BILKENT UNIVERSITY

ENGINEERING FACULTY

DEPARTMENT

OF

ELECTRICAL and ELECTRONICS ENGINEERING

EEE 299

SUMMER TRAINING REPORT

Anıl Tan Aktan

22002351

Performed at

Kuartis Teknoloji

25.07.2022 - 19.08.2022

Table of Contents

1 Introduction

I did my internship at Kuartis Teknoloji, which focuses on AI-driven computer vision solutions. They work with a vast amount of data and make it discoverable, intelligible, and actionable using machine learning. They also mainly work with Turkey's Ministry of National Defence (TMND) and ASELSAN.

I decided to intern at Kuartis mainly because of the company's approach to its employees and teamwork. As for my first summer internship, I wanted to intern at a small company that integrates its interns into its projects and prioritizes communication. Kuartis treats all of its employees equally and tries to benefit from what everyone can contribute to projects so; I see this as a good learning opportunity. My second concern while choosing between companies was having a good mentor. My mentor at Kuartis was the company's Co-Founder Serdar Gedik. I see computer vision and machine learning as viable working areas for me and wanted to see how these areas operate in practice so, I was happy to have a skillful mentor who also gives lectures to other students at Ankara Yıldırım Beyazıt University.

For the internship, my supervisor told me that I will be working in both Software Technologies and Computer Vision Machine Learning departments, as I told him I wanted to experience different work areas and wanted to be a part of as many projects as possible beforehand.

During the first week of the internship, I worked on a project for Turkey's Ministry of National Defence called Veri Kovanı [1]. Veri Kovanı is a data labeling project used for all of the AI and machine learning projects of TMND. It is used to recognize humans, animals, army equipment, specific land vehicles, ships, etc. My supervisor believed every engineer who is working in machine learning should get his hands dirty with data labeling. I spent a few days learning the procedures and reading documentation regarding Veri Kovanı. I spent most of the first-week labeling and giving feedback. At the end of the week, my supervisor gave me my main task.

For the remainder of the internship, I worked on the main task. My supervisor wanted me to write a program using Python that stitches two images together to create a panorama which will be used in other image recognition projects. First, I researched and educated myself on the Computer Vision (CV) topic as I was inexperienced. Then, step by step I designed my algorithm as I learned the theory behind the CV. I created dummy problems for myself and solved them to get a better understanding of the topic. I tried to come up with better ideas and spent the last week optimizing the program to its fullest. More details about the main task and other topics of the internship will be given in related sections.

In the following sections of this report, more details about Kuartis, its projects, its working methods, and my supervisor will be discussed. Then, in the Work Done section, an in-depth look at the data labeling process and image stitching task will be taken. Following that, my performance and how I benefited from this internship will be talked about in the Performance and Outcomes section. Lastly, a summary of the

report and things I would do differently for my next internships will be talked about in Conclusion.

2 Company Information

2.1 About the company

Kuartis is a machine learning company founded in 2012. The company consists of over 40 people and they produce software, hardware, and supporting infrastructure to power large-scale intelligent solutions.

2.2 About the products and production systems of the company

Kuartis mainly works on Autonomous Driving and Object Recognition projects with Turkey's Ministry of National Defence. While a team of employees works on Veri Kovanı to train the AI with vast data, many other teams of engineers work on the software of the Recognition System and hardware of the cameras with desired properties. Kuartis uses the same data from Veri Kovanı to design Autonomous Driving Vehicles too. All teams in Kuartis have meetings daily, weekly, and monthly to give feedback and try to make things more efficient for other projects.

Kuartis also is the main contractor of the ASELSAN. Besides, they commission smaller independent projects for consumer electronics, defense, energy, and vehicle manufacturer companies.

2.3 About your department

I was part of both the Software Technologies and Computer Vision Machine Learning departments, however, Kuartis is structured to work as agile and interoperating teams according to ongoing projects. So, employees contributed to different departments when their skillset fit the project. Engineering team leaders construct their teams from a pool of interdisciplinary engineers using a matrix-based fluid organization. I generally worked together with my supervisor and other team leaders directly. My experience in the company was more mentor and learning-oriented.

2.4 About your supervisor

Name and Last Name:	Osman Serdar Gedik
Job Title:	CTO, Co-Founder
University of B.S. Graduation:	Middle East Technical University
Department of B.S. Degree:	Electrical & Electronics Engineering
Year of B.S. Graduation:	2006
Email:	osmanserdargedik@gmail.com
Phone Number:	0312 906 2214

3 Work Done

Work Done section will consist of two subsections: Data Labeling and Image Stitching. In the Data Labeling part, the process and details of data labeling will be discussed. In the Image Stitching part, the problem my mentor gave me and the steps I take will be discussed. This part will take up most space of this section as my internship resolved around the Image Stitching project and I spent most of my time working on that topic.

