COMP0249 - Coursework 2 part 2

Evaluating monocular ORBSLAM2 using the EVO tool on own sequences

Dhyan Shyam, Will Terry, YuChing Kwong
Computer Science
University College London

Running COLMAP and ORBSLAM2 with your own data comes with its own issues and challenges. In this section of the report we will break the collection, running and evaluation process into the following:

- What makes good data?
- How we go about collecting good data?
- How we run our data on COLMAP?
- How we convert our data to run on OrbSlam?
- How we add both to EVO?
- A comparison of the two mapping tools.

We will then analyses the differences between our COLMAP data and our ORBSLAM2 data, taking the COLMAP as our ground truth.

I. What makes good data?

In order for both COLMAP and ORBSLAM2 to effectively map a scene, object or area, the data should be up to a high standard. The data will never be perfect, but there are multiple steps to achieving high quality data, and they vary for both slightly.

A. COLMAP

The two main aspects we can control are the image quality and the frame rate of our video that we pass in. Based on COLMAP's own GitHub page [1] [2] there are four aspects to achieve strong reconstructions:

- Good Texture
- Similar Lighting Conditions
- High visual overlap
- Multiple Viewpoints

While we might not be able to get good texture everywhere, we can definitely improve the textures with high quality images. This is backed up by Carnegie Mellon University, which states that "We used Samsung Galaxy camera, which captures images that are 4608 pixels wide and 3456 pixels high; and videos that are 1920x1080. The idea is to get better quality images. Higher resolution images usually help preserve detail and get better reconstruction result." [3] So ideally choosing the highest resolution you can will give the best results.

To add to this, the data should be captured in a scene where the lighting is similar. For example data taken outdoors should stay outdoors and data captured indoors should stay indoors, with the same amount of lighting.

Depending on the speed you are moving at for COLMAP and the detail of your reconstruction required will determine the frame rate you pass in, as high visual overlap is required. On a GitHub discussion board for COLMAP, a contributor of the repo responded to a message stating "If you moved in walking speed with not too fast rotation, I would guess that 3-5 frames per second should be more than sufficient." [4]. We will base our frame rate off of these values.

Lastly multiple viewpoints will help achieve good quality reconstruction from all sides of the object/scene. I

Following each of these steps will lead to good COLMAP results and if followed should provide us with a high quality map of the path taken with the camera.

B. ORBSLAM2

ORBSLAM2 [5], [6], [7] has different requirements for effecting mapping of an environment, with the biggest one being the camera intrinsics. Unlike COLMAP, the intrinsics for Orb need to passed in before the mapping starts. It is therefore important to have the focal length, point, distortion etc. locked while recording. These can't be fixed in post so need to be implemented while collecting the data.

In terms of the quality of the images and the frame rate, Orb is meant to be run real time [5] [], so preferably a higher frame rate is required for real time mapping. Running real time with high quality images is very straining on a computer, meaning to run at this real-time a lower resolution should be used.

II. Capturing the Datasets

We have a requirement of having a high resolution video for COLMAP and a high frame rate for Orb, however COLMAP doesn't run well with a high frame rate as its unnecessary and orb doesn't run well at a high resolution as it slows down computation. Fortunately we can fix both of these issues in post by either scaling down the resolution or scaling down frame rate, meaning we should record at a high frame rate and resolution. All of our videos were captured at 4K at 24 FPS.

To add to this ORB has the additional feature of needing the camera the camera intrinsics to be locked, whereas COLMAP doesn't matter as much. Because of this we are using an application that allows us to bypass some of the Apple iPhone 14 built in software that effects the intrinsics. We used the black magic camera app [8], which allowed the following:

- Locked Focal Length: This keeps our fx, and fy parameters constant in the intrinsics.
- Camera Shake Off: software smoothens video by adjusting the frames to produce smoother motion when capturing video. Disabling this leads to shakier motion but keeps our focal point of cx and cy constant in our intrinsics.
- WB and contrast: Locking the White balance and the contrast stops the camera correcting for lighting changes, giving a more consistent scene when recording.
- Shutter Speed and ISO: Using a appropriate value for both the shutter speed and the ISO will give better lighting in the videos, but will be determined by the amount of light in the scene.
- Zoom: The iPhone has wide angle, standard and a telephoto lens. Using the standard lens will provide the least distortion to the distortion coefficients.

