A Warm Up

2. The trajectory error and translation error are in Table 1. The drawn trajectories is in Figure 1.

![Figure 1: Trajectory of original libviso.](image)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotation Error</td>
<td>$2.54 \times 10^{-3}$</td>
</tr>
<tr>
<td>Translation Error</td>
<td>1.33</td>
</tr>
</tbody>
</table>

Table 1: Rotation and translation error of original libviso.

3. 8-point algorithm only allows us to compute camera pose across two different frames, while 3-point algorithm allows us to use feature points from long tracks across multiple frames that have undergone extensive bundle adjustment.

4. "Bucketing" can make the feature distribution more uniform on the image plane, which leads to better conditioning for essential matrix estimation.

5. Since the height of the camera is known, the problem can be solved by first assigning a region of the image to be ground. Then based on the estimated 3D points in that region, we can scale the translation so that the estimated camera position matches the height of the car. A 1-point RANSAC is used to fit a ground plane to 3D points.

6. In libviso, RANSAC is performed by randomly sampling 8 correspondences and computing
an essential matrix. For each essential matrix, the number of inliers is determined. The essential matrix with the largest number of inliers is used to compute rotation and translation.

B Using SIFT [2] for SFM

2 The trajectory error and translation error are in Table 2. The drawn trajectories is in Figure 2. Notice that the location of keypoints predicted by VLFEAT is 1-based, while the keypoints used in libviso is 0-based. However, it won’t have a major impact on the performance.

![Figure 2: Trajectory of original SIFT.](image)

<table>
<thead>
<tr>
<th></th>
<th>2.55 × 10⁻³</th>
<th>1.37</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotation Error</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Translation Error</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Rotation and translation error of SIFT.

3 SIFT algorithm does not perform better than the original libviso framework. For both translation and rotation, the accuracy is slightly worse. The potential reasons why SIFT performs worse include:

- The libviso SFM framework is specifically designed for its feature detection and description (such as multiscale feature matching, triangulation to remove outliers, bucket size), which may not be optimal for other types of features.
- The trajectory of SIFT feature has some outliers after the first turn. These outliers can be due to lack of sufficient number of accurate matches to estimate the camera pose.
- The libviso features are more numerous in the ground region, while SIFT seeks more discriminative matches. Matches in the ground region are important for scale correction.
4 To achieve rotation invariance:

-Compute the direction of gradient in the nearby regions for each keypoint to determine the orientation of keypoint. When computing the feature descriptor, rotate the gradient according to the direction of keypoint.

To achieve illumination invariance:

-Use gradients to compute feature descriptor, so adding a constant in the lighting will not influence the feature.
-Normalize the feature vector to a fixed length so that multiplying a constant to the lighting will not influence the feature.
-Threshold the magnitudes in the unit feature vector to be no larger than 0.2, so that large intensities will have less influence on the final feature descriptor.

To achieve scale invariance,

-Use DoG for keypoint detection, which is only different from normalized Laplacian of Gaussian $\sigma^2 \nabla^2 G$ by a constant factor. And $\sigma^2 \nabla^2 G$ is known to be scale-invariant.
-When computing the orientation and extracting feature descriptor, use the image with the closest scale in the Gaussian pyramid.

C Using SuperPoint [1] for SFM

2 The trajectory error and translation error are in Table 3. The drawn trajectories is in Figure 3.

![Figure 3: Trajectory of original SuperPoint.](image)
<table>
<thead>
<tr>
<th>Rotation Error</th>
<th>$3.46 \times 10^{-3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Translation Error</td>
<td>$1.36$</td>
</tr>
</tbody>
</table>

Table 3: Rotation and translation error of SuperPoint.

3 SuperPoint does not achieve performance better than libviso. It achieves translation better than SIFT feature, but the rotation error is much larger.

- The libviso SFM framework is specifically designed for its own feature detector and feature descriptor, so may not be optimal for other features.
- SuperPoint is trained on COCO images and synthetic images with input size $240 \times 320$, so may not generalize well to KITTI dataset.
- SuperPoint does not have subpixel accuracy. Its feature descriptor is obtained by bilinear upsampling, which might influence performance.
- SuperPoint will output a keypoint every $8 \times 8$ pixels. So that the feature will distribute more uniformly in the image. That may explain why it performs better than SIFT in estimating translation, since there are more feature points on the ground.

4 Obtaining ground-truth for keypoints:

- For synthetic data, the keypoints are directly generated. For real data, homographic adaptation is used to get the pseudo ground-truth. Randomly sampled homographies are applied to real images from COCO dataset and the network trained on synthetic data is used to do keypoint detection. The keypoints are warped back to the original image and then aggregated together as the ground-truth. The intuition is that the network should learn to predict consistent keypoints across homography transformations.

Obtaining ground-truth matches:

- Since the image is transformed by a randomly generated homography, it is easy to find the matches using the homography.

Learning a correlated feature representation for keypoint detection and feature descriptor:

- A single encoder is used to learn the representations for both keypoint detection and feature descriptor.

D Using SpyNet [3] for SFM

2 The trajectory error and translation error are in Table 4. The drawn trajectories is in Figure 4.

3 SpyNet performs worst among the four methods for motion estimation. The reason might be that SpyNet predicts dense matches between two frames. However, there are many textureless regions in the image where there is not enough information to find a good match.
Figure 4: Trajectory of original SpyNet.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Rotation Error</strong></td>
<td>$3.58 \times 10^{-3}$</td>
</tr>
<tr>
<td><strong>Translation Error</strong></td>
<td>1.49</td>
</tr>
</tbody>
</table>

Table 4: Rotation and translation error of SpyNet.

Thus, the overall accuracy of matches found by SpyNet is not accurate compared to sparse keypoint-based methods, which results in poor estimates of rotation and translation.

4 A cascade structure is trained for optical flow estimation. It starts with low resolution and gradually moves to higher resolutions. Each level of the cascade only predicts the residual flow, which is a simpler problem, so a relatively simpler network can be used at each level.

References

