**Vertiport Layout Optimization**

Submitted by: Abdulbaqi Almuzaini

Course: ENGR-4900

Instructor: Professor Sangho Shim

Date: 04/23/2025

**Vertiport Layout Optimization Report**

|  |  |  |
| --- | --- | --- |
| **Section** | **Content** | **Page** |
| 1. Introduction | Overview of project purpose and learning goals | 1 |
| 2. Project Objective | Defines the layout optimization goal | 2 |
| 3. How the Program Works | Step-by-step explanation of the code logic | 2 |
| 4. Outputs | Output types and component chart (Figure 1) | 3 |
| 4.1 Results in Excel | Excel results and pie chart (Figure 2 & 3) | 4 |
| 5. How to Use the Code | Setup and run instructions | 5 |
| 6. Example Usage Scenario | Real-world application example | 5 |
| 7. Value and Marketing Potential | Benefits and potential users | 6 |
| 8. GitHub and Sharing | Availability of code and examples | 6 |
| 9. Conclusion | Reflections and final thoughts | 6 |

**1. Introduction**

This report describes a Python program created as part of a college assignment. The goal of the code is to design a vertiport layout that places different aviation components (like FATO pads, stands, terminals, and taxiways) on a grid while staying within a budget and maximizing the number of vertiport operations. I used the Gurobi solver to create this model. The report also explains how to use the program and the potential value it offers.

**2. Project Objective**

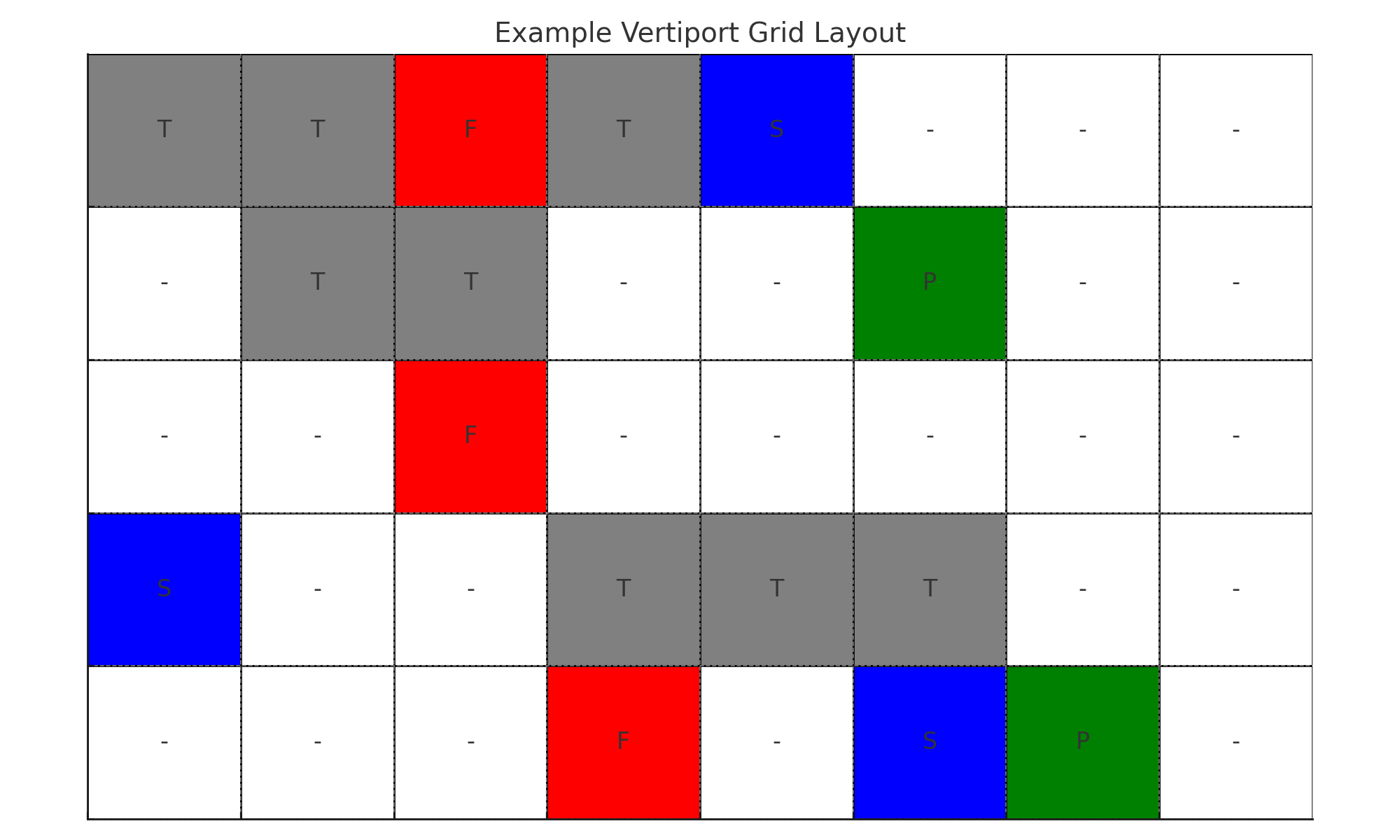
The objective of this project is to find the best layout of components that can handle the maximum number of aircraft operations per hour. The placement must follow safety rules, budget limits, and connection constraints. The program creates a grid and automatically selects where to place the components using mathematical optimization.