One major draw back of some of these parameters is from disabling the camera shake. With this motion becomes a lot less stable and more jumpy then usual, so to counter this some of the data sets were taken with a make shift dolly. We have three datasets in total, we have one indoors, and two outdoors, where one is driving around a neighbourhood and the other is running around a running track.

A. *Inside*

The inside was taken in a particularly well lit room, with very few windows. Extra features were put in place by placing objects on the tables, shown in figure 1, and chairs around the scene in order for ORB to have a better chance at calculating the paths.

Fig. 1: Example of extra features on the table

A make shift dolly was also made to keep the camera steady to counter the effects of disabling camera shaking. An office chair was pushed around the scene with the camera held on top for steady motion. The parameters chosen were:

- Shutter Speed: 1/100
- Focal Length: 0.7
- ISO: 50
- White Balance: 3200k
- FPS: 24

B. *Outdoor - Car*

The car dataset was captured towards dusk to prevent too much glare on the close by windows. The dataset shows a route through a quite neighbourhood with very few moving obstacles in the way for better features extraction. The camera was mounted to the top of the car for stable motion once again, and so that an increase in FPS was not needed the car was driven around 10-15 mph around the roads.

Fig. 2: Example of Car Dataset

- Shutter Speed: 1/100
- Focal Length: 0.8
- ISO: 50
- White Balance: 5600k
- FPS: 24

C. *Outdoor - Track*

Lastly, a data set was taken around a track, with the major difference being a lot more of the features would be picked up a lot further away in the distance compared to the camera, potentially effecting the performance of ORBSLAM2. The Lighting was taken at midday for the brightest shot, as there where no windows about to interfere with lighting. Lastly as it was trickier for stable motion on the track, the video was recorded strapped to someone's chest, and moved very slowly around the whole track.

Fig. 3: Track example, exhibiting features over a larger distance

- Shutter Speed: 1/100
- Focal Length: 0.8
- ISO: 50
- White Balance: 5600K
- FPS: 24

III. Calibrating the Camera using COLMAP

The next stage in the pipeline is to import our models into COLMAP, from this point forward the way in which to adapt

each video for COLMAP and orb is the same, so a general approach will be described.

Firstly the video was split into separate frames using the package ffmpeg, saving frames at a rate of 6 FPS, at full 4K resolution. Once input into COLMAP, there are three stages to creating the model.

A. Feature Extraction

The Feature matching tab allows us to set up how we would like to calculate the intrinsics for the camera, as we heavily rely on these for later use in ORB SLAM. We chose to represent our intrinsics in the OPEN_CV format, to to make sure we were correcting any distortion in the camera lens, with:

- fx, fy: Focal Length
- cx, cy: Principal Point
- k1, k2: Radial Distortion Coefficients
- p1, p2: Tangential Distortion Coefficients

Lastly as we were using images from a video, we can tick the box that says all images share the same intrinsics, leading to better, more accurate results.

B. Feature Matching

SIFT Feature matching with RANSAC is then run on the images and the extracted features. The default settings implement an exhaustive approach, taking a very long time to match all features to one another. Fortunately as the images are taken directly from a video in sequence, we can run it using sequential feature matching. The overlap between frames was also reduced to 5 instead of the default 10 as due to the decrease in frame rate before inputting into COLMAP, checking for matches over 10 frames would be 1.67 seconds of footage. Reducing this number shouldn't reduce the detail, but should reduce the overall time for computing the matches.

