*.
(3) Un-project keypoints from 2D to 3D using the depth estimate.
(4) Match keypoints from each pair of images by descriptor distance.
(5) Estimate the *Procrustes* rigid-body transformation between each pair of keypoint clouds using *RANSAC* for outlier removal.
(6) Refine the rigid-body transformation of each pair of inlier keypoint clouds with *ICP*.
(7) Incrementally merge point clouds through transformations that greedily minimizes the error of transforming inlier keypoints.

Note that this pipeline is computationally demanding, since it calculates the transformation for each pair of images in the dataset. This was done to allow for datasets without clear ordering between the images. The result of this pipeline was compared to Meta's Visual Geometry Grounded Deep Structure From Motion model (*VGGSfM*) [Wang et al. 2024].

4.2 Experiments and results

To test the pipeline, two datasets were used. One is named *Office* and constitutes of images of a table with objects with background elements. The other is named *Car* and constitutes of the front camera of a car navigating in the street. Note that for the *Office* dataset, depth and features were given by faculty. This dataset also had a confidence map for the depth estimates, which was used to filter the keypoints. For the *Car* dataset, these elements were calculated with the described pipeline and a confidence map was built by setting a threshold to the depth estimate. Here, all depth values above 100 meters were discarded. A link with an environment to replicate the depth estimate is available at https://lightning.ai/davigiordano/studios/depth-pro-piv

We report that the pipeline is not able to reconstruct quality 3D scenes from 2D images. Figures 7 and 8 show an attempt to merge frames 0, 3, 4 and 7 to the reference frame of 3 of the *Office* dataset. Figures 9 and 10 show an attempt to merge the first and second frames for the *Car* dataset.

When running *VGGSfM* for both datasets, the output is a sparse point cloud. It was able to reconstruct the *Office* scene, as shown in Figure 11. However, the front camera of the *Car* dataset is not enough to compute a high quality point scene. Figure 12 shows that the model was not able to capture the details of the road. Interestingly, it was also not able to capture the right turn done by the driver. For the *Office* dataset, a L4 GPU with 24GB of RAM was used. For the *Car* dataset, a L40S GPU with 48GB of RAM was used. An environment where these experiments can be replicated is available at this link: https://lightning.ai/davigiordano/studios/piv-vgg-sfm.

5 FUTURE WORK

To improve results of part 1, we would plan to extend the algorithm so that it works better for videos without continuous motion and for videos from multiple cameras. To improve results of part 2, we propose estimating depth with a stereo camera, instead of a monocular estimate. In addition, modern approaches for feature extraction

and matching could be used, such as XFeat [Potje et al. 2024]. Instead of *Procrustes* and *ICP* with keypoints, modern methods with embedded feature matching could be used, such as ColorPCR [Mu et al. 2024] or BUFFER [Ao et al. 2023].

