COMP-4990

Vision-based Obstacle Detection

Joshua Gehl
Austin Formagin
Jarrett Jackson
Brady Malott

Supervisor: Dr. Boufama

Objective

- Vision-based obstacle detection for objects in a straight path using two images
  - Identify key points in two images using the **SIFT** (Scale Invariant Feature Transform) algorithm [1]
  - Pair up the key points using **FLANN** (Fast Library for Approximate Nearest Neighbors) library [2]
  - Use **RANSAC** (Random Sample Consensus) algorithm [3] to compute a homographic matrix, which maps points from the first image to the second image
  - Use the homographic matrix to map points from the first image to the second image and determine the existence of an object

Project Requirements

- Determine the presence of objects on a path by inputting two images
- Implemented using OpenCV and python
- Must use the RANSAC algorithm to compute a homographic matrix
- Intuitive user interface to input images and display the detected objects

Key Point Detection & Feature Matching

- Images are converted to greyscale to reduce the number of channels, which reduces computation time
- Used the SIFT feature detector over ORB due to SIFT's greater accuracy at detecting key points
- Used the FLANN-based matcher to generate a list of possible matched key points between the images. FLANN returns the two best matches for each point in the first image
- Removed ambiguous matches using the Lowe's Ratio Test
  - If the distance between the two best matches is not sufficiently different, the match is removed.
- The result is a list of key points that are matched between the two inputted images.

SIFT (Scale-Invariant Feature Transform)

- Created by D.Lowe in 2004 at UBC
- Detects features within an image regardless of scale or orientation
- Four main steps:
  - **Scale-space extrema detection** – Uses difference of gaussian on the scale-space and finds local extrema amongst neighbouring pixels
  - **Key point localization** – Uses Taylor series expansion of scale space and Harris corner detection to eliminate low contrast and edge key points
  - **Orientation assignment –** An orientation histogram with 36 bins is created from neighbours around a key point. Any peaks above 80% are considered to calculate the orientation.
  - **Key point descriptor –** 16x16 neighbourhood around the key point is divided into 16 sub blocks, each with an 8-bin orientation histogram, represented as a vector.

[4] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," International Journal of Computer Vision, vol. 60, no. 2, pp. 91–110, Nov. 2004, doi: https://doi.org/10.1023/b:visi.0000029664.99615.94.

SIFT (Con't)

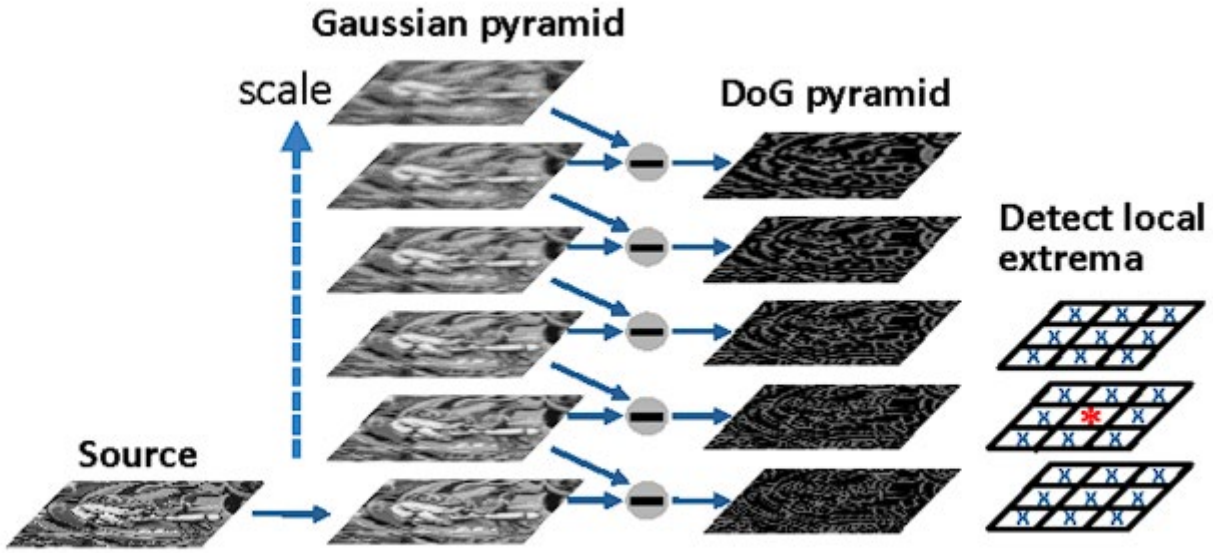


Figure 1: Scale-space extrema detection

Uses difference of gaussian on the scale-space and finds local extrema amongst neighbouring pixels

[5] G. Wang, B. Rister and J. R. Cavallaro, "Workload analysis and efficient OpenCL-based implementation of SIFT algorithm on a smartphone," 2013 IEEE Global Conference on Signal and Information Processing, Austin, TX, USA, 2013, pp. 759-762, doi: 10.1109/GlobalSIP.2013.6737002.