C. Reconstruction

Lastly the construction is as simple as running the start reconstruction, this is the longest part of the process and requires the most time.

This should provide the full COLMAP output, which can be exported normally in order to open it again later, as well as exporting as a ".txt" file so that we can read it for later running our COLMAP in EVO.

IV. Running ORBSLAM2

Running ORBSLAM2 comes with a few challenges. We need to mimic either the TUM or the KITTI dataset, in this case we have chosen the KITTI Dataset. This requires an image folder and the timestamps of the images, found by simply splitting the video into 24 FPS and saving each frame. Based on part one of the report, we will use 2000 features when running ORB, as this generally provided good results across the board, while maintaining efficiency. The last part is making a new YAML file with the intrinsics, but as we have down scaled the resolution, the intrinsics captured in COLMAP need to be adjusted, using the formula:

$$f'_x = f_x \times \frac{W_{\text{new}}}{W_{\text{orig}}} \tag{1}$$

$$f'_y = f_y \times \frac{H_{\text{new}}}{H_{\text{orig}}} \tag{2}$$

$$c'_x = c_x \times \frac{W_{\text{new}}}{W_{\text{orig}}} \tag{3}$$

$$c'_y = c_y \times \frac{H_{\text{new}}}{H_{\text{orig}}} \tag{4}$$

This gives the new intrinsics to add to the YAML file, where the distortions stay the same.

V. Running EVO

EVO is the comparison tool used for to compare how our COLMAP performs against our ORBSLAM2 performance. Some prep is required with the COLMAP file first in order to correctly read the path in a TUM format. There is also a difference in the coordinate frames which needs to be accounted for.

Firstly, the quaternions in the format world to camera are converted into rotation matrix.

$$\mathbf{R}_{cw} = \text{RotationMatrix}(q_x, q_y, q_z, q_w) \tag{5}$$

The Rotation matrix can then be converted into the world to camera format, and the translation is translated.

$$\mathbf{R}_{wc} = \mathbf{R}_{cw}^{\top} \tag{6}$$

$$\mathbf{t}_{wc} = -\mathbf{R}_{wc}\,\mathbf{t}_{cw} \tag{7}$$

Now we can apply the conversion, taking the flip of the Y and Z axis, then applying this transformation camer to world, translation and rotation matrix.

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{8}$$

$$\mathbf{R}_{wc}^{\text{TUM}} = \mathbf{T}\,\mathbf{R}_{wc} \tag{9}$$

$$\mathbf{t}_{wc}^{\text{TUM}} = \mathbf{T}\,\mathbf{t}_{wc} \tag{10}$$

Lastly the rotation matrix is converted back into a quaternion, and the saved in the TUM format.

$$(q_x^{\text{TUM}}, q_y^{\text{TUM}}, q_z^{\text{TUM}}, q_w^{\text{TUM}}) = \text{Quaternion}(\mathbf{R}_{wc}^{\text{TUM}}) \tag{11}$$

$$\text{timestamp} \quad t_x^{\text{TUM}} \quad t_y^{\text{TUM}} \quad t_z^{\text{TUM}} \quad q_x^{\text{TUM}} \quad q_y^{\text{TUM}} \quad q_z^{\text{TUM}} \quad q_w^{\text{TUM}} \tag{12}$$

The TUM format of the colmap data can then be passed into the EVO package along with the results from orb slam, using the COLMAP data as the "ground truth". We will save the Absolute Positional Error results, trajectory results and then compare them in EVO_RES.

VI. Results

Metrics for the Absolute Positional Error were recorded for each dataset to enable a comparative analysis, as summarized in Table I. These metrics have been scaled based on the estimate lengths traversed using google maps and measuring sticks, and the estimated lengths from colmap. In addition, differences in path lengths and timing were captured as supplementary comparison metrics, shown in Table II.

