Stereo Visual Odometry

Marthinus J. (Marno) Nel

MSAI 495 – Introduction to Computer Vision

May 31, 2023

Table of contents

1. Personal views about the field

Computer vision is a multidisciplinary field that focuses on developing algorithms and techniques to enable computers to extract meaningful information from visual data. This interdisciplinary field draws upon concepts from computer science, mathematics, image processing, and artificial intelligence to tackle a wide range of applications. It encompasses various fundamental concepts and methods such as binary image processing, region segmentation, edge detection, motion analysis, geometry, stereo vision, and more. These techniques form the foundation of computer vision and are essential for tasks like image analysis, object recognition, tracking, and visual odometry.

Arguably the most fundamental goal of computer vision is to enable computers to perceive and understand the visual world as good as a human or even exceed human capabilities. This involves fundamental techniques such as feature extraction, and pattern recognition in means of edges, shapes, textures, etc. This can then be used to infer higher-level information about the image or video.

Recently Convolutional neural networks (CNNs) have revolutionized the computer vision field with a large increase in overall performance, and increase in capabilities. These networks require large labeled datasets, they are expensive to train, and they are generally not generalizable. CNN models automatically extract intricate features and enable end-to-end learning without manual feature engineering.

Traditional computer vision techniques require less data, less training time and can be developed at a fraction of the cost than CNNs and are mathematically proven. Thus traditional techniques should be used except if CNNs give remarkably better performance.

Computer vision still faces challenges in real-world variations like occlusions, lighting, and viewpoint changes. Robustness and reliability of computer vision systems is crucial, especially in safety-critical applications.

In this course, I have gained foundational knowledge of the core principles in computer vision and was introduced to advanced topics. I learned basic binary image analysis and connected-component labeling (CCL) with binary images. CCL is used for image segmentation to identify and assign a unique label to each connected component or object in the image which is crucial to identify and separate meaningful regions within an image

I also learned about morphological operators to manipulate images by removing noise, enhancing the image shape and getting the boundary of an object in the image. This is achieved with operations like dilation, erosion, opening and closing.

Histogram equalization was introduced to improve the contrast and brightness of an image. Skin detection was achieved with a histogram-based skin color detector in the HSV color space with minimal training examples.

The course has also covered edge detection, a key technique for identifying boundaries and contours in images. A Canny edge detector is a multi-stage algorithm that detects edges in images by applying a series of filters to reduce noise, calculate image gradients, suppress non-maximum edges, and using a high and low threshold with recursion to determine final edges.

Hough transforms were introduced for line and circle detection in images. It works by converting the image space to a parameter space. Template-matching based tracking was also introduced as a computer vision technique used for object tracking in video streams.

Additionally, I have gained knowledge in camera calibration, pose estimation, image stitching, visual features, stereo vision, texture modeling, and Face and object detection.

Looking ahead, I plan to further develop my skills in computer vision in the field of robotics. I am planning to study local visual features, stereo vision, 3D reconstruction and visual slam. With respect to CNNs I plan on learning more about YOLO and other real time object recognition algorithms. These topics are very interesting and relevant to the field of robotic perception and navigation. They enable robots to sense the world around them and act accordingly.

By continuing to build upon the fundamental concepts learned in this course and delving into more advanced topics, I can further deepen my understanding of computer vision and its applications in the field of robotics. This will equip me with the necessary knowledge and skills to develop robust and efficient computer vision systems for autonomous robots, object manipulation, and navigation tasks.

2. Project description

The goal of this project is to develop a stereo visual odometry pipeline and to test it on the KITTI dataset. Visual odometry is the process of estimating the 3D motion of a camera by analyzing the changes in visual features in consecutive frames. Through stereo vision, which utilizes two cameras to recover depth information, we aim to accurately track the camera's position and orientation over time in 3D space.

The term visual odometry was chosen for its similarity to wheel odometry as both these techniques incrementally estimate the motion of the camera. Visual odometry is a critical component of many computer vision applications, particularly in the field of robotics and autonomous navigation. It provides the information necessary to enable autonomous tasks such as 3D mapping, localization, and path planning for mobile robots. The visual odometry task is challenging due to a lot of factors such as camera noise and environmental conditions. Thus, error accumulation is inevitable and can only be minimized.

