Visual Monocular Odometry: ”Beating Around the Bush”

Viviane BAO
Master 2 SAR
Student number: 3800857

Hao YUAN
Master 2 SAR
Student number: 21117163

***Abstract*—In this research, we delve into enhancing camera localization using advanced computer vision techniques. Initially, we detect points of interest through the Speeded Up Robust Features (SURF) method, which provides a robust foundation for feature detection. We then apply epipolar geometry and triangulation to calculate depth information, essential for understanding the scene’s three-dimensional structure. To estimate the camera’s initial position accurately, we utilize the Perspective-n-Point (PnP) method. Furthermore, to refine our data, we introduce a low-pass filter, smoothing the trajectory estimations and minimizing noise. Integrating these methods, we present a unified strategy that enhances camera tracking and depth measurement, yielding substantial gains in accuracy and reliability for camera localization in complex settings.**

I. INTRODUCTION

This project is a dive into the nuanced world of computer vision, aiming to achieve precise localization of a camera as it orbits a box. Starting with simple geometric markers, the project will evolve to address the intricacies of textured environments. This progression mirrors the transition from theoretical concepts covered in ”Geometry for Vision” to practical, real-world applications. The endeavor goes beyond traditional methods, employing a creative setup to track the camera’s path, thus ensuring our results are grounded in observable truth.

II. METHODS

A. Camera calibration

Camera calibration is a critical process in computer vision that is employed to ascertain the camera’s internal (intrinsic) and external (extrinsic) parameters. The intrinsic parameters are encapsulated in a matrix commonly known as the camera matrix or matrix K, which includes the focal length, the optical center, and any lens distortion coefficients. This matrix is vital for converting the camera’s 2D image plane points into 3D world coordinates accurately. Determining matrix K is a fundamental goal of camera calibration, as it enables precise image analysis and application in various computational photography and 3D computer vision tasks.

B. Camera pose initialization

To estimate the initial position and orientation of the camera, we begin by manually matching points to address the Perspective-n-Point (PnP) problem. The Perspective-n-Point (PnP) method is a computational approach used in computer vision to estimate the pose of a camera in a 3D space. The pose consists of the camera’s position and orientation relative to a known set of points in the environment. Typically at least four non-coplanar points are required, but six points provide redundancy and increase accuracy. By matching these 3D points with their 2D projections in the camera’s image, the PnP algorithms can solve for the camera’s location and rotation using geometric relationships.

First, we obtain the matrix M through SVD decomposition, and then we calculate the matrices R and T through QR decomposition. For estimating camera motion, we simultaneously utilize both epipolar geometry and homography matrix estimation. Both of these methods involve solving overdetermined equations through least squares.
Initially, matrices E (essential) and H (homography) are obtained through SVD. Subsequently, we compute various potential combinations of R and T based on predefined formulas. Through triangulation and the calculation of reprojection errors, we refine these solutions to more plausible ones. For each instance of motion estimation, we are presented with two potential estimates and corresponding errors; the estimate yielding the smallest reprojection error is selected as the most probable solution (this section will be explored in depth in the subsection ”Epipolar Geometry and Triangulation”).

To gain a better understanding of the Singular Value Decomposition (SVD) decomposition, consider the following: the Singular Value Decomposition (SVD) of a matrix E is expressed as follows: $E = U\Sigma V^T$, where U and V are orthogonal matrices, and Σ is a diagonal matrix with singular values $(\sigma, \sigma, 0)$.

Within the framework of SVD decomposition, for any given matrix E, there exist two potential pairs of vectors t and rotation matrices R associated with it. These pairs are calculated as follows:

$$t_1 = UR_z\left(\frac{\pi}{2}\right)\Sigma U^T \qquad R_1 = UR_z^T\left(\frac{\pi}{2}\right)\Sigma U^T$$
$$t_2 = UR_z\left(-\frac{\pi}{2}\right)\Sigma U^T \qquad R_2 = UR_z^T\left(-\frac{\pi}{2}\right)\Sigma U^T$$

Here, $R_z\left(\frac{\pi}{2}\right)$ represents a 90-degree rotation matrix about the Z-axis. Both -E and E are equivalent, choosing a negative sign for any vector t will yield the same result. Therefore, when decomposing E into t,R, there are a total of four potential solutions to consider.