3.1 Data Labeling

In my first week, to make me comfortable and familiar with the company, I was tasked to work on simple data labeling tasks. I spent my first-day reading documentation and watching tutorials regarding Veri Kovanı (unfortunately all of these files are private but I will try to give as much detail as possible). Throughout my internship, I have done labeling or given feedback for at least 1 hour every day. Thereby, I have been a part of many different projects in Veri Kovanı.

Before getting into details of each different project, I want to give brief information about data labeling. In Veri Kovanı, we are given a video where we can see each frame one by one. We label each object of interest in each frame to train the AI by giving examples of valid recognition. In the figure below, an example of a frame in Veri Kovanı is shown:

Figure 1. *Data Labeling in Veri Kovanı*

In Veri Kovanı, there are many types of objects ranging from humans to specific armored vehicle models. These types change with every project in Veri Kovanı and we need to specify each object type with as much detail as possible to give rich feedback to AI. For example, if we identify an armored vehicle and we decided it is an M60 model, we should declare it that way. With each different project, I read the project's documentation to learn details about these details

regarding models of vehicles or states of humans. For example, M60 Tank has sharp edges in its front and a Leopard Tank's top is almost flat, so I paid extra attention to figure out the most detailed classification I can declare. You can see a hierarchy tree from a land vehicle project in the figure below:

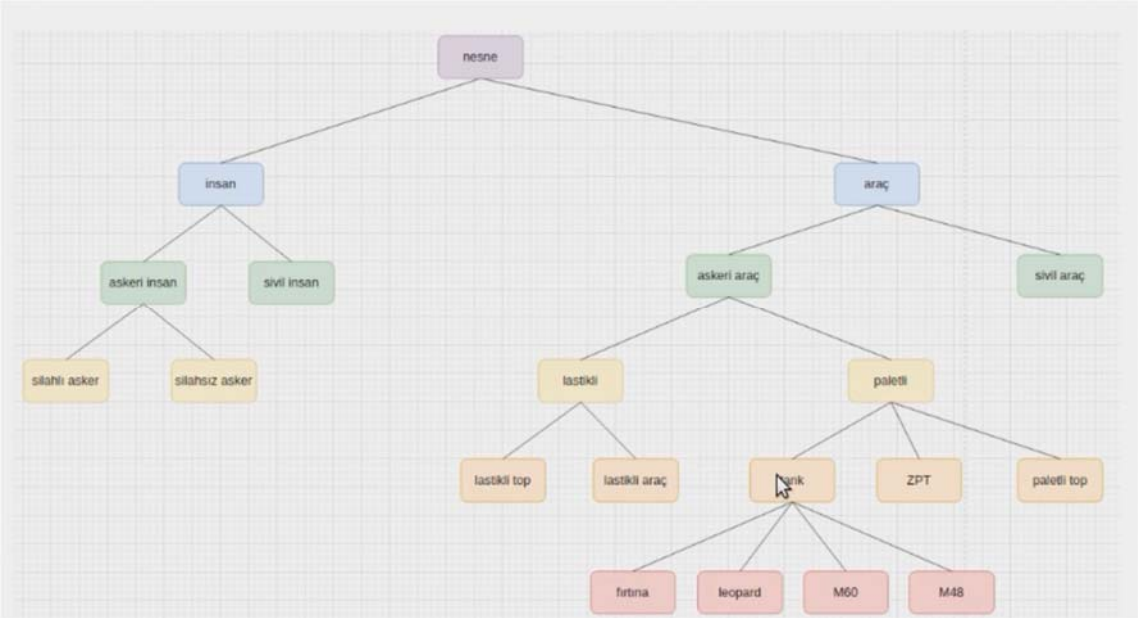


Figure 2. *Hierarchy Tree of a Land Vehicle Project*

After declaring the type of the object, we also need to consider its state of it. If an object is behind another object, so if we can't clearly see the object, we declare it as Occluded to indicate to AI that this feedback doesn't represent the whole object but a part of it.