This project focuses on implementing stereo visual odometry on the KITTI dataset. The KITTI dataset is a widely used benchmark dataset for autonomous driving research, consisting of stereo image pairs, lidar data, and ground truth poses. The dataset provides a realistic and challenging environment.

The main objectives of the project are as follows:

1. **Stereo Image Preprocessing:** Load stereo images, camera projection matrix and ground truth poses.

2. **Feature Extraction and Matching:** Perform feature extraction and matching between two consecutive left camera frames. Utilize Lowe's ratio test to identify good features.
3. **Disparity Map:** Generate the disparity map using the stereo image pair.
4. **Stereo to Depth Map:** Construct the depth map using the disparity map, the camera's focal length and the stereo camera's baseline distance.
5. **Motion Estimation:** Utilize the matched features and the camera's intrinsic matrix to estimate the camera's motion between consecutive frames. This involves triangulating the matched feature points and computing the camera's rotation and translation using algorithms like the 8-point algorithm or RANSAC. Which is known as 3D to 2D correspondence.
6. **Trajectory Accumulation:** Calculate the cumulative transformation matrix by applying the dot product of the between frames transformation matrix and total transformation matrix.
7. **Trajectory Evaluation:** Evaluate the accuracy of the estimated camera trajectory by comparing it with the ground truth poses provided in the KITTI dataset. This step allows for quantitative analysis of the system's performance and the identification of areas for improvement.

By successfully implementing stereo visual odometry on the KITTI dataset, this project aims to develop an accurate and robust pipeline for estimating camera motion for future work in visual SLAM.. Through this project, valuable insights into the challenges and techniques associated with stereo visual odometry in the realms of robotics and autonomous systems will be gained.

My partner and I wrote individual code for this project. However, we discussed and implemented each step together, referred to the same documentations, and obtained and evaluated the results together.

3. Pipeline design

This section describes the pipeline design and implementation of stereo visual odometry on the KITTI dataset. Visual odometry is a technique used to estimate the motion of a camera by analyzing the

changes in visual features captured by the camera over time. The KITTI dataset provides stereo image sequences along with ground truth poses, which allows us to evaluate the accuracy of our odometry estimation. Figure 1 below displays the general pipeline for the visual odometry problem. In this project 3D to 2D motion estimation was implemented.

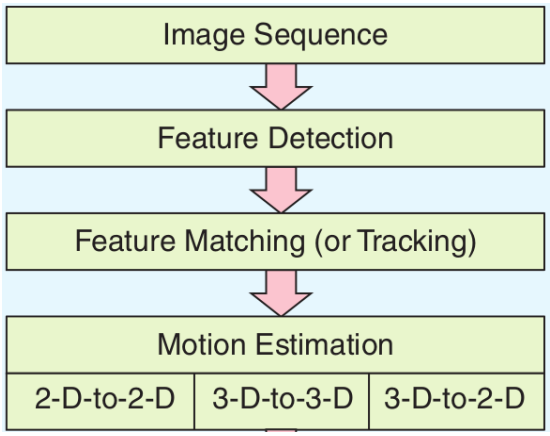

Figure 1: Stereo Visual Odometry Pipeline.

1. **Preprocess data:**

The camera's projection matrix is loaded from the calibration file. The intrinsic intrinsic matrix is then extracted from the projection matrix with the function *cv2.decomposeProjectionMatrix()*.
Next the ground truth poses for the dataset is loaded into a list that is used to evaluate the performance of the visual odometry algorithm. Lastly the grayscale stereo images are loaded into a list for the stereo visual odometry algorithm.

2. **Feature detection:**

The features detection function takes in the detector type, the grayscale image to extract features from, and an optional mask to limit the search region for an increase in performance. It performs feature detection and extraction using either of the following feature detectors:

- **Scale-Invariant Feature Transform (SIFT) -** It detects distinctive keypoints in an image that are invariant to scale changes, rotation, and affine transformations, and provides robust descriptors for matching and recognizing those keypoints. It works by identifying scale-space extrema in the image at multiple scales using Difference of Gaussian (DoG) filters. The descriptor is then computed for the keypoint to capture the local appearance and orientation of the keypoint with histogram gradient orientations. Figure 2 below displays how SIFT works as described above.