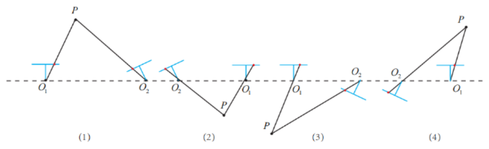


Fig. 1. The four solutions obtained from the decomposition of the essential matrix

Only the first scenario shows point P with a positive depth in both camera views, allowing for the correct solution by depth verification. The usual practice is, after performing the SVD decomposition of E obtained by the eight-point method, we get a singular value matrix $\Sigma = diag(\sigma_1, \sigma_2, \sigma_3)$, assuming $\sigma_1 > \sigma_2 > \sigma_3$, we take: $E = Udiag(\frac{\sigma_1+\sigma_2}{2}, \frac{\sigma_1+\sigma_2}{2}, 0)V^T$. This is equivalent to projecting the obtained matrix onto the manifold where E resides. Of course, a simpler method is to set the singular value matrix to diag(1,1,0), because E has scale invariance, and this is also a reasonable approach.

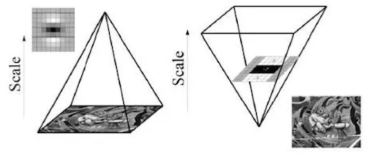
C. Interest point detection and matching

The SURF method is used to detect and match interest points, offering rapid and robust image comparison. Its key feature is the speedy computation of operators with box filters, enabling real-time applications like tracking and object recognition. SURF is composed of two steps: feature extraction and feature description.

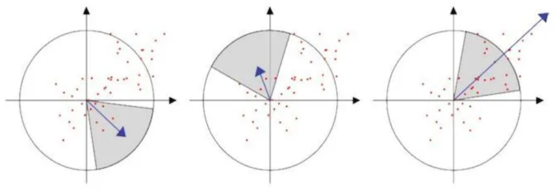
- Feature extraction : this approach relies on a basic Hessian matrix approximation and utilizes Integral Images or Summed-Area Table for rapid calculations of pixel values or average intensity within a given image. The integral image is given by : $I_\Sigma(X)$=$\sum_{i=0}^{i\leq x}\sum_{j=0}^{j\leq y} I(i,j)$.

  Surf uses the Hessian matrix because of its good performance in computation time and accuracy.Given a pixel, the Hessian of this pixel is something like: $H(f(x,y)) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix}$ For adapt to any scale, we filtered the image by a Gaussian kernel, so given a point X = (x, y), the Hessian matrix H(x, σ) in x at scale σ is defined as: $H(x,\sigma) = \begin{bmatrix} L_x x(x,\sigma) & L_x y(x,\sigma) \\ L_x y(x,\sigma) & L_y y(x,\sigma) \end{bmatrix}$ where $L_x x(x,\sigma)$ is the convolution of the Gaussian second order derivative with the image I in point x, and similarly for $L_x y(x,\sigma)$ and $L_y y(x,\sigma)$.

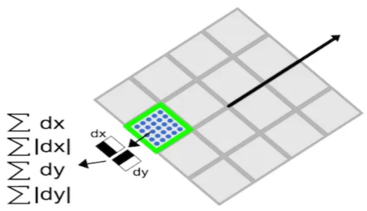
  Gaussians are optimal for scale-space analysis but in practice, they have to be discretized and cropped.
  Scale space in image processing is typically formed by creating a pyramid of images that are progressively smoothed and reduced in size. However, the SURF algorithm simplifies this by using box filters and integral images to apply large filters at constant speed without reducing the image size. This method increases filter size for each octave and doubles the sampling interval to detect interest points efficiently. Non-maximum suppression is used to pinpoint the precise interest points across scales.



- Feature description : The SURF descriptor is created through a two-step process: first by assigning a consistent orientation using the area around the keypoint, and then by forming a square region aligned with this orientation from which the descriptor is extracted.
  To achieve rotation invariance, SURF calculates the Haar-wavelet responses within a circular area around a key-point, using the scale of detection to determine the size and sampling step. It then sums these responses, adjusting the orientation incrementally until it finds the one with the largest response, which becomes the keypoint's main orientation. Integral images are utilized for efficient computation at various scales.