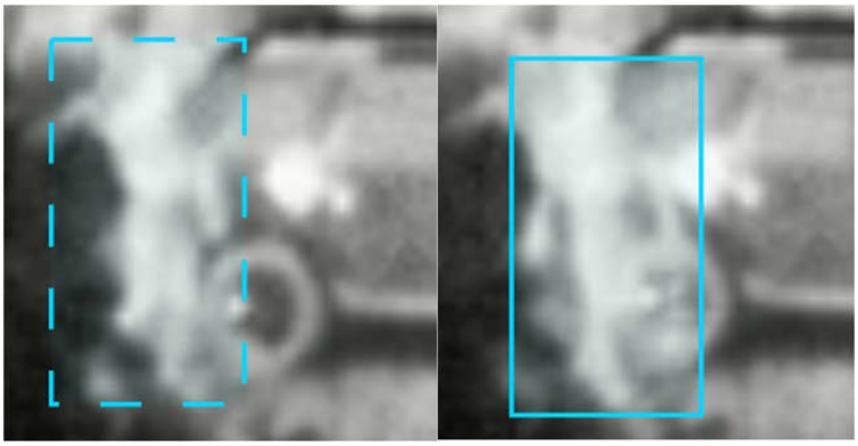


Figure 3a. *Occluded Object* ***Figure 3b.*** *Non-Occluded Object*

Besides, we also declare an object as Texture Visible or Non-Texture Visible. The human eye tends to classify an object without seeing all of its features. For example, without seeing the wheels of an object or looking at its shape we may decide that object is a car. So, to make the predictions of the Recognition AI more

precise, we indicate Non-Texture Visible labels do not contain all of the features of said object.

Lastly, we also declare a moving object as Dynamic. Knowing if an object is moving or not is very important for the AI projects that Turkey's Ministry of National Defence uses. So, we want all of the labeled data to contain this property too.

The first project I worked on was Night Time Human and Vehicle Recognition. This project was relatively easy and it mainly focused on the two states of the human figures: Full-Body and Half-Body. Soldiers were constantly changing their stance to take cover and reduce their visible area. In the figures down below, you can see the two states:

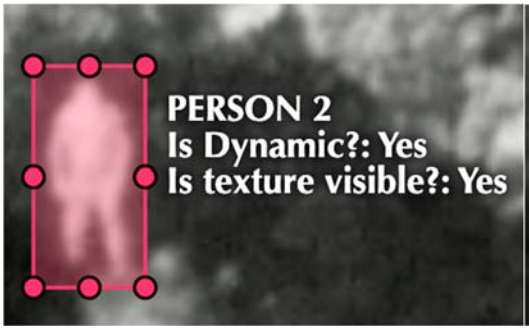


Figure 4a. *Full Body State*

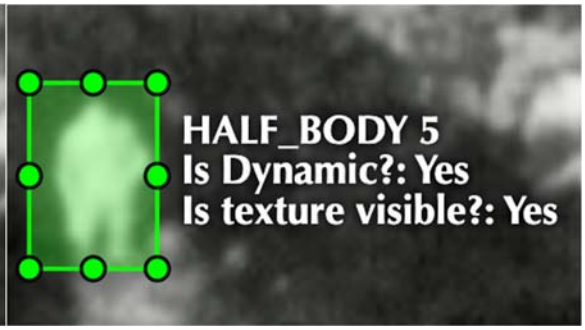


Figure 4b. *Half Body State*

The second project I worked on was Ship Recognition. It was harder as ships and the camera constantly changed their position with waves. So, when I did a mistake, it was very tiring to fix the whole labeling process. Thereby, I tried to do everything perfectly in the first place. You can see an example frame from the Ship Recognition Project down below:

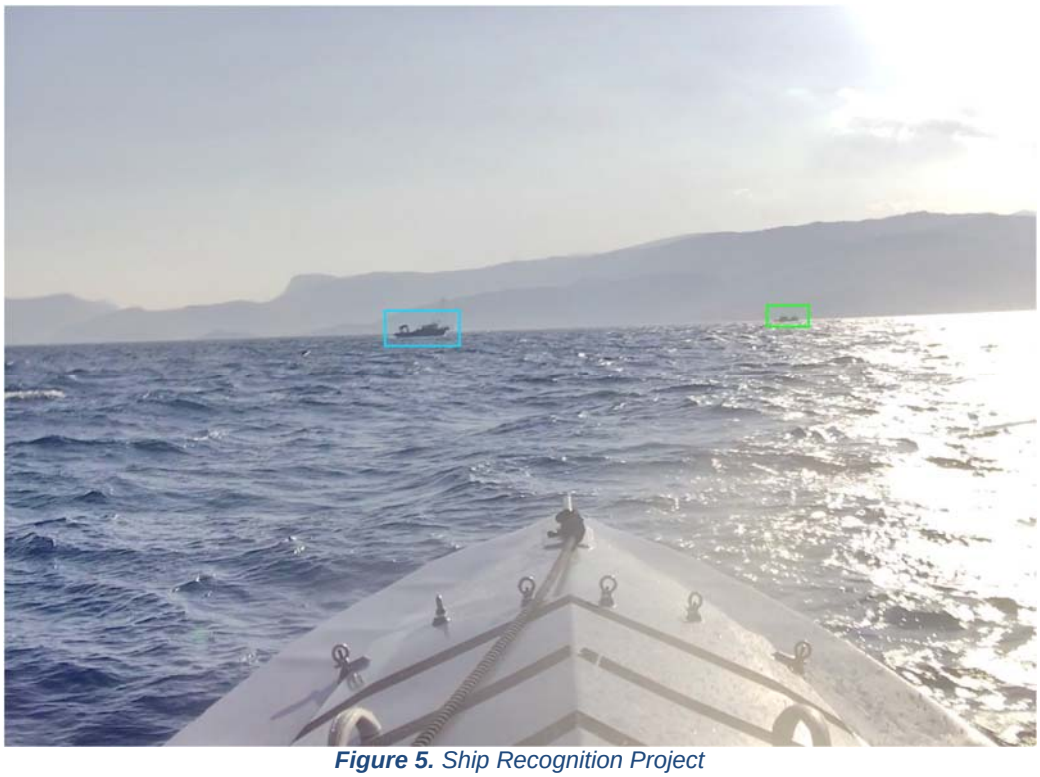


Figure 5. *Ship Recognition Project*

The last project I worked on was High-Resolution Recognition. This project contained a vast number of details as videos were captured with very high-resolution cameras and it was important to label all objects of interest. You can see an example of this detail down below where Figure 6b shows the area inside the red square in Figure 6a and a car that might be easily overlooked:

***Figure 6a**. Original Frame*

***Figure 6b**. Close up to Red Square*

This project also contained many objects very close to each other so it took a lot of time and attention to follow the movement of each object. This project was by far the hardest one but I felt more comfortable working on it because my labeling skills had improved. You can see a crowded scene example with two different frames in the figures down below:

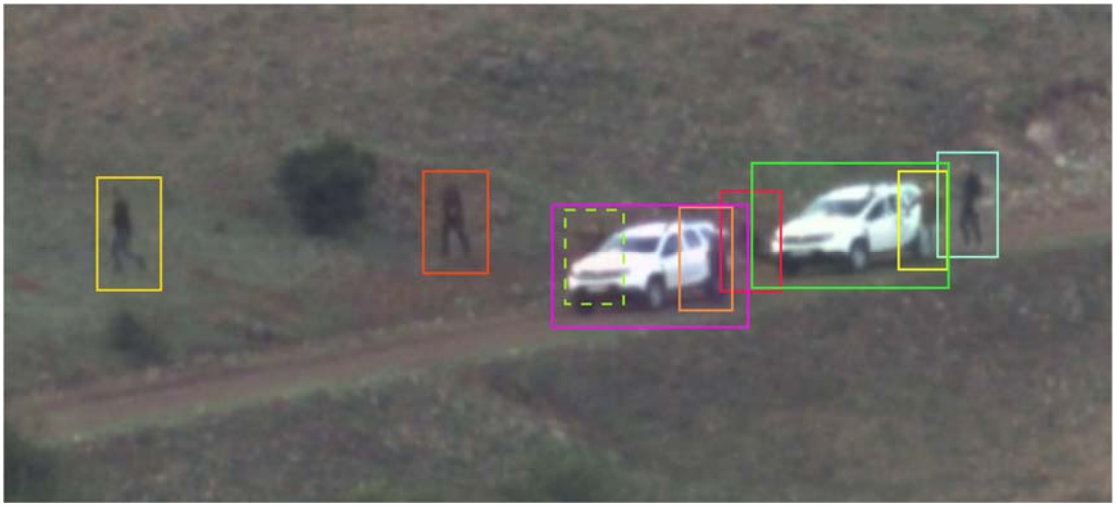


Figure 7a. *High-Resolution Recognition Scene, 1st Frame*

Figure 7b. *High-Resolution Recognition Scene, 501st Frame*

3.2 Image Stitching

At the end of my second week during the internship, my supervisor gave me my main task and wanted me to complete it before my internship finished. My task was to write a program that combines two images and creates a panorama of them if they have common areas. I have spent my second week researching the topic of computer vision and image stitching. I tried to learn the fundamentals as this was the first time that I worked on a project regarding computer vision. With every topic I get a good understanding of, I completed that part of the project and completed it step by step. I gave myself dummy problems and tried to solve them to familiarize myself with the knowledge I learned. I want to elaborate on each of the steps I took in their own subsections to explain the process of this project better and clearer. I will also discuss my algorithm with bullet points under these subsections one by one.