TABLE I: Scaled Comparison APE Metrics between The Car and Indoor

Dataset	Max	Mean	Median	Min	RMSE	SSE	STD
Car	1.48	0.56	0.48	0.059	0.62	218.24	0.26
Inside	0.014	0.0068	0.0068	0.000 68	0.0073	0.0350	0.0028
Track	56.03	25.79	26.07	2.21	28.83	466 960.12	12.41

In contrast to Part 1 of the report, and due to COLMAP's unreliable length estimations and the absence of ground truth measurements, the ORB-SLAM path lengths are expressed as a percentage relative to those reported by COLMAP, under the assumption that COLMAP provides more consistent path estimates.

TABLE II: Comparing Path Length and Time of run

Dataset	Total Length (m)	Path Completion (%)	Time (s)	Time Diff (s)
Car	396	99.77	95.67	−0.68
Indoor	11.5	99.79	56.00	−1.27
Track	460	48.75	82.92	−36.17

A more detailed evaluation of each dataset will follow, with a thorough discussion of these comparative results presented in the subsequent comparison section.

A. Car

COLMAP generated a highly accurate sparse reconstruction of the traversed area, as shown in Figure 4. The map successfully achieved loop closure, and textured surfaces such as house roofs and brickwork were well captured. This reconstruction provides a reliable basis for a "ground truth" reference against which the ORB-SLAM2 results can be compared.

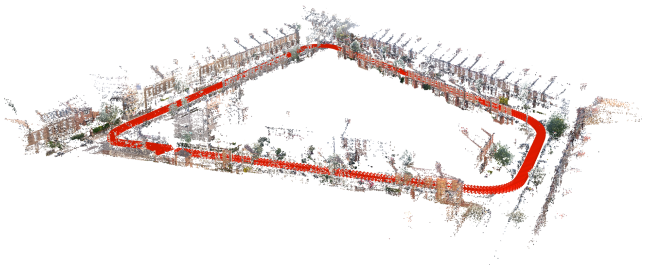
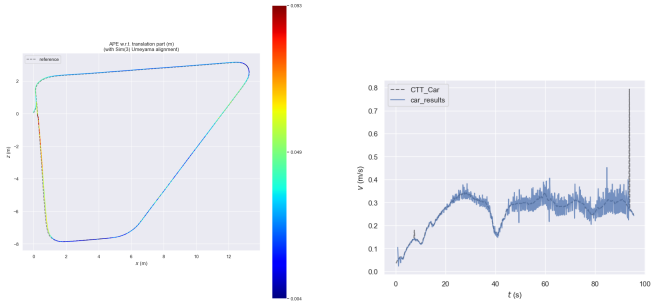


Fig. 4: Sparse reconstruction of the car route using COLMAP.

ORB-SLAM2 produced results that closely matched the COLMAP reconstruction, achieving this in just one minute compared to COLMAP's one-hour processing time.



(a) (b)

Fig. 5: Comparison of the car's trajectory and corresponding speeds.

Notably, ORB-SLAM2 performed very well around corners. This is likely due to the reduced speed in these sections, resulting in smaller frame-to-frame differences and less motion blur, which enabled better feature tracking leading to positional errors as low as 0.059m. In contrast, performance slightly degraded on the straight sections following turns, notably between 40–60 seconds and 80–100 seconds, where the car experienced higher acceleration. These segments likely suffered from fewer lateral feature changes compared to turns, reducing tracking robustness.

An interesting observation is that the noise in speed estimates gradually increased throughout the trajectory but sharply dropped once loop closure was achieved at approximately 94 seconds, when the car re-entered a previously mapped section of the road. The largest Absolute Positional Errors (APE) occurred towards the end of the route, where fewer visual features were present due to an open area on the right side of the road. Once loop closure was reached, the APE rapidly decreased.

In terms of RMSE, the car dataset exhibited a relatively low RMSE of 0.62 meters, indicating that overall, ORB-SLAM2's performance was consistent and accurate for most of the route. This is further supported by the very low mean APE of 0.56 meters (Table I).