6 APPENDIX

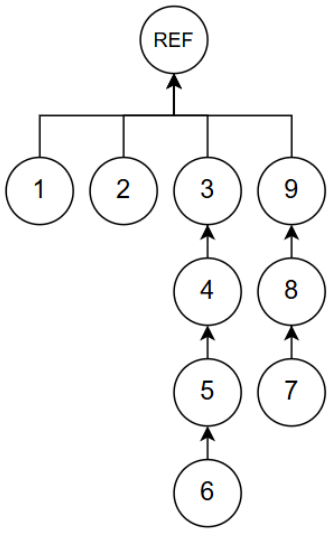

Figure 3: transformation graph of Isr_wall dataset

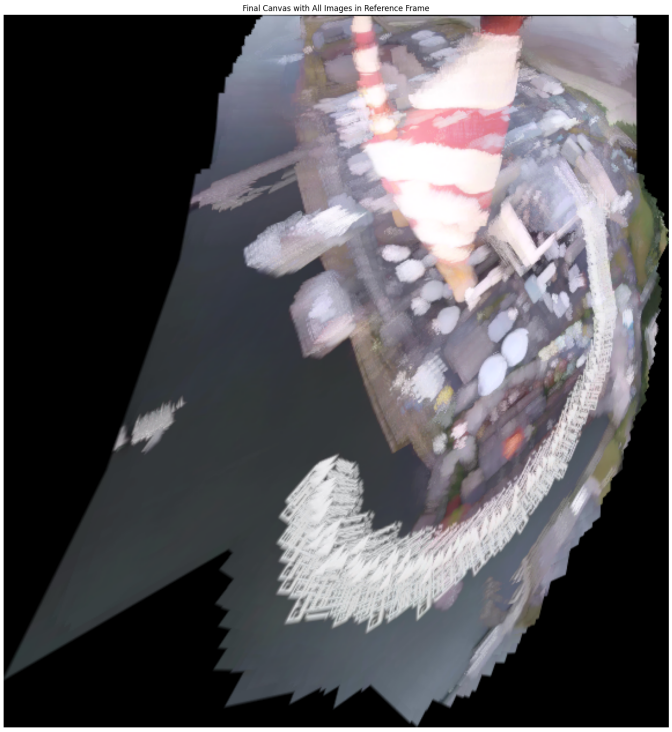

Figure 5: power plant dataset stitched images

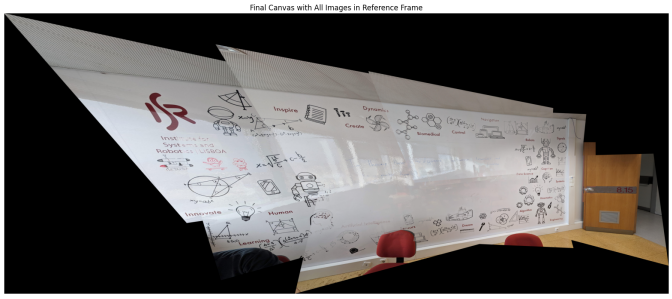

Figure 4: Isr_wall dataset stitched images

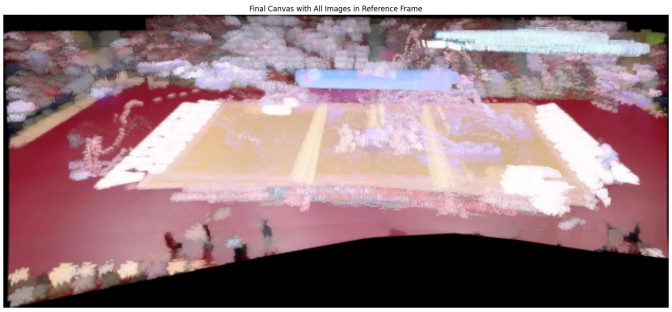

Figure 6: volley dataset stitched images

Figure 7: Front view of *Office* scene.

Figure 8: Top view of *Office* scene.

Figure 9: Front view of two frames of *Car* dataset.

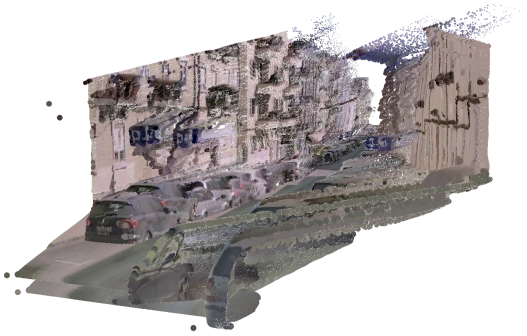

Figure 10: Side view of two frames of *Car* dataset.

Figure 11: *Office* point cloud with VGGSfM.

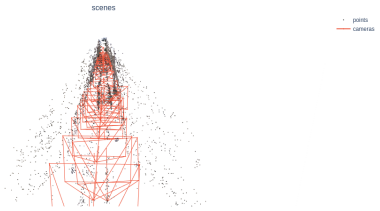

Figure 12: *Car* Point cloud with VGGSfM.

REFERENCES

Sheng Ao, Qingyong Hu, Hanyun Wang, Kai Xu, and Yulan Guo. 2023. Buffer: Balancing accuracy, efficiency, and generalizability in point cloud registration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1255–1264.

Aleksei Bochkovskii, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R. Richter, and Vladlen Koltun. 2024. Depth Pro: Sharp Monocular Metric Depth in Less Than a Second. *arXiv* (2024). https://arxiv.org/abs/2410.02073

Martin A. Fischler and Robert C. Bolles. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* 24, 6 (June 1981), 381–395. https://doi.org/10.1145/358669.358692

D.G. Lowe. 1999. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Vol. 2. 1150–1157 vol.2. https://doi.org/10.1109/ICCV.1999.790410

Juncheng Mu, Lin Bie, Shaoyi Du, and Yue Gao. 2024. ColorPCR: Color Point Cloud Registration with Multi-Stage Geometric-Color Fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 21061–21070.

Guilherme Potje, Felipe Cadar, Andre Araujo, Renato Martins, and Erickson R. Nascimento. 2024. XFeat: Accelerated Features for Lightweight Image Matching. arXiv:2404.19174 [cs.CV] https://arxiv.org/abs/2404.19174

Jianyuan Wang, Nikita Karaev, Christian Rupprecht, and David Novotny. 2024. VGGSfM: Visual Geometry Grounded Deep Structure From Motion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 21686–21697.