3.2.1 Understanding the Problem

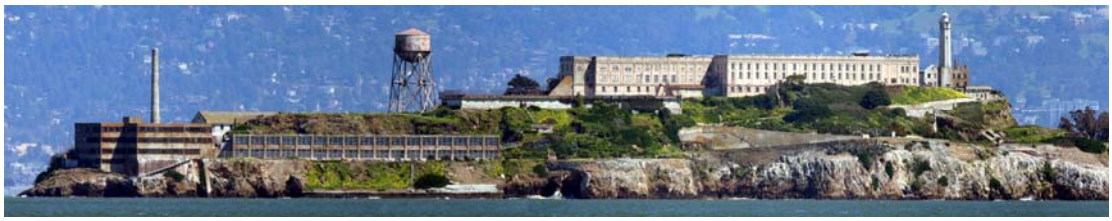


Figure 8a. *Left Image*

Figure 8b. *Right Image*

I thought about everything I know about the problem and decided what I will need to do. There are two images that have some areas matching and we need to generate a panorama image by combining them. In our case, those matching areas are in the right half and left half of the images (we will use this knowledge to optimize our code further later). After understanding the problem, I worked on my pseudo algorithm and what I needed to do so I divided the big problem into six small steps:

- Detect key points in both images.
- Match the most similar key points that we found.
- Compute the transformation needed for matched points.
- Find a method to eliminate outliers to get an accurate model.
- Use filters to blend both images so the outcome looks seamless.
- (Lastly, optimize the code to make the generation more efficient.)

3.2.2 Detecting Key Points

To detect key points in the images, I thought about turning both images into gray-scale (where each pixel has a gray level value between 0-255). I figured we could find the edges by looking at the gradients as gray-level values were changing significantly through the edges, those points had high derivatives.

At this point, I gave myself the first dummy problem as I wanted to test out and get a better understanding of different corner/edge detection techniques. The first technique I researched was Harris Corner Detection [2] and I wanted to write a standalone program that used Harris Corner Detection using Python and OpenCV library [3]. In the program,

- I converted the images into gray-scale
- I used the Harris Corner Detection technique with a threshold to eliminate low derivative points and detect the corners
- I went over all the points and drew them on the images and displayed both results on the screen

You can also find the code on my GitHub (I tried to write detailed comments for each of my programs and tried to explain what I was trying to do in each step of the code):
https://github.com/aserman01/Computer-Vision/blob/main/Corner_Detection.py

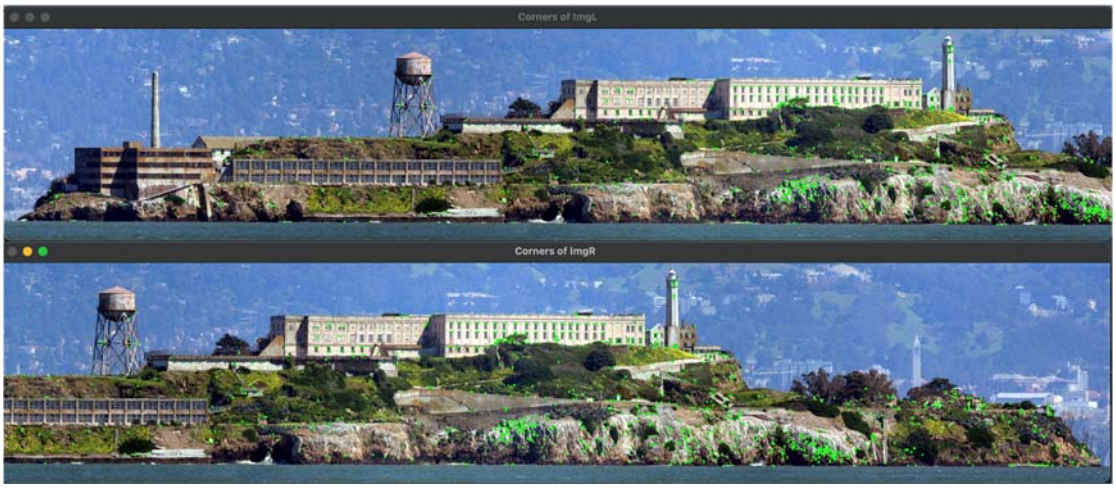


Figure 9. *Harris Corner Detection Technique Result*

After writing the program, I felt more comfortable getting into the next step and I gave myself the second dummy problem. I wanted to write a program that used SIFT (Scale-Invariant Feature Transform) [4] to detect features and match them. I also used the same SIFT algorithm in my main code. In the program,

- I converted the images into gray-scale
- Using SIFT, I created descriptors for right and left images and detected all of the features in both images
- I used brute force to match the points by checking each matches distances and finding a pattern
- I eliminated false matches by only using the matches with the shortest distances
- I drew the matches on screen and displayed both images together

