3D Augmented Reality A.Y. 2022-2023

Final Project: Local Features Compression Using Autoencoders

Nicoletti Gianpietro 2053042, Ege Alp Turkyener 2049576
University of Padua:
ICT for Internet and Multimedia
(Cybersystems)

Abstract

Local feature compression is a crucial problem in the field of computer vision and image processing, as it allows for the efficient storage and transmission of high-dimensional local features extracted from images. In this project, we explore the use of auto-encoders for local feature compression and evaluate their performance on various image datasets. We use COLMAP and SIFT to extract local features from the images and train auto-encoders to compress these features while preserving the important information contained in them.

Introduction

Castle P30 and P19 were used as train datasets, while Fountain P11 was used for the test. The objective of this project are:

- To explore the use of COLMAP for extracting local features from images and evaluate its performance in combination with the auto-encoder-based compression method.
- To compare the computational efficiency and storage requirements of the Structure for Motion (SfM) algorithm both before and after compression
- To utilize the autoencoder in order to decrease the number of noisy points

For this project, we selected Python as the programming language and used the pycolmap library to extract keypoints and descriptors, while the Keras library was utilized to implement the autoencoder.

Compression techniques using NN: Autoencoder

One common approach with Autoencoder is to optimize the reconstruction error or loss, which measures the difference between the original feature vectors and their reconstructed counterparts. The goal is to minimize the reconstruction error while still preserving the important information contained in the features.
To reduce the dimensionality of our descriptors, we employed an autoencoder. The mean squared error was used to evaluate the discrepancy between the input and output. The input dimension was initially set at 128, a simple sequence of linear layers, it was reduced to 8. Adam was selected as the optimizer due to its generally faster computation time and superior performance on this task. Given the relatively simple and small size of the data, we chose to utilize the basic layout with Adam optimizer.

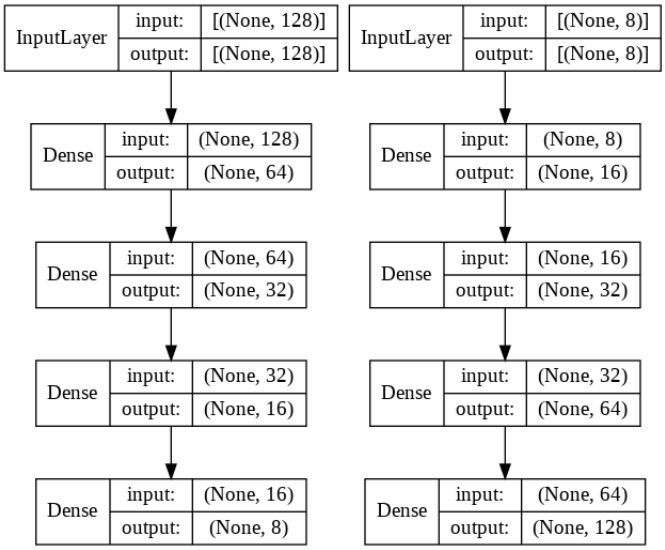

Figure 1: The scheme of the encoder on left, the decoder on right

Test setup and impact on the computational time

The extraction of the features was carried out using the PyColmap library, and the data was trained using 20 epochs. Upon completion of the 20 epochs on the training data using the MSE as loss function, the mean batch loss was found to be 0.00190. Subsequently, the test set was utilized to test the model, and the mean batch loss was determined to be 0.00189. The similarity between the training and test losses indicate that the network has effectively adapted to various datasets, suggesting that it has learned about the SIFT descriptor rather than any specific dataset.
The matching of the 3D points was performed using OpenCV and the time to compute them was saved since it were used to evaluate the impact of local features compression using autoencoders on the reconstruction time. The reconstruction time is an important factor in determining the efficiency of the reconstruction process and can influence the practicality and scalability of the reconstruction algorithm.
In the project three different version of the datasets were considered:

- **Original**: the features extracted by PyColmap;
- **Compressed**: the features compressed using the encoder network;
- **Reconstructed**: the features reconstructed by the decoder network.

Table 1: Time to compute the matching for each datasets for the different versions.

	Original	**Compressed**	**Reconstructed**
Castle-P19	184.74s	64.43s	184.70s
Castle-P30	502.17s	174.22s	494.17s
Fountain-P11	103.71s	36.37s	105.52s

Our results show that the reconstruction time for the compressed versions of the P19 and P30 data-sets are 64.43 seconds and 174.22 seconds, respectively, which are significantly lower than the reconstruction time for the original versions (184.74 seconds and 502.17 seconds, respectively). The reconstruction time for the reconstructed versions, obtained by applying the autoencoder to the compressed versions, are 184.70 seconds

and 494.17 seconds, respectively.
For the Fountain P11 dataset, the reconstruction time for the original version is 103.71 seconds, while the reconstruction time for the compressed and reconstructed versions are 36.37 seconds and 105.52 seconds, respectively.
Autoencoders for compressing local features can significantly reduce the reconstruction time for the P19 and P30 datasets, while having a minimal impact on the reconstruction time for the Fountain P11 dataset. The compression process leads to a reduction in the reconstruction time for the compressed versions, which can be advantageous for scenarios where computational resources or time are limited. However, it is worth noting that the reconstructed versions have a similar reconstruction time to the original versions, indicating that the reconstruction process has a similar computational cost regardless of whether the features have been compressed.
It's important to notice that, during the matching phase, a technique to remove the noisy point was adopted: Given a point p_i and the distance d_1 from the closest point and the distance d_2 from the second closest point, during the matching phase are keeping only the matches that has $d_1 \leq 0.75 d_2$.