**3. How the Program Works**

- The grid size is set by the user (for example 5 rows by 8 columns).  
- The user defines the costs for each component: FATO (F), Stand (S), Taxiway (T), and Terminal (P).  
- The program uses the Gurobi solver to place the components so that:  
 - Only one component is placed in each cell.  
 - F, S, and P must be next to a Taxiway.  
 - FATO pads must not be in the center of the grid or too close to each other.  
 - All components are connected in the grid using flow logic.  
 - The total cost must not exceed the available budget.

**4. Outputs**

Once the program runs, it prints and saves:  
- The number of operations supported by the layout.  
- A grid layout showing where each component is placed.  
- The total count of each component.  
- The total budget used.  
- These results are saved into two files:  
 **- layout.csv**: shows the grid layout.  
 **- summary.csv**: includes totals and performance values.



**Figure 1**: Example output showing component layout (F: FATO, S: Stand, T: Taxiway, P: Termina.

**4.1 Results in Excel**

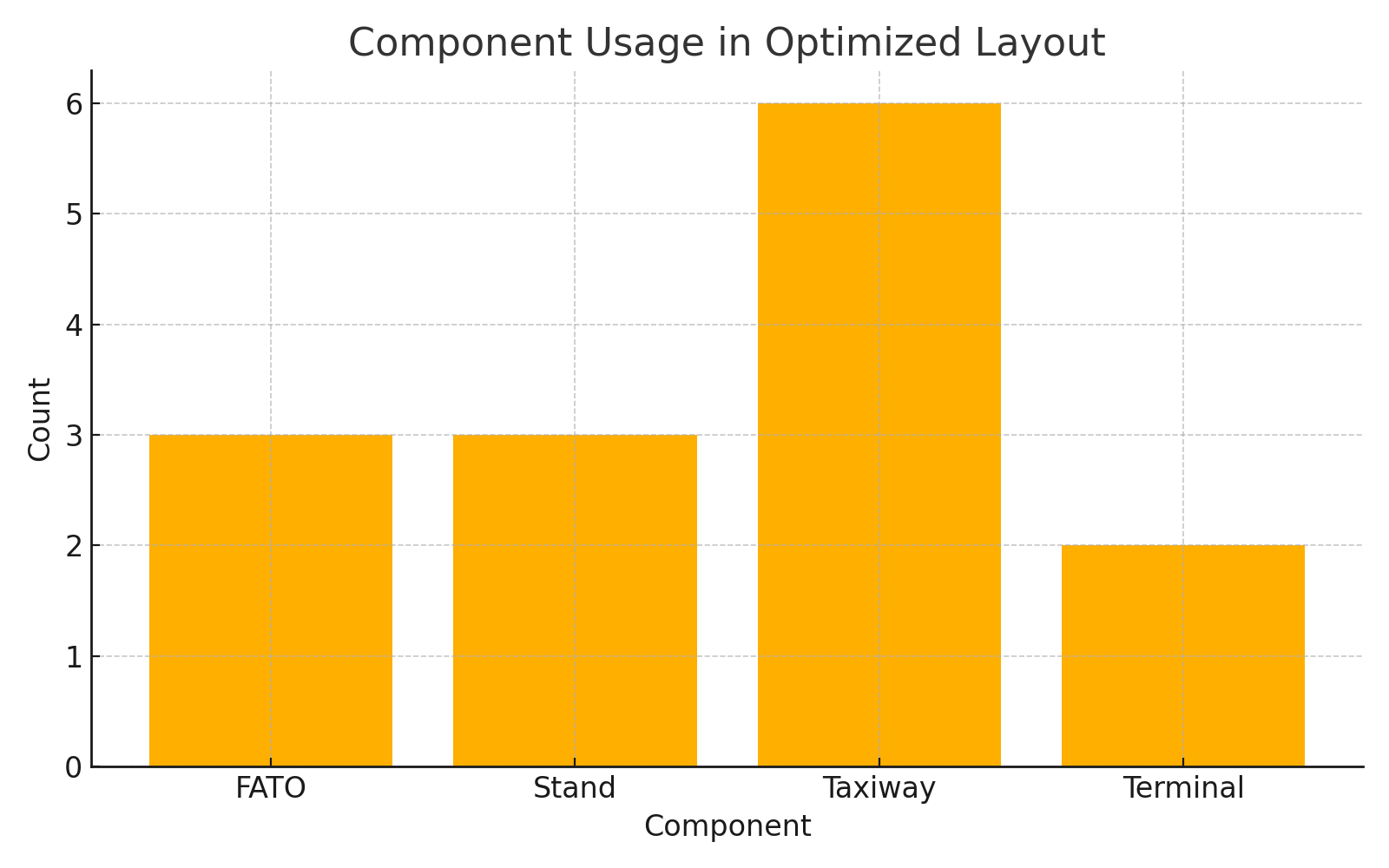
After running the program, two CSV files are created:

* **layout.csv:** shows where each component (F, S, T, P) is placed in the grid.
* **summary.csv:** shows how many of each component was used, how many operations the layout can handle, and the total budget used.

You can open these files in Excel or Google Sheets to review them easily. They are useful for analyzing and presenting the results visually.

* These charts below were created from the .csv files using Excel.
* The pie chart below shows how the budget was distributed in the optimized layout, comparing the portion that was used with the total available budget