You can also find the code on my GitHub:
https://github.com/aserman01/Computer-Vision/blob/main/SIFT_Matching.py

Figure 10. *Image of Best 100 Matches Detected by SIFT Algorithm*

3.2.3 Feature Matching

In the main code, after detecting features in both images using SIFT algorithm, I used a technique called kNN (k-Nearest Neighbour) [5] to match the

most similar points and group them under arrays using Python's NumPy [6] library. In the feature matching part of the program,

- I take all the matches we found using SIFT and create an empty list ("good_points")
- I check every two matches' distances using the kNN technique and if the points lie in our desired threshold, we append them to "good_points"
- I also draw these matches on the images and display them

You can also find the main code (and full resolution pictures) on my GitHub: https://github.com/aserman01/Image-Stitching-OpenCV/blob/main/ImageStitching.py

Figure 11. *Image of 100 Matches Found by kNN*

3.2.4 Transformation Matrix (Homography)

I had found and matched the features in both images, I needed to calculate a transformation matrix for these points. When I did my research, I found that a 2D-to-2D transformation which may involve warping and moving needs at least a 3x3 matrix for its transformation, this matrix is also called the Homography Matrix. We can calculate the Homography using a minimum of 4 points and luckily these images had over 1000 matches. After using OpenCV's findHomography [7] function with key points in both images, I got my needed 3x3 matrix.

3.2.5 Better Estimation (RANSAC)

When I was calculating the Homography, I also looked into techniques to get a better estimation and I come across RANSAC (Random Sample Consensus). By using RANSAC Method, we find and use the subset with the most inliers and by doing so eliminate the outlier points and matches.

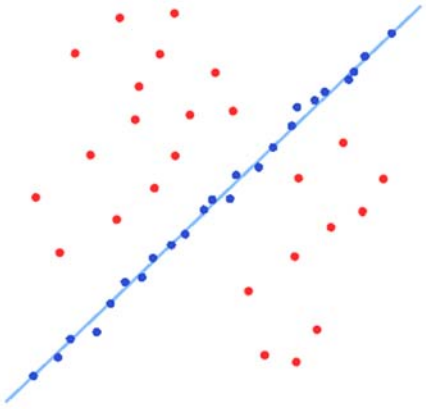


Figure 12. *An Example of the Subset with Most Inliers*

3.2.6 Warping and Blending

Lastly, I used weighted filters to make the transition between both images natural as images could have different lightning or colors and there may be sharp lines in the middle of the panorama. After combining both images, we got our panorama generated.

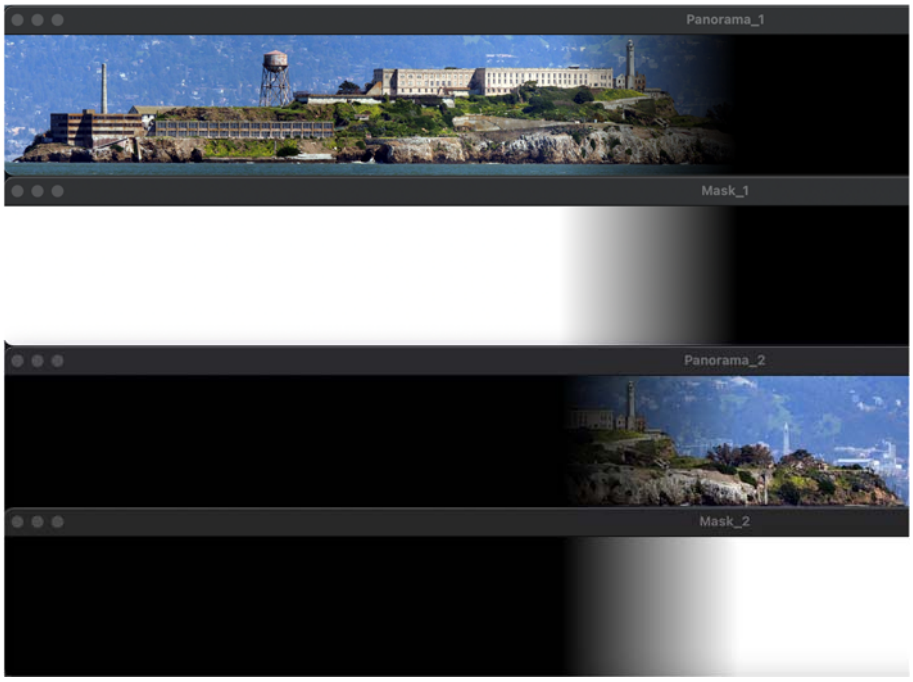


Figure 13. *Images Together with Their Weighted Filters*

Figure 14. *Output Panorama of the Program*

3.2.7 Optimization

After completing the program, I spent my last week adding life quality improvements and optimizing my code. For better optimization, I tried two different methods and in the end, I decided to use both of the methods together.