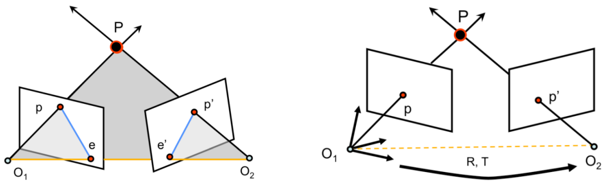


To extract the SURF descriptor, a square region around the keypoint is created, aligned with the previously determined orientation, and divided into 4x4 sub-regions. Haar wavelet responses, dx and dy, are calculated at 5x5 sample points within each sub-region, weighted by a Gaussian (σ=3.3s) centered at the keypoint for robustness.



These responses and their absolute values are summed to form a four-dimensional vector per sub-region (V=($\sum dx, \sum dy, \sum |dx|, \sum |dy|$)), resulting in a 64-dimensional descriptor for the entire region, contributing to SURF's speed compared to SIFT's 128-dimensional vector.

D. Epipolar geometry and triangulation

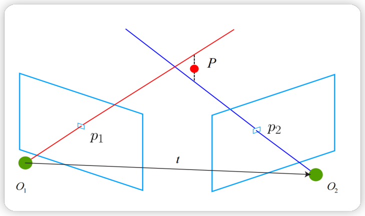


In multiple view geometry, epipolar geometry is central to understanding how a pair of cameras, a 3D point, and its projections onto the cameras' image planes are interrelated. Two cameras (with centers atO_1 and O_2) observe a 3D point P. The projections of P onto the image planes are p and p', and the line connecting O_1 and O_2 is the baseline. The epipolar plane is defined by the 3D point and the two camera centers. Where this plane cuts the image planes, it defines the epipolar lines, which converge at the epipoles (e and e') on each image plane.

An interesting scenario arises when the image planes are parallel, causing the epipoles to lie at infinity and making the epipolar lines parallel. This configuration simplifies the geometry and is particularly relevant in image rectification, which makes the stereo images more manageable for analysis and processing.

Moving from the geometric setup to practical application, let's delve into how we map points and epipolar lines between views using camera projection matrices M and M'. These matrices transform 3D points to their 2D projections in each camera's image plane, with M=K[I 0] and $M' = [R^T - R^T T]$, assuming the world reference system aligns with the first camera. Particularly in the context of canonical cameras (where K=K'=I), these matrices simplify, facilitating a clearer understanding of the interaction between camera orientation, position, and the observed 3D structure..

Central to these relationships is the Essential Matrix (E), defined as E=[T_x]R T_x where is the matrix representation of the cross product with the translation vector T and R is the rotation matrix. This matrix encapsulates the epipolar constraint $p^T E p' = 0$, ensuring that corresponding points in two camera views are collinear with their respective camera centers. The Essential Matrix is not just a theoretical construct; it's instrumental in computing epipolar lines and essential for recovering camera motion and 3D reconstruction in stereo vision, provided the camera's intrinsic parameters are known.



Since from a fundamental matrix and a set of corresponding points, the scene can only be reconstructed up to a projective transformation. Thus, it is crucial to have a triangulation method that is invariant to projective transformation. Denote τ as the triangulation method used to compute a 3D point X from a point correspondence x$\leftrightarrow$ x' and a pair of camera matrices P, P'. We have: X=τ(x,x',P,P'). The invariance to projective transformation H is expressed as: τ(x,x',P,P')= $H^{-}1\tau(x, x', PH^{-}1, P'H^{-}1)$.
Ultimately, decomposing the Essential Matrix reveals several potential rotations R and translations T that could describe the relationship between two camera views. To find the most physically plausible camera setup, the process involves triangulating 3D points from matched pairs of 2D points in the image planes and evaluating the resulting depth values. This approach effectively filters out any camera configurations that lead to impossible scenarios, such as points appearing behind the camera, and confirms the correct configuration for accurate scene reconstruction. The chosen configuration is the one that maximizes the number of points with positive depth, leading to a consistent and geometrically valid understanding of the scene.

E. Homography

Homography is a transformation in computer vision that maps points between two planes, typically relating two different views of a scene in images. It is captured by a 3x3 matrix known as the Homography Matrix (H), which relates points p in one plane to points p' in another as $p' = Hp$, or
$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} h_1 1 & h_1 2 & h_1 3 \\ h_2 1 & h_2 2 & h_2 3 \\ h_3 1 & h_3 2 & h_3 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$
The matrix accounts for rotation, translation, scaling, and perspective changes. Although it assumes a flat scene or purely rotational movement, homography is crucial in applications such as image stitching, object recognition, and 3D reconstruction, helping to understand and manipulate the spatial relationships between different viewpoints.