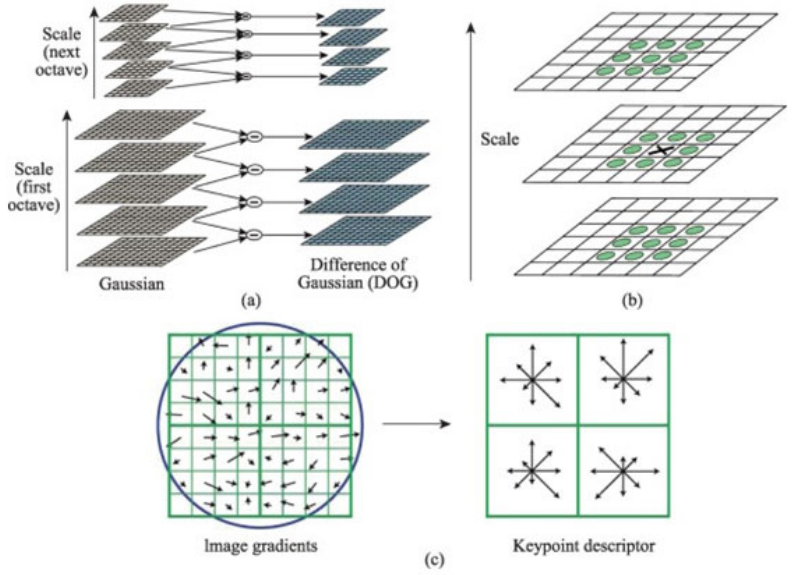

Figure 2: Visualization of Scale-Invariant Feature Transform (SIFT) derector.

- **Oriented FAST and Rotated BRIEF (ORB)**: It combines the key points from the FAST (Features from Accelerated Segment Test) algorithm and the descriptors from the BRIEF (Binary Robust Independent Elementary Features) algorithm. ORB first detects keypoints using the FAST algorithm, which identifies pixels with high intensity changes. These keypoints are used to compute binary descriptors using BRIEF, which compares pairs of pixel intensities to create a distinctive representation of the keypoints. Image rotations are handled by incorporating orientation estimation.

Figure 3 below is an example of the features detected on KITTI's dataset 09:

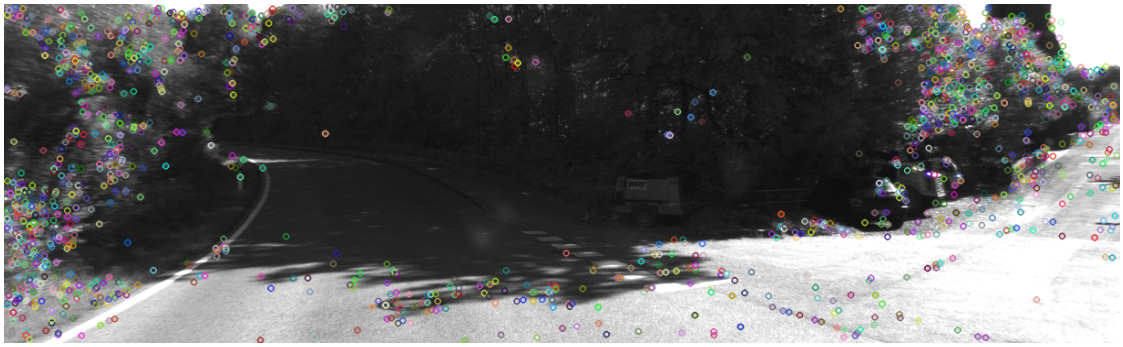

Figure 3: ORB features.

3. **Feature matching:**

The Brute-Force (BF) Matcher was used for feature matching between two consecutive frames. It is a straightforward algorithm, but it is computational intensive and is often used as a baseline.