Overall, ORB-SLAM2 closely reproduced the COLMAP path, achieving 99.77% path coverage as shown in Table II. The system initialized extremely quickly, with only a 0.68-second difference, indicating that the dataset provided a rich set of visual features and allowed for real-time operation. With a maximum APE of 1.48m and a mean of 0.56m (Table I), the ORB-SLAM2 results demonstrate very good accuracy most of the time, confirming that the data collection was highly successful.

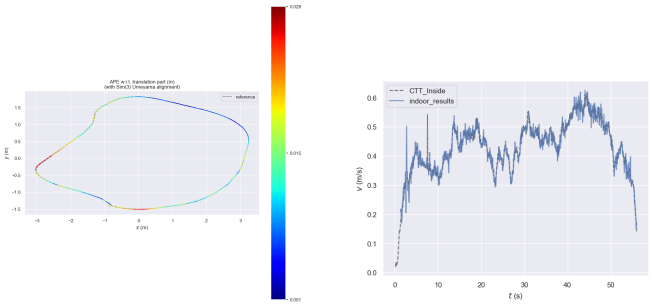
B. Indoor

Once again, COLMAP produced a highly accurate reconstruction of the environment, achieving successful loop closure. The addition of textured features across the scene significantly improved the mapping quality, transforming the previously featureless pool table into a surface rich with visual information. Although the bright yellow sofas were not fully

captured, sufficient detail from the objects placed on them was reconstructed, contributing to the overall accuracy of the scene.

Fig. 6: Sparse reconstruction of the indoor environment using COLMAP.

Similarly, ORB-SLAM2 produced a trajectory closely matching that of COLMAP, with the Absolute Positional Error (APE) graph showing nearly indistinguishable differences between the two systems.



(a) (b)

Fig. 7: Comparison of the indoor trajectory and corresponding speeds.

The indoor setting performed exceptionally well during long straights and gradual turns where the speed remained consistent, with errors as low as 0.0068 meters recorded between 30–45 seconds. However, performance degraded around sharp corners, where positional errors of up to 0.014 meters were observed. Reducing speed during these sharp turns would likely have improved performance, as the dataset maintained a relatively constant speed throughout.

In terms of RMSE, the indoor dataset demonstrated an impressively low value of 0.0073 meters, reinforcing the exceptional accuracy achieved by ORB-SLAM2 for this environment.

Overall, ORB-SLAM2 successfully captured 99.79% of the COLMAP path, as shown in Table II, with a time difference of 1.27 seconds. This slight delay suggests that initialization took longer, likely due to minimal movement during the first two seconds, as indicated by low speed values. This highlights that although lower speeds are generally beneficial for accurate reconstruction, excessively slow movement can hinder initialization due to insufficient disparity between frames.

Finally, the mean APE for the indoor dataset was 0.0068 meters (Table I), demonstrating that ORB-SLAM2 provided an extremely accurate reconstruction of the environment.

C. Track

The COLMAP reconstruction of the track demonstrated a notable decline in performance compared to the previous datasets. While the reconstruction effectively captured details of the exterior scene, including the grass and the tree in the background, it unfortunately failed to achieve loop closure, as evidenced by the path of the frames.