3.2.7.1 Rectangle Masking with Percentage

The first method came to my mind very early on in the project. As we know which image is left and which image is right, we can scan the %75 of both images' right and left parts, that way program can work in a more optimized manner. So, I created a mask to force the program to work only in the white section of the mask.

My test results showed that scanning only %75 of the images helps us save 2-3 seconds for each stitching and this value still can increase as we reduce the scan area without losing any details in the panorama.

Figure 15. *Rectangle Mask, White Area Indicates Scanned Area*

3.2.7.2 Bucketing

After explaining to my supervisor my idea about using the first method I found for optimization, he gave me more ideas about how I can optimize my code further and so I implemented this second method.

We can separate our image into little rectangles and can only take some of those rectangles to save computing time. As these rectangles are homogenously disturbed through our image, the precision of the stitching doesn't change. I created again a mask with 50x50 pixel squares and 50x50 black squares resembling a chess board.

My test results showed that using a 50x50 pixel mask of the images helps us save 4-5 seconds (which is very drastic) for each stitching. This method works better as we only scan the %50 percent of the image and also this method can be used in any project in which we don’t know which image is right or left.

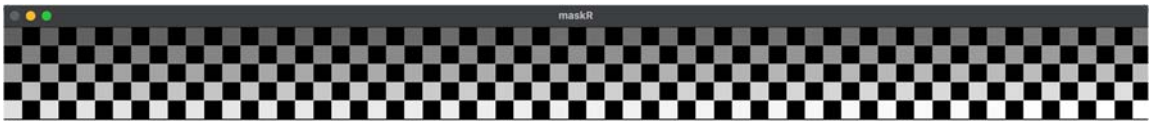


Figure 16. *Bucketing, White Area Indicates Scanned Area*

4 Performance and Outcomes

4.1 Solving Complex Engineering Problems

I wanted my project to work as efficiently and as accurately as possible. After working in Kuartis, I realized these were also the main concern of the engineers in this field. I spent my last week working on optimization and blending. I didn’t want to put the project behind; I wanted it to work to its fullest. I discussed these concerns of me with my supervisor and tried different methods. In the end, I am very happy and very surprised with the outcome. In the beginning, I didn’t even think that I could complete writing the main code.

Also, while working on the detection of the key features, matching, and calculating homography, I had to determine many thresholds and variables. I determined some of these numbers by trial and error but mostly I did extra research on that exact topic and read many articles. I have used many of my values according to a study by Caparas, Fajardo, and Medina about Feature Based Image-Stitching [8].

4.2 Recognizing Ethical and Professional Responsibilities

When I was researching feature detection methods, I came across many, and before deciding on using SIFT I wanted to use SURF (Speeded-Up Robust Features) algorithm [9] as I found out that it was faster. But I had to use SIFT algorithm to prevent myself (and the company) from any copyright issues as I had the license of the SIFT algorithm but didn't have the license of the SURF algorithm in the OpenCV library.

Also, while I was working on the blending section of the main project, I came across an open-source function code of the user "linrl3" [10] that fit perfectly for my project. So, I used the function he wrote in my project and I gave credit to him and cited his GitHub account in my code and my own GitHub account.

4.3 Making Informed Judgments

While I was working on labeling projects, I also gave feedback on other people's labeling documents and sometimes corrected their mistakes. These feedbacks required making many judgments and I filled individual reports for each of them. For example, one time, one of the labeling documents was so poorly done, that I had to think thoroughly about what action should be taken. One of my colleagues advised me to correct each error one by one by changing each label but I decided to delete the entire document and start over. I learned doing things perfectly in the first place makes things work faster and smoother while working on the Ship Recognition project. By doing everything from scratch I completed the assignment in a very surprisingly short time, I am sure it would be harder to find every error and change the label error with the label it should have been.

Further, in the main project, after implementing the bucketing method for optimization, my supervisor thought it was all we could do and it was enough. I decided to use both methods of optimization together and told him I can implement it in a short time. By doing so we saved 6-7 seconds for each picture.

4.4 Acquiring New Knowledge by Using Appropriate Learning Strategies

My internship consisted of me learning new things and applying them in practical ways as a whole. To learn a topic better, I created myself dummy projects and I believe they helped me to learn the topics very well also started my brain, and new ideas for the main project started to pour in. I read many documents regarding libraries and their sub-functions that I used [2, 3, 4, 5, 6, 7, 8, 9]. I took some lectures from my supervisor about the fundamentals of computer vision and took notes of every step of the project. I also checked other people's projects on GitHub [10] and get many inspirations from them.

