Augmented Reality Tracking in Air Traffic Control (ATC) Operations

Jason D. Lazar
Department of Computer Science, Stanford University
jasonl24@stanford.edu

Abstract

It is well established that the aviation industry relies on a robust air traffic control (ATC) network to maintain safe, redundant, and effective operations both on the ground and in the air. However, one of the greatest barriers to safety at major airports is extreme, low-visibility weather conditions. Thus, it is imperative to begin implementing and integrating systems to increase aviation safety when operating in suboptimal conditions. To do so, this paper introduces a relatively simple, yet effective computer vision approach to implement a real-time virtual display that will help aircraft controllers identify aircraft from the control tower, in addition to their radar screens. By using radar data provided by Flight Radar 24 [4], I manually extracted top-down coordinates with respect to a bounded region. Subsequently, I manually defined correspondences from a static alternate perspective of the aircraft, and computed an optimal perspective homography H^*, allowing the system to effectively track an aircraft in a toy example. Finally, I conducted the same methodology on dynamic footage in low visibility scenes, computing a homography matrix for each frame $H_1, ..., H_n$ and transforming each radar coordinate. Since the system is built on ground truth radar data, I achieved 100% tracking accuracy for both static and dynamic cameras on real-world footage. Compared to openCV's mean point algorithm for object tracking and CNN-based motion detection, my system worked incredibly well while the other systems did not work when introduced to Gaussian noise and occlusions.

1. Introduction

Currently, air traffic controllers in control towers are trained to implement low visibility procedures (LVP), in the event of CAT II/III conditions. That is, approaches, landings, and take-offs with a visibility range below 550 m. They often rely on reduced visual cues in tandem with radar screens to determine aircraft positioning at the airport and to coordinate multiple aircraft at once. Furthermore, ground controllers overseeing parking and takeoffs have to rely on visual aid to coordinate multiple aircraft taxiing on the ground simultaneously. However, as seen with the disaster at Tenerife, this becomes extremely difficult in low visibility conditions. Relying without any real-time visual aid increases the risk of a ground collision, or worse, an approach collision due to planes crossing a runway on accident at airports such as San Francisco (KSFO). Thus, it is imperative to implement systems to increase aviation safety when operating in sub-optimal conditions. Currently, the FAA has not implemented or mandated any form of visual assistive technology in ATC operations. Most of the time, controllers rely on their vision and looking down occasionally at their radar screen to direct landings and takeoff clearances.

2. Related Work

2.1. Augmented Reality in ATC Operations

The concept of using computer graphics to augment visual reality for ATC towers was initially proposed by Lloyd Hitchcock at the FAA Technical Center in the 1980s. [3] By 1970 and 1980, there were many successful demonstrations of heads-up displays (HUD) in aircraft cockpits to aid pilots, but not ATC operators. The idea for this system has been discussed before. In 2006, a group of NASA researchers presented the idea of software designed to aid controllers in their day-to-day operations. However, they explicitly stated *"This paper does not present a finished design for a tool ready for implementation"* [3]. Researchers from SESAR's RETINA Project attempted to tackle this problem in 2018 using simulations. However, they also stated *"... since this technology is not yet mature enough for full deployment in a safety critical environment, further research is required to demonstrate it in a real environment."* [2]. Further, SESAR's Netherlands Aerospace Center started their efforts to expand on RETINA starting in 2021. However, their research has not been completed as of 2024, nor implemented yet on real footage but rather on simulation.

Another relevant discussion of this problem is Brian

Hilburn's research [6], where he discusses notions of *head down time* (HDT) of controllers, and solutions to reduce this such as the NOMAD augmentation system to implement the system discussed in this proposal (A thorough review of this research paper is recommended to understand the issue being discussed). However, while Hilburn suggests such a system, there is no mention of what the software would look like in reality. It is clear that this idea has been thought about, but little research has been conducted to begin the implementation of such a system on real world footage.