SIFT (Con't)

Figure 2: Key points drawn with orientation

[1] "OpenCV: Introduction to SIFT (Scale-Invariant Feature Transform)," Opencv.org, 2020. https://docs.opencv.org/4.x/da/df5/tutorial_py_sift_intro.html

Feature Matching

- The two types of feature matchers that were considered include:

- **FLANN (Fast Library for Approximate Nearest Neighbors)**
  - Contains a collection of algorithms that work best for nearest neighbor searches in high-dimensional spaces [2]

- **Brute Force**
  - Matches features using distance measurements such as:
  - Euclidian Distance: L1, L2, L2SQR
  - Hamming Distance

Key Point Detection & Feature Matching (Con't)

Figure 3: Two images used to test, which were scaled down to test different resolutions

Key Point Detection & Feature Matching (Con't)

Images		SIFT/FLANN		ORB/Brute Force	
Image 1 Resolution	Image 2 Resolution	Execution Time (s)	Key Point Matches Found	Execution Time (s)	Key Point Matches Found
3852x3000	4080x3072	8.983	1485	0.541	160
1926x1500	2040x1536	1.477	912	0.269	140
963x750	1020x768	0.383	196	0.199	122
385x300	408x307	0.064	16	0.173	114

Figure 4: Execution time and number of matches found for SIFT/FLANN and ORB/Brute Force

Key Point Detection & Feature Matching (Con't)