4.5 Applying New Knowledge As Needed

While working on the main project, I divided the project into little parts and completed each one by one with the new knowledge I acquired. I didn't stop my research with only problems I came across, I also researched many ways to improve the project and tried to implement every new technique I came across.

4.6 Diversity, Equity, and Inclusion (DEI)

Kuartis is a very welcoming company. Every employee works in a big open area where even the founders are next to you without any walls separating any employees. There are personal offices and meeting halls anyone can use freely. As an intern, I sat next to other employees and worked with them under the same conditions. This welcoming feeling is also present in the projects. Each new idea is welcome and even encouraged.

When I talked with HR, they told me they prioritize gender equality in their job applications. She also told me that they try to hire interdisciplinary people so their employees can feel comfortable and they can be a part of any department, she told me transfers between projects and departments are very flexible for the sake of the employee. Lastly, she told me they use reports and reviews to evaluate their employees' performances to prevent any injustice evaluation.

5 Conclusions

I am very happy that I did my internship at Kuartis, it was a very fun and educating experience. However, there are some things I would do differently for my future internships. I would try to include myself more. I got the courage and confidence to include myself in bigger projects only in the latest weeks and didn't share my ideas as comfortably before.

I got many experiences working in a small company and working in the field of computer vision. I am very happy to work in the field of computer vision and machine learning. I hope I will get more experience in these fields in the future.

Lastly, I want to talk about one thing I found very important was the ability to teach myself new things. Bilkent helped me a lot about this and I see its importance just now. I have come over many problems because I was able to teach myself.

References

[1] “Veri Kovanı”.
https://verikovani.ssb.gov.tr/home.html. [Accessed: Sep 9, 2022]

[2] “Harris Corner Detection”
https://docs.opencv.org/3.4/dc/d0d/tutorial_py_features_harris.html [Accessed: Aug 1, 2022]

[3] “OpenCV Introduction”
https://docs.opencv.org/3.4/d1/dfb/intro.html [Accessed: Aug 1, 2022]

[4] “Introduction to SIFT (Scale-Invariant Feature Transform)”
https://docs.opencv.org/4.x/da/df5/tutorial_py_sift_intro.html [Accessed: Aug 2, 2022]

[5] “Understanding k-Nearest Neighbour”
https://docs.opencv.org/3.4/d5/d26/tutorial_py_knn_understanding.html [Accessed: Aug 3, 2022]

[6] “NumPy documentation”
https://numpy.org/doc/stable/# [Accessed: Aug 3, 2022]

[7] “CV2.findhomography: Things You Should Know”
https://www.pythonpool.com/cv2-findhomography/ [Accessed: Aug 5, 2022]

[8] “Feature-based Automatic Image Stitching Using SIFT, KNN and RANSAC”
https://www.warse.org/IJATCSE/static/pdf/file/ijatcse18911sl2020.pdf [Accessed: Aug 5, 2022]

[9] “Introduction to SURF (Speeded-Up Robust Features)”
https://docs.opencv.org/3.4/df/dd2/tutorial_py_surf_intro.html [Accessed: Aug 2, 2022]

[10] “GitHub of Runlong Lin”
https://github.com/linrl3 [Accessed: Aug 9, 2022]

Appendices

I collected all of the codes I used in my GitHub profile and added detailed comments for easy understanding:

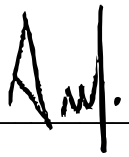
- Codes of the dummy problems (Harris Corner Detection and SIFT)
https://github.com/aserman01/Computer-Vision

- Main Image Stitching Code and Pictures
https://github.com/aserman01/Image-Stitching-OpenCV

Self-Checklist for Your Report

Please check the items here before submitting your report. This signed checklist should be the final page of your report.

- ☑ Did you provide detailed information about the work you did?
- ☑ Is supervisor information included?
- ☑ Did you use the Report Template to prepare your report, so that it has a cover page, has all sections and subsections specified in the Table of Contents, and uses the required section names?
- ☑ Did you follow the style guidelines?
- ☑ Does your report look professionally written?
- ☑ Does your report include all necessary References, and proper citations to them in the body?
- ☑ Did you remove all explanations from the Report Template, which are marked with yellow color? Did you modify all text marked with green according to your case?

Signature: 

While writing your summer internship reports, you should follow the rules of ethical writing. You can find an extensive guide on ethical writing at:

https://ori.hhs.gov/content/avoiding-plagiarism-self-plagiarism-and-other-questionable-writing-practices-guide-ethical-writing