2.2. Augmented Reality

A key component of this research also relies on augmented reality. That is, the integration of software into a real-world, 3D environment. Object tracking has been a task defined by the industry for many years. In 2014 (known as the deep learning era for object detection), Girshick et al. proposed the *Region-based Convolutional Neural Network (RCNN)*, which passed proposed regions into a CNN, extracting features and effectively identifying objects in a camera frame [5]. Many rely on deep neural networks to detect objects in the real world. However, this relies on having direct visual contact with the object. Thus, one can envision a system that leverages the accuracy of deep neural networks to identify aircraft in optimal conditions but then uses radar data to continue identifying aircraft even when you can't see the tarmac.

This project serves as an initial implementation of an object tracking system that can handle occlusions and track when aircraft cannot be seen. The intent is for future researchers to build on the approaches laid out in this initial implementation, and come up with a way to integrate hardware (such as HUD glass or spatial computing headsets) to implement this into ATC operations.

3. My Approach

3.1. Overview

To begin implementing a system that provides visual aid to ATC operators, we must simplify the problem as much as possible, and then gradually increase the complexity of the system. While many would propose the use of complex object tracking algorithms making use of neural networks, or highly advanced 3D reconstruction techniques, these approaches all rely on visual contact with the object for tracking.

Additionally, since controllers have real-time access to radar positioning data, it is evident that this data can be leveraged to track aircraft on the tarmac. At a high level, I first implemented the system using a toy example of 391 frames, where a Boeing 737-9 was conducting takeoff role, remaining on the ground for the full video duration. After implementing this system, the next step was to expand the approach to a dynamic camera, simulating head movement for further implementations using vision goggles that controllers can wear.

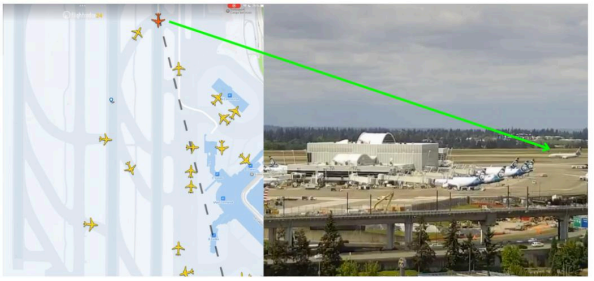


Figure 1. Side by Side, preprocessed video clips consisting of 391 frames. Timing corrections were implemented based on visual cues and GPS coordinate data.

3.2. A Static Toy Example with 391 frames at Seattle - Tacoma International Airport (KSEA)

As mentioned previously, I first considered a toy scenario with a single runway (Runway 34R/16L at KSEA), and a single aircraft (Boeing 737-9, ASA653). I first defined a coordinate plane, which was simply the pixel coordinates of the real world video. I then defined a bounded region within this coordinate system to focus on for tracking. Importantly, because of how I defined the coordinate system, I had to ensure that I preprocessed both the radar footage and the real-world footage to match frame rate, frame number, correct timing, and same dimensions. (see Fig. 1).

By using an open source, live video feed of Seattle-Tacoma International Airport, in tandem with real-time flight data by Flight Radar 24, I was able to pre-process both screens efffectively. (see Fig. 2).

Collecting 3D Coordinates with Respect to a Bounded Region

Each video consisted of 391 frames, and the first task was to obtain the coordinates of the aircraft in the flight radar video for each frame. Though this process could have been done manually, I used OpenCV's cv2 library to object track and record the 3D coordinates (x, y, h) of the aircraft on the radar. After minor adjustments of the output from the object tracking algorithm, I was able to obtain