To compute homography, typically four or more pairs of corresponding points are needed between the two views. The Direct Linear Transform (DLT) algorithm is commonly used for this purpose.This algorithm solves homogeneous system. In case of estimating a homography, it takes the

following form: x'=Hx, x∝Hx. The algo follows these steps:
1. Gets at least 4 point correspondences between two views
2. For each point correspondence, setup a partial A_i matrix of the form 2x9
3. Assembles A matrix to the form of 8x9 or 2nx9
4. Applies Singular Value Decomposition to the A matrix
5. Assuming the entries in D are sorted in descending order, the last column of V is the solution to h. Reshape h 9x1 into H of the form 3x3

The resulting homography matrix can then be used to warp one image to align with the other, rectify images taken from different perspectives, or extract geometric and motion-related information about the scene or the camera.

F. Data filtering and camera trajectory tracing

Data filtering involves smoothing 3D tracking data by reducing its sampling rate and applying a low-pass filter to remove high-frequency noise. This results in a cleaner trajectory of an object's movement, which is then visually compared to the original data through a 3D plot, highlighting the effectiveness of the noise reduction process.

For camera tracking, we construct a path based on a series of camera movements. It begins by setting an initial transformation matrix that includes both rotation and translation components. For each subsequent camera position, a new transformation matrix is created and multiplied with the accumulated transformation to update the camera's position in 3D space. The result is a series of points representing the camera's trajectory. This trajectory is then visualized in a 3D plot, allowing one to observe the camera's path through space with respect to the X, Y, and Z axes. The plot is enhanced for clarity with grid lines, markers at each point, and labeled axes, providing a comprehensive view of the camera's motion over time.

III. Experimentation and Evaluations

A. Camera calibration

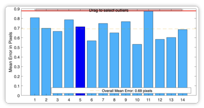


Fig. 2. Distribution of Mean Localization Error per Image with Overall Mean Error

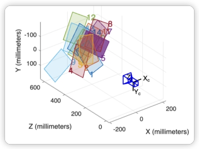


Fig. 3. Visualization of Camera Pose Estimation Relative to Calibration Patterns"

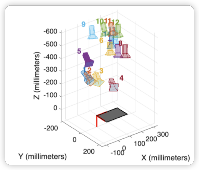
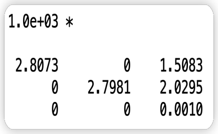


Fig. 4. Multi-View Camera Pose Alignment with Reference Object"

The three images offer a comprehensive visual representation of camera pose estimation in a multi-view setup. In the first image, we see a series of calibration patterns and the camera's estimated position and orientation in relation to these patterns, which are crucial for determining the intrinsic and extrinsic parameters. An overall mean error of 0.69 pixels suggests that, on average, the localization or feature detection error is less than one pixel. In the field of computer vision, especially in tasks requiring high precision like camera calibration or 3D reconstruction, a mean error value under one pixel is typically considered very good. The third image provides a clear depiction of the camera's trajectory, highlighting the dynamic range of motion captured during the calibration process. Furthermore, it adds depth to our understanding by showing the camera's alignment from various perspectives, further validating the calibration's accuracy.

The matrix K obtained is:



```
1.0e+03 *

   2.8073         0    1.5083
        0    2.7981    2.0295
        0         0    0.0010
```

Fig. 5. Matrix K

B. Camera pose initialization

As previously mentioned, the PnP method requires the identification of six points. These points have three-dimensional coordinates (X, Y, Z) relative to a world frame defined by the observer. By analyzing the initial sequence, one can infer the pixel coordinates of each of the six points, thereby establishing the camera's initial pose within the image's frame of reference.

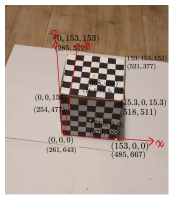


Fig. 6. Geometric Calibration of a 3D Object in a 2D Image

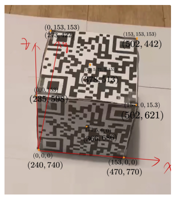


Fig. 7. Geometric Calibration of a 3D Object in a 2D Image

C. Interest point detection and matching

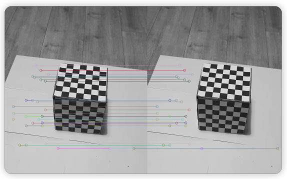


Fig. 8. Geometric Calibration of a 3D Object in a 2D Image for 20 Frames (chessboard)

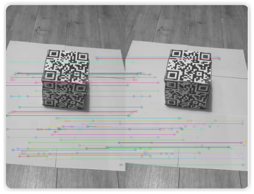


Fig. 9. Geometric Calibration of a 3D Object in a 2D Image for 20 Frames (QR code)

The results demonstrate an accurate point correspondence between two stereoscopic images of a checkered cube, reflecting successful stereo calibration and feature point extraction.