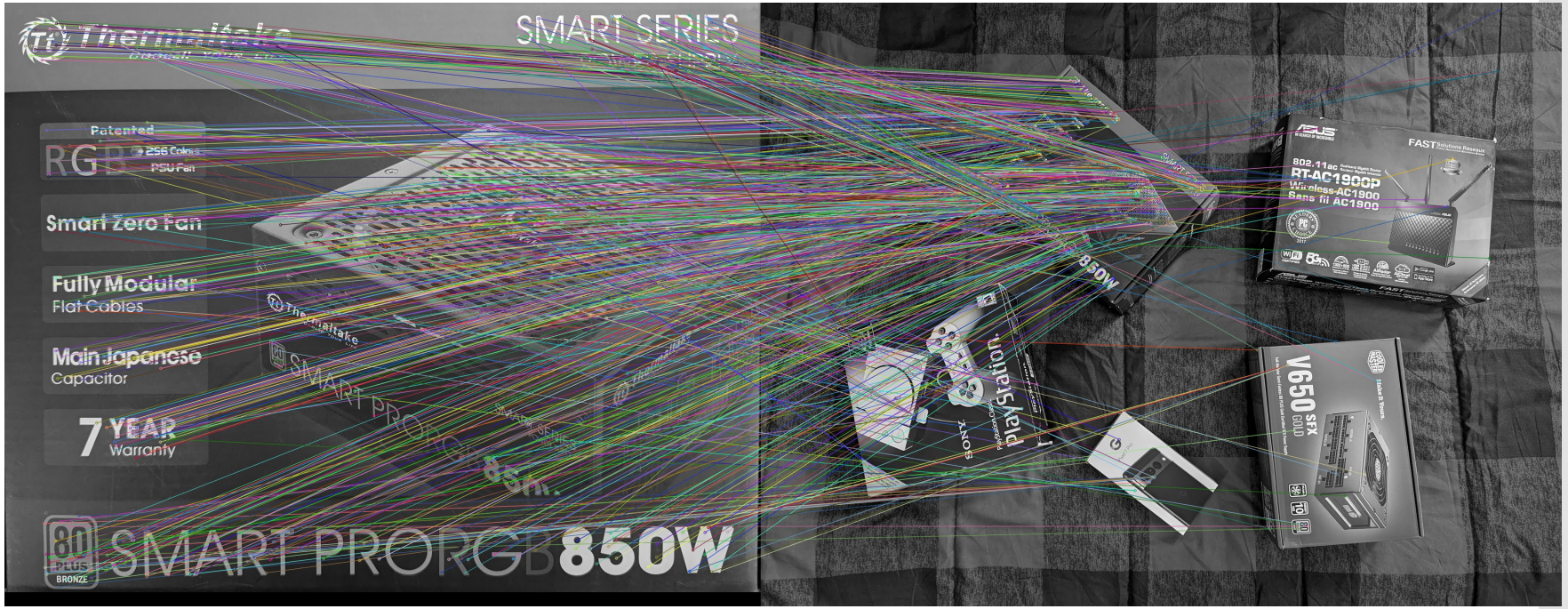


Figure 5: Results of SIFT key point detection and FLANN-based matching on the 50% size images