It works by comparing each feature descriptor of one image to all the feature descriptors of the next image and finds the best matches based on a chosen distance metric. The chosen distance for SIFT is *cv2.NORM_L2* which is used for real values and *cv2.NORM_HAMMING* for ORB which is binary values.

K-Nearest Neighbors (KNN) is used with k = 2 as the classification algorithm that assigns two nearest neighbors to the feature in the next image. Thus each feature in the current image is matched with two features in the next image.

Figure 4 below is an example of the matched features in two consecutive images:

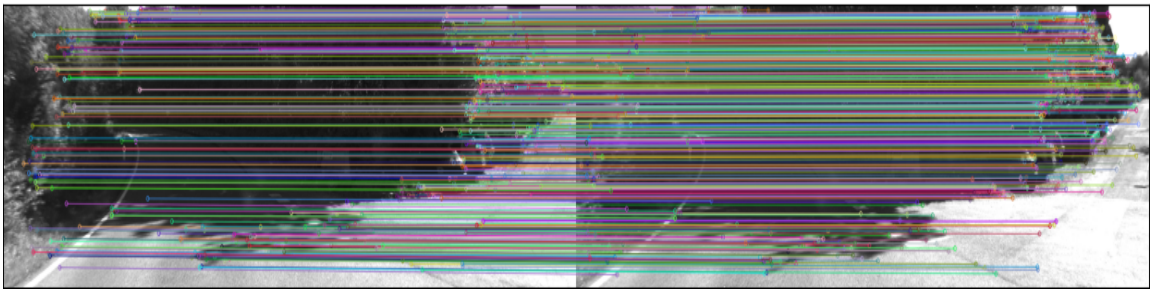

Figure 4: Matched feature between two images with the Brute-Force matcher.

4. Lowe's Ratio Test:

Lowe's ratio test is a technique used in feature matching to filter out unreliable matches. The test involves comparing the distance ratio between the best and second-best matches, from the KNN and feature matching algorithm, for each feature. If the ratio is below a certain threshold the match is considered ambiguous and discarded. This test helps improve the robustness and accuracy of feature matching by reducing the number of false matches. A ratio of 0.45 was used for SIFT and 0.6 for ORB. Figure 5 below is a visualization of Lowe's ratio test.

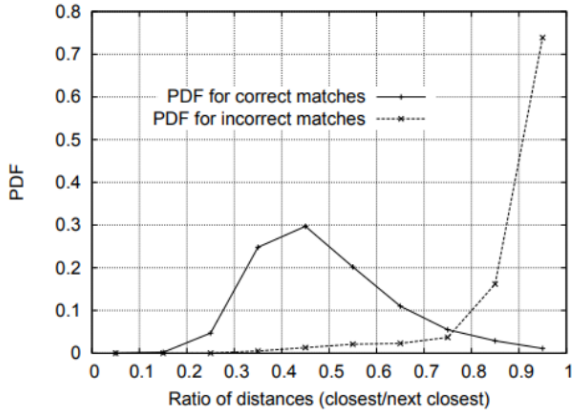

Figure 5: Lowe's ratio test visualized.

5. **Masking:**

A mask is generated to restrict feature detection to a specific region of the image. In this case, a rectangular mask is defined to exclude the area where the stereo images do not overlap. The mask is represented as a binary image.

6. **Compute the Disparity Map:**

The disparity map is computed from the stereo images. The disparity map assigns a disparity value to each pixel in the left image, indicating how far the corresponding pixel in the right image is horizontally shifted. This disparity value represents the depth information or the distance between the camera and the objects in the scene. The below two algorithms were used to generate the disparity map:

- **Stereo Block Matching (StereoBM)** estimates the disparity map between a pair of stereo images by dividing the stereo images into small blocks and finding the corresponding block in the other image by comparing the pixel intensities. It measures the similarity between blocks using a matching cost, such as the sum of absolute differences (SAD) or the sum of squared differences (SSD). The algorithm then selects the disparity value that minimizes the matching cost for each block, producing a disparity map. StereoBM is simple and computationally efficient, which results in a lower accuracy and increase in sensitivity to noise and occlusions.
- **Stereo Semi-Global Block Matching (StereoSGBM)** is based on block matching, where correspondences between pixels in the left and right images are found by comparing local image patches as in StereoBM. StereoSGBM takes into account both local and global information by aggregating matching costs over multiple paths in the disparity space. This helps to reduce errors caused by occlusions and textureless regions. It also has a cost aggregation step and a disparity refinement step to produce an accurate disparity map. Its robustness and ability to handle challenging stereo pairs with varying lighting conditions and complex scenes makes it a good choice for increased robustness in the VO pipeline.

