Combining a Deep Regression Network Fast Generic Tracker with Object Pose Estimation

Gregory Kravit
SCPD CS231a Final Project
San Diego, CA
gkravit@stanford.edu
Project Github Location: https://github.com/gkravit/CS231A-Final-Project

Abstract

The project aims to attempt to combine the GOTURN Deep Regression Network (DRN) generic tracker with an aerial refueling drogue basket pose estimation algorithm to attempt to implement a high performance computer vision algorithm for use in a potential autonomous aerial refueling (AAR) application. Unfortunately, not much progress was made because I was unable to integrate the DRN tracker into a feasible processing pipeline, which left little time to devote to the pose estimation aspect of the paper. Instead, this paper will discuss the approach intended and the lessons learned so it may be of use to future readers.

1. Introduction

In the project proposal, I stated my interest in evaluating a novel approach to solving the object pose estimation problem, the "problem of determining the transformation of an object in a 2D image which gives the 3D [object's] relative position and bearing to the camera" [1]. I've been kind quite interested for some time in the autonomous aerial refueling (AAR) capability, specifically drogue-style refueling. The task itself is quite well suited for a computer vision based navigation solution due to the operating environment which requires the pilot to navigate visually to insert a fuel probe at the front of the aircraft into a basket being dragged by the tanker aircraft. The pilot is implicitly tracking the basket, estimating the relative pose of the basket, and adjusting the aircraft accordingly. Ideally, an automated application would need to accomplish the same tasks, all within the wheelhouse of modern computer vision techniques as taught in CS231a.

From surveying existing literature on AAR vision-based navigation and from personal industry experience, approaches to this pose estimation task have tended to focus on the pose estimation task with the tracking component tagged on as afterthought or incorporated into the complex algorithm. The computer vision approaches were often very computationally intensive and had to compute across the entire video frame leaving performance in the single frames per second. While this performance could be tolerated when combined with differential radio signals to provide a navigation solution, it is very much still a far cry from the deep-learning capabilities of self-drive vehicles being actively developed by the automotive and technology companies.

Driven by that understanding, I had proposed that one could attempt to avoid having to solve the entire tacking and pose estimation in a single code solution, and that you could in fact modularize the problem into discrete components, tracking and pose estimation. I just so happened in the spirit of the class push to consider deep learning approaches decided to utilize an open sourced deep regression network (DRN) fast generic tracker to provide the optimal window. With that optimal window, I aimed to limit the computation needed to process the pose estimation solution.

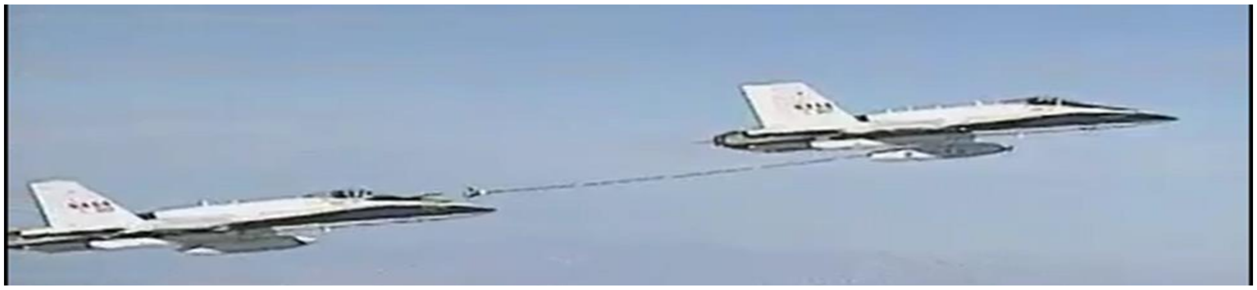


Figure 1: Two F-18s flying in formation for NASA AAR Study [2]

2. Background/Related Work

2.1. Autonomous Air Refueling Description

The in-air refueling operation is a conditional rendezvous problem where in two aircraft fly in close formation where some apparatus forms a bridge to deliver fuel from the lead aircraft to trailing aircraft. In drogue-style refueling, most commonly practiced by the US Navy aircraft and other European air forces, that bridge mechanism is a basket type object that is being towed behind the lead tanker aircraft and the trailing receiving aircraft aims to insert a fuel probe extending from the aircraft to insert into the center of the basket to create a continuous connection. A screenshot from a NASA AAR test is shown is shown in FIGURE 1.