Key Point Detection & Feature Matching (Con't)

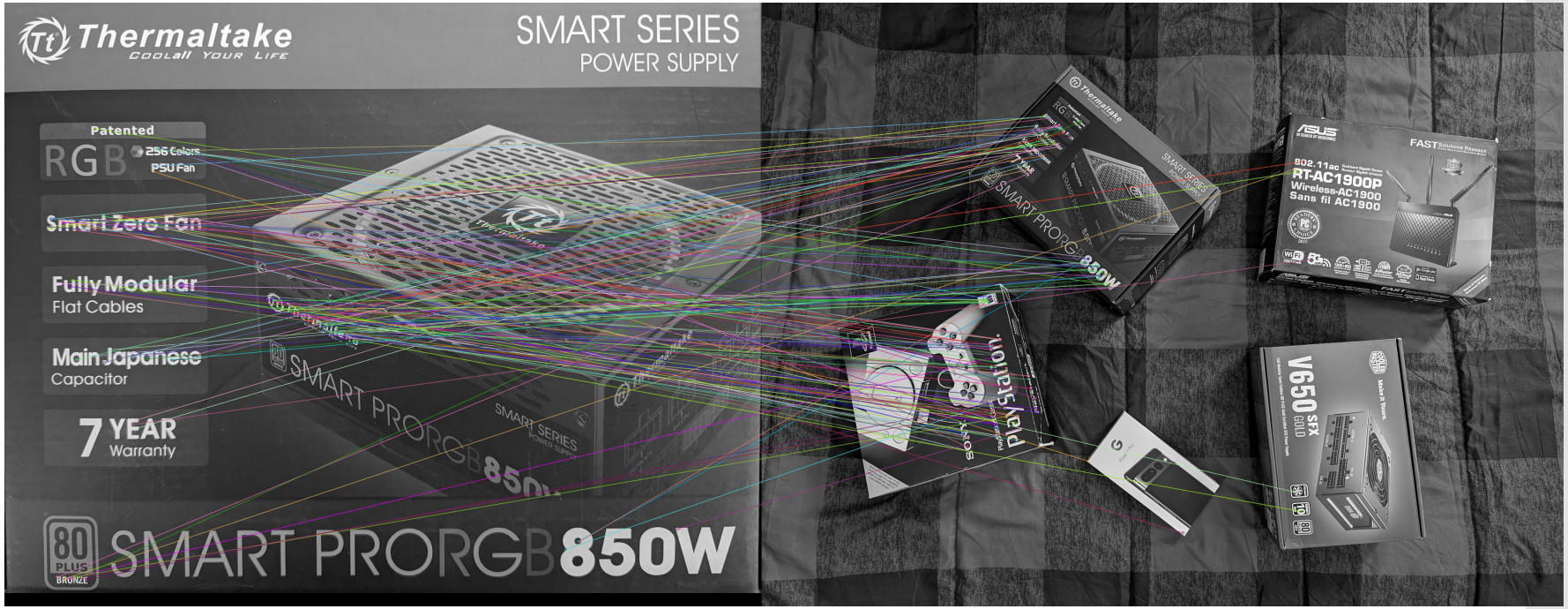
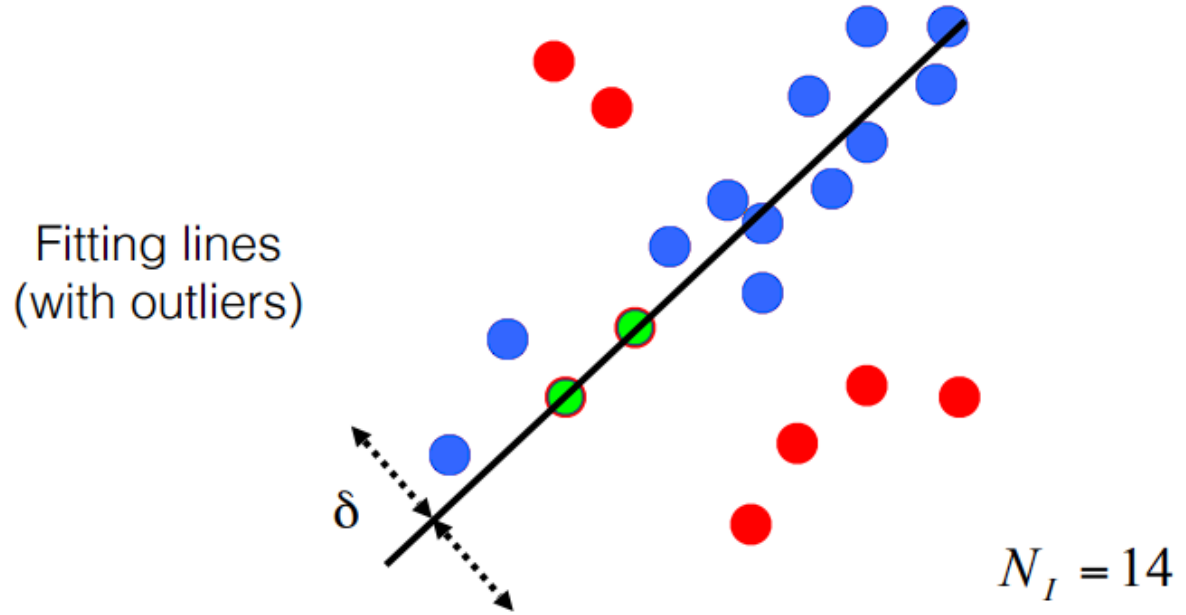