Results

The time efficiency was already discussed, while the space efficiency is trivial due to the high compression rate of the autoencoder. What it's important now is to verify that these improvements in terms of time and space are supported by the accuracy of the matching. We evaluated the impact of local features compression using autoencoders on points, the mean track length, and the mean projection error of 3D points on the three different datasets.

Table 2: Comparison between the considered metrics for each dataset and each version.

		Original	**Compressed**	**Reconstructed**
Castle-P19	**Points**	10583	7182	7959
	Mean Track Lenght	3.8402	3.2277	3.3563
	Mean Reprojection Error	0.4535	0.4570	0.4738
Castle-P30	**Points**	19979	16752	17552
	Mean Track Lenght	4.8122	3.8404	4.0423
	Mean Reprojection Error	0.4767	0.4722	0.4782
Fountain-P11	**Points**	10585	5327	6452
	Mean Track Lenght	4.7864	3.8136	4.0116
	Mean Reprojection Error	0.3552	0.2583	0.2675

The number of points is a measure of the detail and density of the reconstructed scene and is an important factor in determining the computational cost of the reconstruction process. Our results show that the number of points in the compressed versions of the P19 and P30 datasets are 7182 and 16752, respectively, which are lower than the number of points in the original versions (10583 and 19979, respectively). The number of points in the reconstructed versions, obtained by applying the autoencoder to the compressed versions, are 7959 and 17552, respectively. For the Fountain P11 test dataset, the number of points in the original version is 10585, while the number of points in the compressed and reconstructed versions are 5327 and 6452, respectively. The compression process leads to a reduction in the number of points in the compressed versions of the datasets, but the reconstruction process is able to recover a significant portion of

the original scenes in the reconstructed versions.
The mean track length is a measure of the number of images that contribute to the estimation of the 3D position of a point and is used as a proxy for the reliability of the 3D point estimate. Our results show that the mean track length for the compressed versions of the P19 and P30 datasets are 3.2277 and 3.8404, respectively, which are lower than the mean track length of 3.8402 and 4.8122 for the original versions, respectively. The mean track length for the reconstructed versions, obtained by applying the autoencoder to the compressed versions, are 3.3563 and 4.0423, respectively. For the Fountain P11 test dataset, the mean track length for the original version is 4.7864, while the mean track length for the compressed and reconstructed versions is 3.8136 and 4.0116, respectively. These results suggest that the compression process has removed some image observations of the 3D points, leading to a reduction in the track length which indicates a less reliable 3D point estimate but may also improve the efficiency of the reconstruction process. However, the mean track length values for the original and reconstructed versions are relatively close to each other, indicating that the reconstruction process has been able to recover a significant portion of the original scene despite the compression.
The mean reprojection error is a measure of the accuracy of the 3D point estimates and is calculated as the average distance between the observed image points and the projected 3D points onto the image planes.Our results show that the mean reprojection error for the compressed versions of the P19 and P30 datasets are 0.4570 and 0.4722, respectively, which are slightly higher than the mean reprojection error of 0.4535 and 0.4767 for the original versions, respectively. The mean reprojection error for the reconstructed versions, obtained by applying the autoencoder to the compressed versions, are 0.4738 and 0.4782, respectively.For the Fountain P11 test dataset, the mean reprojection error for the original version is 0.3552, while the mean reprojection error for the compressed and reconstructed versions are 0.2583 and 0.2675, respectively. In general, our results suggest that the use of autoencoders for compressing local features has a minimal impact on the accuracy of the 3D point estimates, as measured by the mean reprojection error. The mean reprojection errors for the compressed and reconstructed versions of the datasets are similar to the mean reprojection error for the original versions, indicating that the compression and reconstruction processes have not significantly degraded the quality of the reconstruction.
For the Castle-19 and Fountain P-11 datasets mean projection error on the compressed version were slightly smaller compared to the original dataset. There are several potential reasons why the mean reprojection error is slightly higher for the compressed version of the P19 dataset compared to the original version, while it is slightly lower for the compressed versions of the P30 and Fountain P11 datasets compared to the original versions. One possibility is that the characteristics of the P19 dataset, such as the quality and quantity of the images, the camera parameters, the reconstruction method, or the characteristics of the scene itself, may make it more susceptible to information loss during the compression process. Another possibility is that the performance of the autoencoder model may vary across the different datasets, with the autoencoder being more effective at compressing the features in the P30 and Fountain P11 datasets compared to the P19 dataset due to differences in the distribution or complexity of the features.
It is important to consider multiple evaluation metrics when evaluating the performance of a 3D reconstruction algorithm, as different metrics may provide complementary or alternative perspectives on the reconstruction quality. In this study, we used three metrics: the number of points, the mean track length, and the mean re-projection error, to assess the reconstruction quality. By considering all of these metrics together, we can gain a more complete and comprehensive understanding of the reconstruction quality and identify any potential trade-offs or limitations of the reconstruction process. Therefore, it is important to consider a range of evaluation metrics and to analyze the results in the context of the specific characteristics and goals of the reconstruction problem.