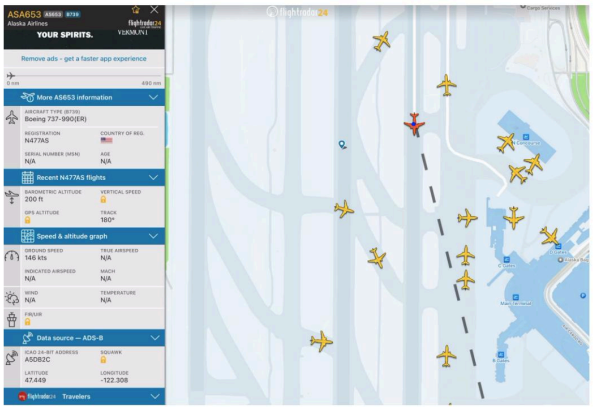


Figure 2. This figure shows two key aspects. First, the region I chose to bound for tracking the aircraft. In other words, future implementations would be able to track all aircraft that enter this region. Second, this figure shows a blue dot in the center of the target aircraft. This was a result from the object tracking coordinates from OpenCV's `cv2` Python library, and it followed the aircraft with 100 % accuracy for all 391 frames after minimal offset applied

accurate coordinate data in real-time, serving as ground truth. It is important to note that since the aircraft remained at flight level 000 for the duration of the video, the altitude component $h = 0$ for all 391 frames. Thus, I was able to discard h from the coordinates during the computation and transformation in the toy example.

Figure 2 demonstrates the accuracy of the coordinates. A blue dot that following the path of the coordinates perfectly matches the motion of the orange target aircraft during the takeoff procedure.

Transforming Ground Truth Radar Data into Real World Footage

Subsequently, the task was to take the sequence of 391 2D coordinates and calculate a perspective transformation that mapped to the tower's view of the same aircraft performing a takeoff role. This was relatively straightforward.

Consider Figure 3, which depicts a subset of the correspondences chosen. Given at least four correspondences, I was able to calculate a 3x3 homography matrix H using `OpenCV` that transformed the set of coordinates to track the aircraft from a different angle. This required immense experimentation and different correspondence choices, solving many different systems of equations to compute the optimal H. To streamline this process, I wrote software that allowed me to manually click correspondences based

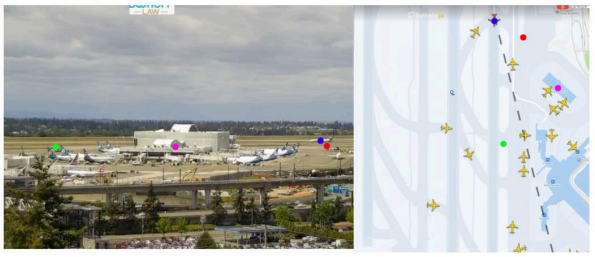


Figure 3. Approximation of four correspondences used to calculate a perspective transformation in order to effectively track an aircraft on takeoff roll.

Figure 4. The two examples I chose from the plane spotting video. Notice how they are low visibility conditions. Further, the videos have minimal movement and maintain the aircraft in a center position for simpler computation purposes

on visual markers at the airport, terminal buildings, the aircraft itself, and other features relating to the video. The choices for computation had immense effects on the video output, and thus significant experimentation was required.

3.3. Testing the Approach on Dynamic, Low-Visibility, Real World Footage at San Francisco International Airport (KSFO)

Once an optimal homography H^* was computed on an example where the camera remains static between frames, the next step was to compute multiple optimal homographies $H_1^*, ..., H_n^*$ for examples with n frames, and transform each respective radar coordinate to effectively track aircraft in low visibility conditions. I chose two examples taken from HD Melbourne Aviation's plane spotting video at KSFO (see Fig. 4) [1]:

1. Night Landing of a Boeing 777-3 Turkish Airlines (TK80), $n = 584$ frames
2. Foggy Landing of a Boeing 757-2 Delta Airlines (DL1384), $n = 534$ frames

Obtaining the Radar Data

To obtain the radar data for both flights and to have the ability to time sync the footage to the real-world video, I once again used FlightRadar24's historical data feature, which allows users to obtain radar data for specific aircraft at any time for the past 2 years.