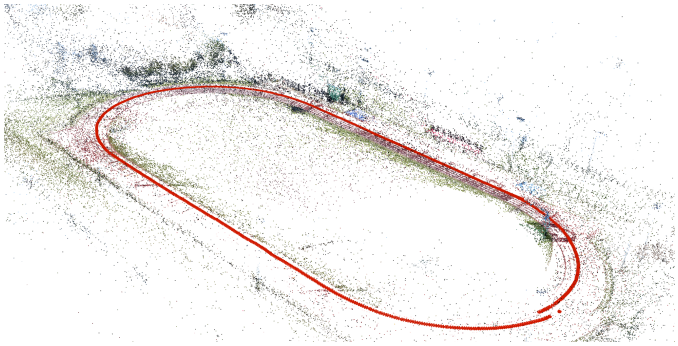
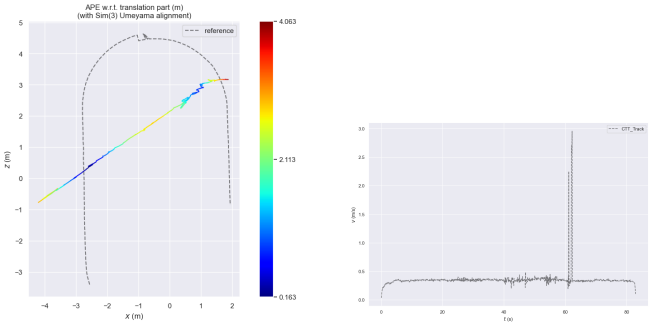


Fig. 8: Track Colmap

ORB-SLAM2 performed exceptionally poorly on the track. Initialization took an unusually long time, and once it was complete, the resulting map showed little resemblance to the COLMAP reconstruction.



(a) (b)

Fig. 9: Comparing the Car's Trajectory with Speeds

The poor performance of ORB-SLAM2 can be attributed to the lack of distinguishable features in the scene. As the track was located in the middle of a sparse field, the only reliable features were the trees and objects far from the track. The vast distances over 100 meters made it difficult for ORB-SLAM2 to distinguish any meaningful differences between features between frames, especially with the camera resolution set at 480p. Interestingly the ORB-SLAM2 managed loop closure for this dataset, causing the large cut across the image, as it reconnected towards the end. This added to the poor performance, as situationally the camera position looks very stagnate while ORB was running

When examining the performance quantitatively, errors as large as 56.03 meters were observed, indicating significant discrepancies between the mapped and actual paths. Even the minimum error was as large as 2.21 meters, suggesting that the path was never accurately reconstructed. Furthermore, the initialization time of 36 seconds highlights the lack of nearby features, further contributing to the poor map generation. Ultimately, only about half of the path was captured, which added to the poor performance in the positional error in the ORB-SLAM2 map.

In conclusion, ORB-SLAM2 struggles when there is a lack of close features to generate sufficient disparity, leading to underperformance in such scenarios. The track dataset exhibited a high RMSE value of 28.83 meters, emphasizing how poor the system's performance was in this setting.

D. Comparison

The car and indoor datasets demonstrated highly successful performance in capturing accurate reconstructions, with very low positional errors, reflected in their low RMSE values of 0.62 meters and 0.0073 meters, respectively. These datasets provided sufficient visual features for ORB-SLAM2 to work with, allowing for reliable map generation. The track dataset, however, presented a challenge due to a lack of distinguishable features, which led to poor performance, as indicated by a much higher RMSE of 28.83 meters. This highlights the importance of feature richness in the environment for ORB-SLAM2's performance. Overall, the differences in how these datasets were captured directly influenced the accuracy and robustness of the SLAM results, with well-featured environments leading to superior performance compared to sparse, featureless areas.

REFERENCES

[1] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixelwise view selection for unstructured multi-view stereo," in *European Conference on Computer Vision (ECCV)*, 2016.

[2] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] C. M. U. NavLab, "3d reconstruction with colmap," https://www.cs.cmu.edu/~reconstruction/colmap.html, 2015, accessed: April 2025.

[4] C. Contributors, "Image overlap in colmap," https://github.com/colmap/colmap/issues/388, May 2018, gitHub Issue #388.

[5] M. J. M. M. Mur-Artal, Raúl and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[6] S. Julier, "Refactored orb slam2," https://github.com/sjulier/Refactored_ORB_SLAM2, 2021.

[7] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[8] Blackmagic Design, "Blackmagic camera," https://www.blackmagicdesign.com/products/blackmagiccamera, 2023, accessed: April 2025.