Using COLMAP, we were able to determine the positions of the cameras in the three datasets used in this project. The positions of the cameras are an important factor in the 3D reconstruction process, as they determine the view of the scene and the relationship between the 3D points and the images. By accurately estimating the positions of the cameras, we can obtain a more accurate and complete 3D reconstruction of the scene. The results, shown in Figures 2, 3, and 4, demonstrate the ability of COLMAP to accurately estimate the positions of the cameras for each version of the datasets. By comparing the positions of the cameras across the different versions of the datasets, we can assess the impact of the compression and reconstruction processes on the accuracy of the camera positions and the overall quality of the reconstruction.

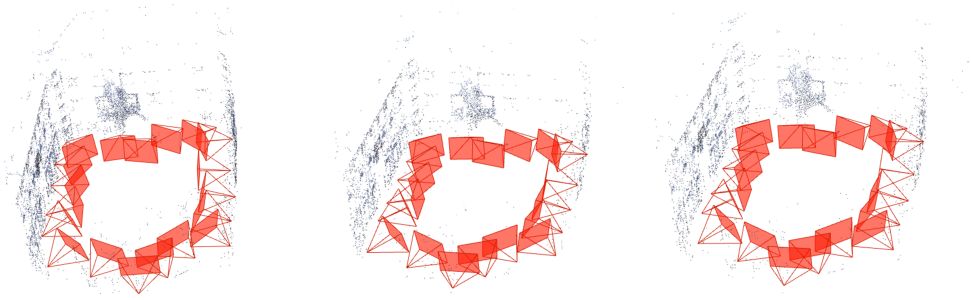

Figure 2: Original, Compressed and Reconstructed of Castle-P19 dataset

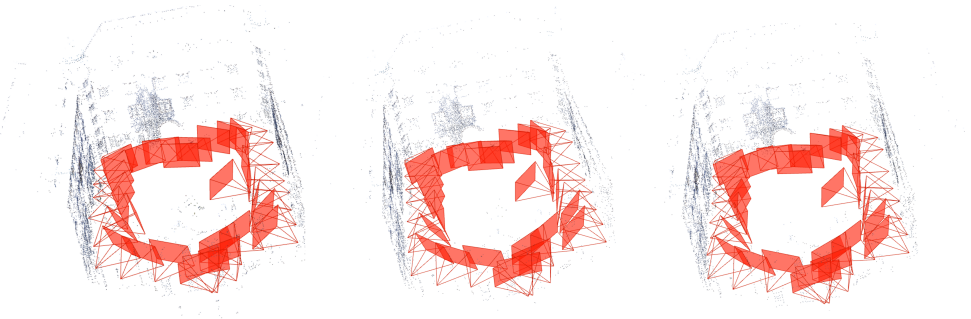

Figure 3: Original, Compressed and Reconstructed of Castle-P30 dataset

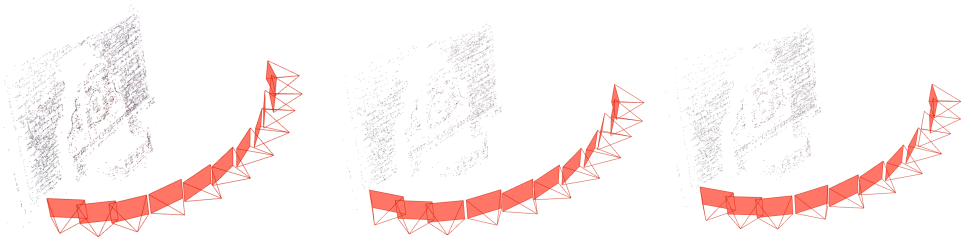

Figure 4: Original, Compressed and Reconstructed of Fountain-P11 dataset

Conclusion

The results that we have obtained suggest that the use of autoencoders for local features compression is a promising approach for improving the efficiency of 3D reconstruction algorithms without significantly affecting the accuracy in the case of Castle-P19 and Castle-P30. Instead, for Fountain-P11, the results suggest us that the usage of autoencoder permits to identify better the noisy points. This fact can be due to the more noise present in the test-set even if the losses of the training and the test set are similar.

References

[1] Simone Milani, *3D Augmented Reality lectures*, AY 2022/2023, University of Padova, `https://stem.elearning.unipd.it/course/view.php?id=2670`