**Figure 2:** Budget Usage vs Limit based on layout summary.

**Figure 3**: Bar chart showing the count of each component in the optimized layou

**5. How to Use the Code**

To run the code, follow these steps:  
1. Install Python and the Gurobi Optimizer.  
2. Save the Python file as **vertiport\_model.py**.  
3. Open the Command Prompt.  
4. Navigate to the folder where the file is saved using cd.  
5. Run the command:  
 **python vertiport\_model.py**  
6. Once it finishes, open layout.csv and summary.csv in Excel to review the results.

**6. Example Usage Scenario**

Let’s say a city wants to build a vertiport. They need to know where to place terminals, landing pads, and other facilities on the available land. This program allows the planner to try different grid sizes, budgets, and costs to find the most efficient layout. It shows how many operations the layout can support and makes sure the layout follows safety and access rules.

**7. Value and Marketing Potential**

This program can be used by:  
- Urban planners who want to simulate vertiport designs.  
- Students learning optimization and Python.  
- Researchers working on smart city or air mobility projects.  
  
It is easy to use, requires only basic Python knowledge, and provides useful visual outputs in CSV format. It can also be extended to include more features like 3D visualization or interactive layout editing.

**8. GitHub and Sharing**

The full code and example files can be uploaded to GitHub for easy access. Users can download it, test different values, and adapt it to their own needs.

**9. Conclusion**

This project helped me learn how to write an optimization model using Python and Gurobi. I understood how to apply rules and logic to control the layout and how to save and share the results. The program is useful for basic planning and can be a good starting point for anyone working on similar transportation or logistics problems.