The proper alignment of the epipolar lines and the correct matching of points attest to the consistency and precision of the stereo correspondence, which are crucial for a reliable depth estimation in this project.

D. Reprojection error over a sequence of frames

A reprojection error is the distance between a pattern keypoint detected in a calibration image, and a corresponding world point projected into the same image

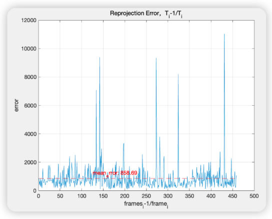


Fig. 10. Reprojection error for Frame 50 to 400 (Chessboard)

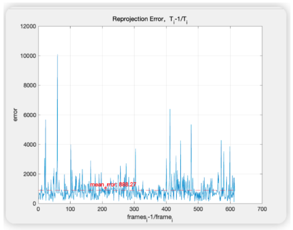


Fig. 11. Reprojection error for 615 Frames (QR Code)

It is observed that the estimated errors are generally significant, except for a few isolated instances. However, for the majority of the estimates, the errors appear to be stable, fluctuating between seven hundred and eight hundred. Determining whether this average error is large or small is challenging without a thorough understanding of the camera's intrinsic parameters, as such an assessment could be directly influenced by them. It is also noted that the camera's intrinsic parameters are around the order of magnitude of 10^3, which, when put into perspective, could make the average error relatively acceptable. Nevertheless, it would be prudent to analyze the impact of these errors on the final applications, because even an average error that seems moderate could have significant consequences on the precision of 3D reconstruction or on the performance of camera-assisted navigation systems.

E. Data filtering and camera trajectory tracing

The two images illustrate the effect of applying a filter to a camera's motion track data. In the first image, the filtered track displays a smooth and continuous path, reflecting a successful noise reduction process. The second image presents both the unfiltered and filtered camera tracks. The unfiltered track is chaotic and full of noise, indicating erratic movement that is likely not representative of the actual camera path. We have taken into account the frames from 50 to 400 for our analysis, as the initial frames were affected by hand tremors.

For the last two figures provided, we observe a similar effect of filtering on the camera trajectory.

IV. Conclusion

The project reached a successful conclusion with the precise mapping of the camera's trajectory, a testament to the meticulous application of image processing techniques. Our methodical approach, underpinned by diligent calibration and analysis, yielded a transparent and trustworthy charting of the

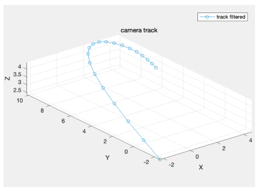


Fig. 12. Camera track for Frame 50 to 400, track filtered(Chessboard)

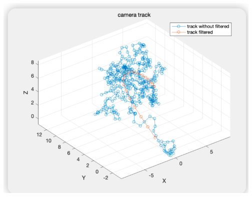


Fig. 13. Comparison between track filtered and not filtered (Chessboard)

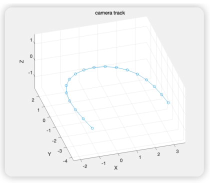


Fig. 14. Camera track for Frames 50 to 400, track filtered (QR Code)

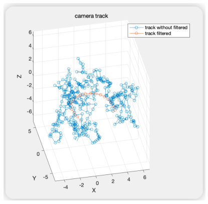


Fig. 15. Comparison between track filtered and not filtered (QR Code)

camera's motion around the focal point. This achievement not only validates the effectiveness of our methods but also opens the door to future research and development, particularly in enhancing real-time tracking techniques and motion analysis in various practical contexts.

References

[1] Richard Hartley and Andrew Zisserman, "Multiple View Geometry in Computer Vision Second Edition", March 2004.
[2] J. Shi and Tomasi, "Good features to track, 1994.
[3] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf", 2011
[4] D. Forsyth and J. Ponce, "Computer Vision - A modern approach 2nd ED", 2012
[5] X. Gao and T. Zhang, "Introduction to Visual SLAM", 2021