Figure 6 is a disparity map generated with StereoSGBM on the KITTI dataset. It is evident that a car is in the middle right of the picture and a tree at the left top of the picture. It can also be concluded that the car is closer than the tree by evaluating the change in color or the increase in disparity value.

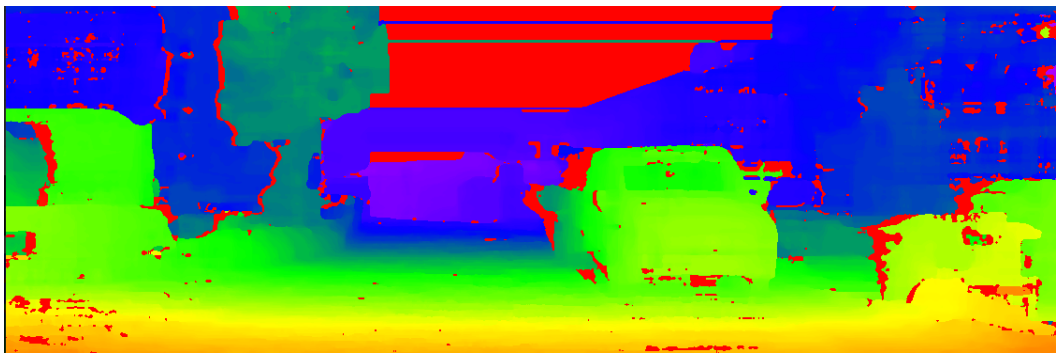

Figure 6: Disparity map output from the StereoSGBM algorithm.

7. Compute the Depth Map:

A depth map provides information about the distance of objects in a scene relative to the camera. Each pixel is assigned a depth value, indicating how far away objects are from the camera's viewpoint at that pixel. The formula below is used to calculate the depth of a object at a specific pixel using the camera intrinsic parameters and the disparity map:

$$Depth = \frac{f * b}{disparity}$$

The focal length of the camera is denoted as f and b is the baseline of the stereo camera. To handle uncertain values in the disparity map, the program sets zero values and -1 (indicating no overlap between left and right camera images) to a small value of 0.1 in the disparity map. This ensures that the depth estimation for these points is considered very far away and can be ignored. Figure 7 below shows the similar triangles used to calculate the depth of a calibrated stereo camera system.

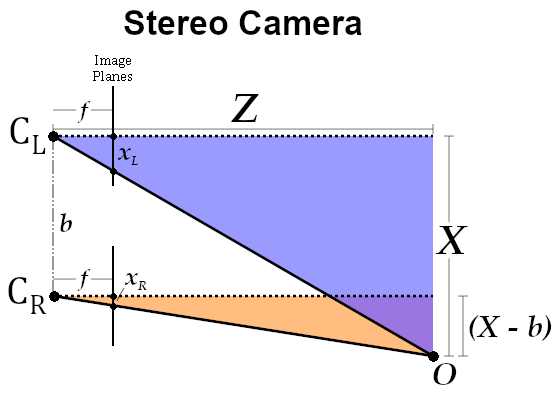

Figure 7: Similar triangles used for stereo camera's depth estimation.

The resulting depth map is displayed in Figure 8 below..