Video Pre-processing

Once again, I had to preprocess both pairs of videos to ensure that the frame rate, timing, number of frames, and size was consistent to ensure the functionality of the implementation. This was done using video editing software Final Cut Pro X.

Differences in this Approach Compared to the Toy Example

For the most part, most of the implementation remained the same, except with increased complexity. For example, the altitude component h became did not remain the same, since the aircraft examples I chose consisted of both a landing and takeoff in low visibility. Thus, I used the altitude data provided by FlightRadar24 to add this 3D component to the 2D coordinates I originally computed with object tracking. No offset was required to track the 2D radar positioning footage, as the `OpenCV` object detection algorithm worked optimally. Further, my implementation now required manually defining correspondences for every frame in each example, which required immense manual effort. I was able to compute a list of 584 homographies for the first example, and a list of 534 homographies for the second example. Once this was done, the process became relatively straightforward. I applied each transformation matrix to each respect coordinate at each frame of the video, providing a different basis of transformation for each coordinate. This allowed the implementation to track the aircraft effectively leveraging the radar data.

4. Results and Evaluation

4.1. Performance on the Toy Example

Overall, the implementation worked very well on the toy example. Due to the calculated transformations, I was able to achieve 100% accuracy since I transformed the ground truth data directly for both the toy and dynamic examples. Figure 5 demonstrates the results for my implementation compared to `OpenCV` in varying conditions, including both nominal and blurred. To simulate low visibility conditions, I introduced Gaussian noise, and tested both systems. The motion tracker clearly failed, whereas my initial implementation worked flawlessly since it was based

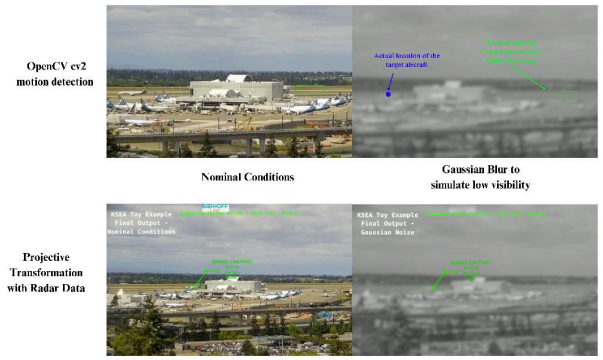


Figure 5. Comparison of the motion tracking solution, compared to my initial implementation. Clearly, motion tracking falls short and fails to track effectively

Figure 6. Shots of the final output performing on the two real-world dynamic examples. As shown, the implementation correctly tracks both aircraft, and performs better than `Open CV's` mean point approach depicted in blue

on real-time data (see Fig. 5 for results). As shown in the matrix, the toy example far exceeds the performance of the mean point algorithm, due to the presence of the building occlusion and the Gaussian blur (*please refer to the Appendix for an expanded view of the Figure 5* and Figure 6).

4.2. Performance on the Real World, Dynamic Examples

The implementation worked very well on the real-world dynamic examples compared to `Open CV's` mean point algorithm for object detection (see Fig. 6). Once again, I achieved a 100% accuracy due to the transformation on ground truth data. `OpenCV's` object tracking worked fairly well, but this was due to the fact that the aircraft remained centered in the frame, as the dynamic camera was following the position of the aircraft. When comparing the two approaches, it is clear that my implementation does much better, and it's more accurate with respect to the positioning of the aircraft in both low visibility conditions. Specifically, it is much more centered over the course of the frames in the video, and tracks much better overall. (see

Fig. 6). These results show that radar data can be used to visually track aircraft without relying on complex object tracking techniques.

5. Discussion and Limitations

Though at a high level, this approach was relatively simple, it took an immense amount of manual effort and validation, with constant experimental manipulation to ensure the system worked as intended. The specifics of the implementation required significant attention to detail, and if the homographies computed contained the slightest error, there would be significant affects to the results.