Figure 6: Results of ORB key point matching and Brute Force – based matching on the 25% size images.

RANSAC Algorithm

- Using the matched points, we compute a homographic matrix using the RANSAC algorithm and OpenCV's cv2.findHomography() function
  - RANSAC repeats an iterative process to compute a model with as many inlier points and as few outlier points as it can
  - The homographic matrix is a 3x3 matrix which represents the projective transformation from the first image to the second image

RANSAC Algorithm (Con't)



Algorithm:

1. Sample (randomly) the number of points required to fit the model
2. Solve for model parameters using samples
3. Score by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Figure 7: Pseudocode for RANSAC algorithm with diagram

[6] K. Kitani, "RANSAC 16-385 Computer Vision," Carnegie Mellon University. https://www.cs.cmu.edu/~16385/s17/Slides/10.3_2D_Alignment__RANSAC.pdf, (Accessed Mar. 2023)

RANSAC Algorithm (Con't)

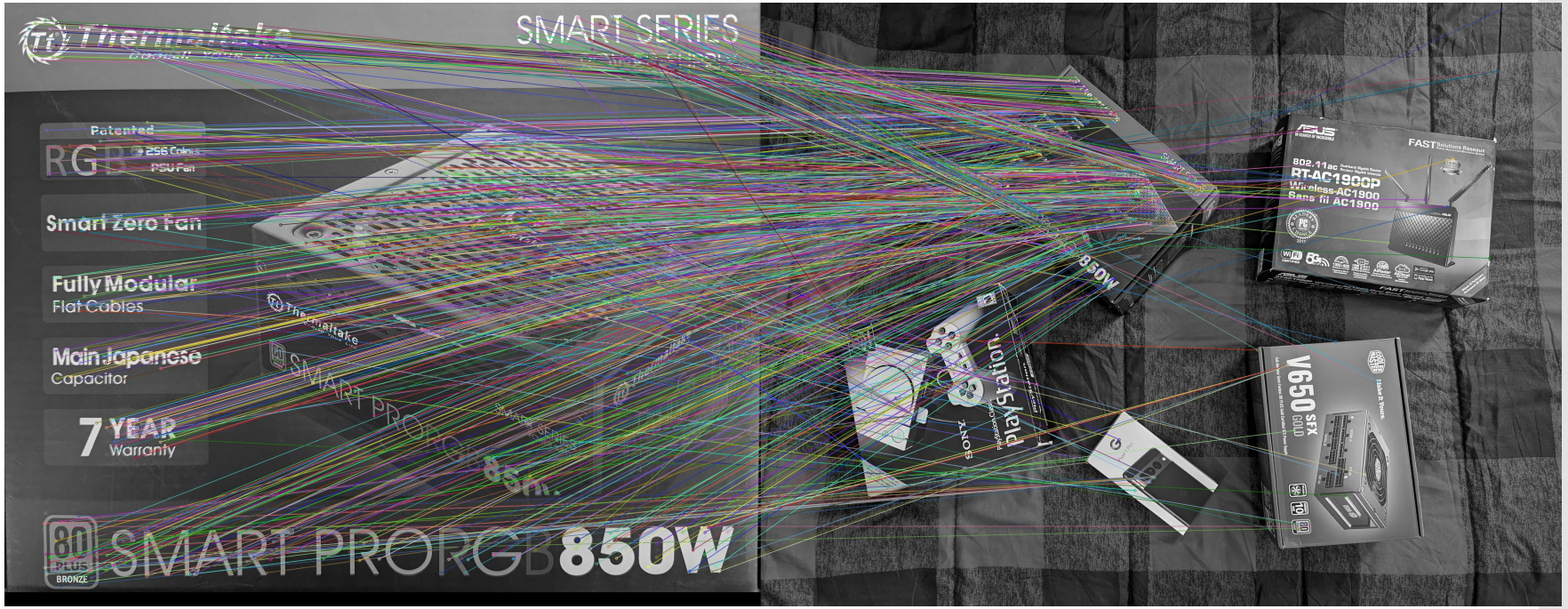


Figure 8: SIFT / FLANN matched images before RANSAC outlier removal

RANSAC Algorithm (Con't)

Figure 9: Inlier matches computed by RANSAC algorithm

Simple UI + Web Server

- Created a self-hosted flask web app to test the project
- Users can upload two images and choose a minimum number of points
- Then, clicking a button runs the algorithms and detects the objects in the path
- The two images are displayed to show the matching of key points

Challenges & Achievements

- **Challenges:**
  - Collaborating frequently was difficult because we all had very busy schedules throughout the year
  - SIFT algorithm was very computationally expensive, which limited our real time capabilities especially on large images
  - It was difficult to prevent our algorithm from detecting key points from details in the image other than the object itself
  - We had to rely on using a static mask to outline the path in our images rather than determining it based on the image
- **Achievements:**
  - Successfully implemented the obstacle detection
  - Created an intuitive user interface to experiment with our work
  - Gained valuable knowledge on the SIFT and RANSAC algorithms and the process behind detecting and matching features between two images

Demonstration

Special Thanks

- Thank you Dr. Boufama for his guidance throughout the project. He articulated his vision for the project very clearly and provided sufficient guidance as we developed the project
- Thanks to all our team members for accomplishing all our main goals for this project, despite the busyness of our school year

Any Questions?

References

- [1] "OpenCV: Introduction to SIFT (Scale-Invariant Feature Transform)," Opencv.org, 2020. https://docs.opencv.org/4.x/da/df5/tutorial_py_sift_intro.html
- [2] "OpenCV: Feature Matching with FLANN," Opencv.org, 2021. https://docs.opencv.org/3.4/d5/d6f/tutorial_feature_flann_matcher.html
- [3] K. G. Derpanis, "Overview of the RANSAC Algorithm," 2010. http://www.cse.yorku.ca/~kosta/CompVis_Notes/ransac.pdf
- [4] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," International Journal of Computer Vision, vol. 60, no. 2, pp. 91–110, Nov. 2004, doi: https://doi.org/10.1023/b:visi.0000029664.99615.94.
- [5] G. Wang, B. Rister and J. R. Cavallaro, "Workload analysis and efficient OpenCL-based implementation of SIFT algorithm on a smartphone," 2013 IEEE Global Conference on Signal and Information Processing, Austin, TX, USA, 2013, pp. 759-762, doi: 10.1109/GlobalSIP.2013.6737002.
- [6] K. Kitani, "RANSAC 16-385 Computer Vision," Carnegie Mellon University. https://www.cs.cmu.edu/~16385/s17/Slides/10.3_2D_Alignment__RANSAC.pdf, (Accessed Mar. 2023)