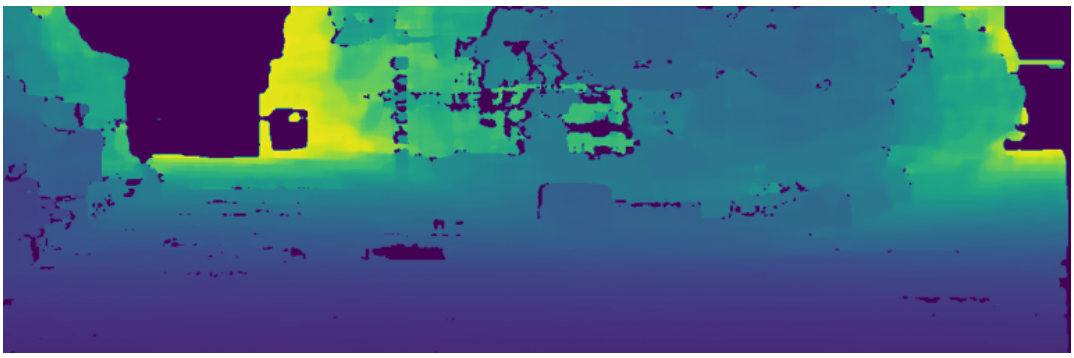

Figure 8: Depth map.

8. Estimate Camera Motion:

The final step in this stereo visual odometry is 3D-to-2D motion estimation of the camera over time. The matched features, keypoints, depth map, and a depth threshold is used to estimate the transformation matrix (T) that represents the motion between two consecutive frames. The 2D coordinates of the matched features in the previous and current frames are extracted from the keypoints. Then, the camera's intrinsic parameters (cx, cy, fx, fy) are obtained.

Next, a loop iterates over the matched feature points. For each point, the corresponding depth value is retrieved from the depth map. If the depth value exceeds the specified threshold, the point is considered unreliable and marked for deletion. Otherwise, a 3D point is computed using the depth value and camera parameters with the below equations:

$$x = \frac{u - cx}{fx} * depth$$

$$y = \frac{v - cy}{fy} * depth$$

$$z = depth$$

After removing the unreliable points, the remaining feature points and their corresponding 3D points are used to estimate the transformation matrix, rotation and translation vectors, using the *solvePnPRansac* function from OpenCV. The resulting rotation vector is then converted to a rotation matrix using Rodrigues' transformation.

The total camera transformation matrix (C) is calculated after each frame's motion estimation to keep track of the overall trajectory of the camera. It is calculated by multiplying the total camera transformation matrix with the inverse of the current transformation matrix (T). This accumulates the transformations over time, allowing the estimation of the camera's trajectory.

Figure 9 displays the stereo visual odometry problems evolution over time. The camera’s start position to a fixed world frame is represented by the transformation matrix Ck-1. At the next frame Tk,k-1 is calculated as the motion estimation between frames k and k-1. To estimate the total camera trajectory at the new frame the camera's previous transformation matrix Ck-1 is then multiplied with the inverse of Tk,k-1 to generate the new camera transformation matrix Ck. This is done for each frame with C always holding the transformation matrix of the camera relative to its starting position.

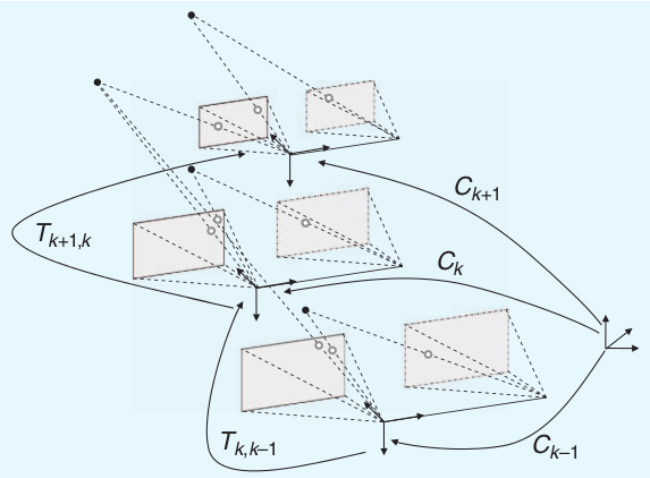

Figure 9: 3D-to-2D Stereo visual odometry motion estimation.

4. Results and analysis

The stereo visual odometry pipeline was evaluated on the KITTI dataset 09. The two factors that influenced the stereo visual odometry outcomes the most were the disparity map algorithms and the feature detectors. The ground truth duration of the dataset is 2 min and 45 seconds.