However, the results yield a significant milestone of a novel approach to thinking about aircraft tracking. Currently, there is a wide breadth of startups devoted to the problem of object tracking with respect to airports and airport operations. One example is Traverse 3D, where my colleague Huy Nguyen et al. at Stanford are working on detecting aircraft debris and vehicles on runways using computer vision techniques.

I hope to contribute an approach that is simple, intuitive, and one that leverages accurate radar data in order to increase redudancy and provide a safer operational environment for the industry.

This research certainly has limitations. For one, I only used 3 total examples of pre-recorded footage, which is significantly different than real-time visual input from a camera mounted on glass or a human head. Additionally, my implementation solely focuses on a single aircraft at a time. In the future, I hope that researchers will be able to expand the software to multiple aircraft moving at once, displaying information for each one simultaneously to the operator.

Additionally, there was a slight delay between the movement of the aircraft in the real world and the digital depiction in FlightRadar24. This is due to the latency time it takes for the radar data to update to the cloud and web. In real-world applications, this would not be an issue since the system could theoretically depend on actual radar data over a third part source.

6. Conclusion

Through my research, I have learned an incredible amount of knowledge relating to the problem of real-time object tracking. There are many different approaches to the problem, and some are much more complex than others. Many propose that the use of 3D reconstruction and RCNNs to object track are the optimal way to do so. As proposed earlier, this system could most certainly be improved if integrated with my implementation in cases where an aircraft cannot be seen. One can envision a system where more complex object-tracking approaches are used in most nominal conditions but then rely on the radar data when reaching CAT II/III conditions.

Further, as companies such as Apple are developing spatial computing headsets, future researchers can leverage this new ecosystem to increase the quality of the UX/UI design of the software, leveraging FlightRadar24's API to display additional aircraft information.

7. Supplementary Material

1. Github Repository
2. Link to the final output video presentation

References

[1] HD Melbourne Aviation. 20 minutes of amazing plane spotting at san francisco sfo international airport [sfo/ksfo], 2023. 3

[2] Sara Bagassi, Mohamed Ellejmi, Antonio Nuzzo, Alan Ross Groskreutz, and Tom Nuydens. Retina project conclusions. Technical report, RETINA Consortium, August 2018. Deliverable ID D6.2, Project name: RETINA, Grant: 699370, Call: H2020-SESAR-2015-1, Topic: Sesar-06-2015. 1

[3] Ronald Reisman et. al. Design of augmented reality tools for air traffic control towers., 2006. 10.2514/6.2006-7713. 1

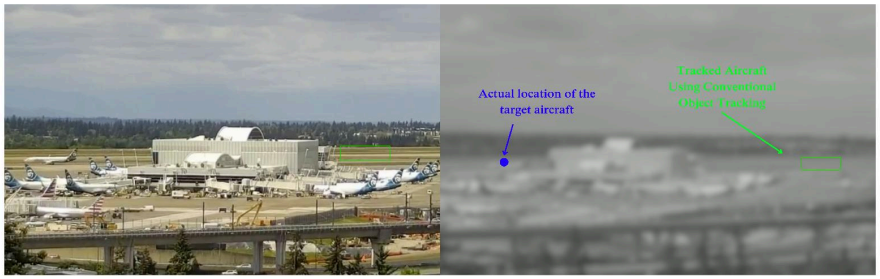
[4] Flightradar24. Flightradar24, 2024. 1

[5] Ross Girshick. Rich feature hierarchies for accurate object detection and semantic segmentation, 2014. 2

[6] Brian Hilburn. Head down time in aerodrome operations: A scope study, 2004. 2

8. Appendix

OpenCV cv2 motion detection



Nominal Conditions | **Gaussian Blur to simulate low visibility**

Projective Transformation with Radar Data

An enlarged view of Figure 5

An enlarged view of Figure 6