The two best results obtained with the respective feature detectors are listed below:

1. SIFT as the feature detector with a lowe's ratio of 0.45 and StereoSGBM was used to calculate the disparity map. This resulted in a total runtime of 10 min and 35 seconds with an endpoint euclidean error of 3.6 m. Figure 10 below displays the groundtruth path in black and the VO estimated path in red. It is evident that these specific parameters perform well at estimating the camera motion over time and minimizing the error drift over time. This experiment's runtime is 3.8 times that of the ground truth which is expected as SIFT and StereoSGBM are both computational inefficient.

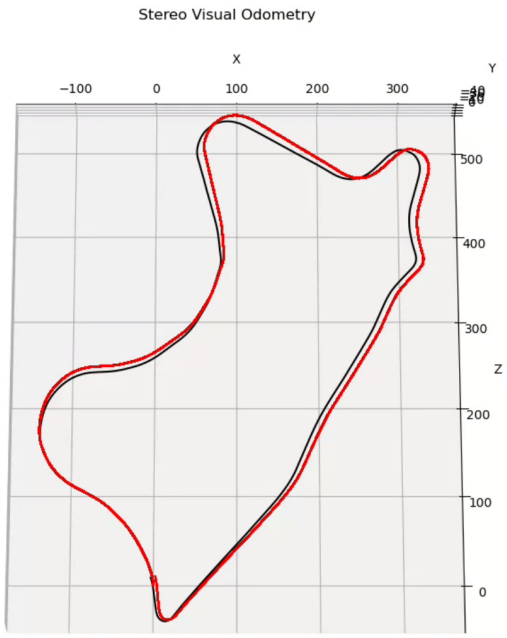

Figure 10: Stereo visual odometry path with SIFT and StereoSGBM.

2. ORB as the feature detector with a lowe's ratio of 0.6 and StereoBM was used to calculate the disparity map. This resulted in a total runtime of 5 min and 50 seconds with an endpoint euclidean error of 66 m. Figure 11 below displays the groundtruth path in black and the VO estimated path in red. It is evident that these specific parameters perform well at estimating the camera motion over time and minimizing the error drift over time. This experiment's runtime is 2.12 times that of the ground truth which is expected as ORB and StereoBM are both more computationally efficient than SIFT and StereoSBGM.

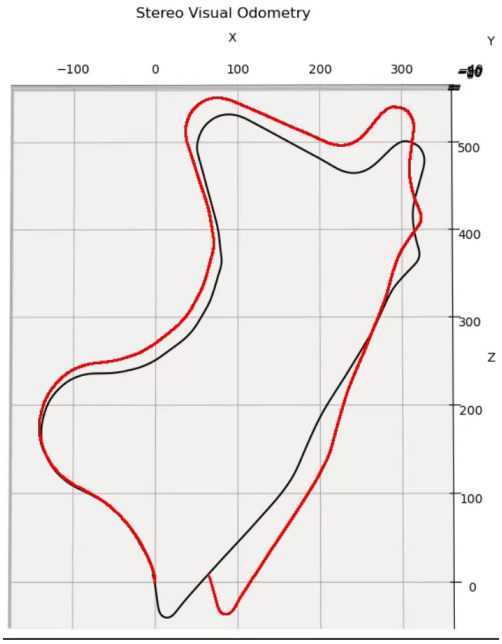

Figure 10: Stereo visual odometry path with ORB and StereoBM.

Table 1 below summarizes the findings from the two experiments.

Table 1: Result from stereo visual odometry experiments.

Experiment parameters	Total runtime [Time]	Endpoint euclidean error [m]
Ground truth	2 min 45 seconds	NA
ORB, StereoBM, and ratio = 0.6	5 min 50 seconds	3.8
SIFT, StereoSGBM, and ratio = 0.45	10 min 35 seconds	66 m

From the table above it can be derived that neither ORB or SIFT can be applied as is for real-time visual odometry on an entire image. The experiment with ORB is calculated to be 55% more computational efficient than SIFT where SIFT's experiment has 94.5% higher accuracy to the end-point on dataset 09 of the KITTI dataset.

5. Future work

- **Real-Time Performance:** Optimize the computational efficiency of the stereo visual odometry pipeline to achieve real-time performance on grayscale images. This could involve algorithm optimizations, parallelization, or hardware acceleration techniques.
- **Sensor Fusion:** Integrate other sensors, such as IMU or Lidar to improve accuracy and robustness of the estimated motion. A Kalman filter or particle filter can be applied to combine the data from multiple sensors effectively .
- **Lidar Data:** Incorporate lidar depth data for 3D reconstruction by mapping the lidar point cloud to the camera pixels.
- **ICP (Iterative Closest Point) Odometry:** ICP odometry is a technique used in robotics and computer vision to estimate the relative motion between two 3D point clouds by iteratively aligning them through the correspondence of their closest points, enabling accurate localization and mapping in environments with overlapping or dynamic objects.
- **Depth Estimation:** Investigate methods to improve depth estimation from grayscale stereo images. This could involve techniques such as deep learning-based depth estimation models.
- **Loop Closure:** Incorporate loop closure techniques into the visual odometry pipeline to handle the re-observation of previously visited locations, improving the accuracy and consistency of the estimated camera trajectory.
- **Visual SLAM:** Extend the stereo visual odometry system to a full-fledged visual SLAM system, integrating loop closure, map building, and pose graph optimization to simultaneously estimate camera motion and reconstruct the environment.

6. Course feedback and suggestions

The course covers a comprehensive range of computer vision topics with an emphasis on the practical application of computer vision techniques. This is great as students receive hands-on experience with a wide range of fundamental computer vision techniques.

I personally do not have a lot of computer vision experience and this was a great course to lay the foundation and introduce advanced computer vision topics. Providing a historical overview of computer vision and contrasting it with modern neural network approaches effectively explained the current landscape of the field. Additionally, comparing traditional computer vision techniques and neural networks offered valuable insights into their respective strengths, enabling a better understanding of the subject.

The project was a great way to apply computer vision in the field of robotics and the course significantly helped me in understanding and implementing the stereo visual odometry project.

References

[1] Scaramuzza, Davide & Fraundorfer, Friedrich. (2011). Visual Odometry [Tutorial]. IEEE Robot. Automat. Mag.. 18. 80-92. 10.1109/MRA.2011.943233.

[2] Fraundorfer, Friedrich & Scaramuzza, Davide. (2012). Visual Odometry: Part II - Matching, Robustness, and Applications. IEEE Robotics & Automation Magazine - IEEE ROBOT AUTOMAT. 19. 78-90. 10.1109/MRA.2012.2182810.

[3] Geiger2012CVPR, Andreas Geiger and Philip Lenz and Raquel Urtasun, Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite, Conference on Computer Vision and Pattern Recognition (CVPR), 2012

[4] Biadgie, Yenewondim & Sohn, Kyung-Ah. (2014). Feature Detector Using Adaptive Accelerated Segment Test. ICISA 2014 - 2014 5th International Conference on Information Science and Applications. 33. 1-4. 10.1109/ICISA.2014.6847403.

[5] Lindeberg, Tony. (2012). Scale Invariant Feature Transform. 10.4249/scholarpedia.10491.

[6] Rublee, Ethan & Rabaud, Vincent & Konolige, Kurt & Bradski, Gary. (2011). ORB: an efficient alternative to SIFT or SURF. Proceedings of the IEEE International Conference on Computer Vision. 2564-2571. 10.1109/ICCV.2011.6126544.

[7] Calonder, Michael & Lepetit, Vincent & Strecha, Christoph & Fua, Pascal. (2010). BRIEF: Binary Robust Independent Elementary Features. Eur. Conf. Comput. Vis.. 6314. 778-792. 10.1007/978-3-642-15561-1_56.

[8] Benlakhdar S, Rziza M, Thami R. O. H. A Robust Model using SIFT and Gamma Mixture Model for Texture Images Classification: Perspectives for Medical Applications. Biomed Pharmacol J 2020;13(4). Available from: https://bit.ly/3heperH

[9] Scaramuzza, Davide and Friedrich Fraundorfer. “Visual Odometry Part I: The First 30 Years and Fundamentals.”.