From the trailing aircraft's perspective as shown in annotated FIGURE 2, there are number of features in view that one could track and analyze: the drogue basket, the hose length, the lead aircraft fuselage, the lead aircraft wings, and the lead aircraft engines. In reality, the primary concern is to track the drogue basket in order to insert the fuel probe. The other features are there to provide orientation.

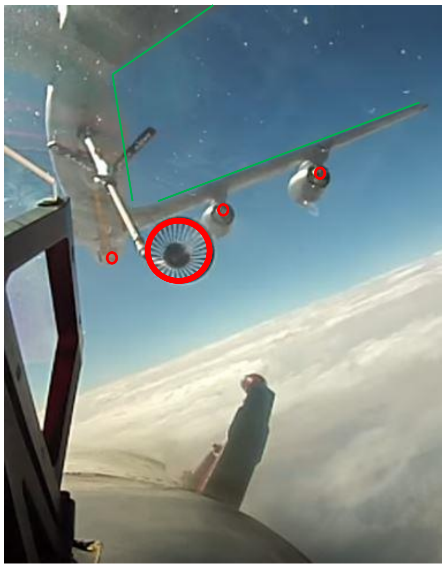


Figure 2: Annotated cockpit view of F-18 leaving refueling operations [3]

2.2. Background Literature

Approaches in the literature to the drogue pose estimation and AAR problem have mainly followed two different approaches: 1) aircraft feature tracking to generate a relative navigation solution (*Spencer* [4]); and 2) or modified circle tracking using analytical methods & modeling to feed an Extended Kalman Filter (*Xu et al*. [5], *Wilson et al[6]*).

2.2.1 Aircraft Feature Tracking Approach

In the aircraft feature tracking approach, the algorithm extracts known aircraft features such as the engines, aircraft structure, aircraft wings and drogue fuel assembly based on known calculated geometric positions. To help aid tracking of the features, previous approaches have proposed attaching LEDs and other distinguishing light sources to make the tracking more straight forward and easier to process (*Spencer et al [4]*, *Wilson et al* [6], *and Valasek et al* [7]). A navigation solution is then computed based on those correlated features to known geometric position and a differential navigation radio (GPS positions). The navigation solution is often aircraft tanker and receiver specific so it is not easily extended without additional tuning and testing between aircraft platforms.

2.2.2 Analytical Position Estimators

In the modified circle tracking and similar approaches, the actual position of the drogue is computed through computer vision algorithms and tracked through known analytical models as part of an individual model updates. These model updates can then be fed into Extended Kalman Filters or other similar position estimators. This often increases the complexity of the control system by requiring an additional control loop. It also requires relying an accurately modeling the drogue and aircraft correctly.

2.3. Pose Estimation

For the pose estimation algorithm, I considered two classes of approaches: Feature Detection and Feature Matching.

2.3.1 Feature Detection

Feature detection compares extracted geometric features and correlates them with a known analytical model. Features or keypoints may be extracted through any of the various methods such as SIFT, HoG or Hough space/probabilistic voting. Those features are then correlated an analytical model. In the case of the AAR problem, features of the drogue can be recognized and then correlated to the physical model of the hose-drogue combo. The correlated model and features would then be used to derive a pose estimate.

One method found in the literature is ellipse extraction as shown in *Wilson et al* [6] Known locations on the drogue basket were tracked. The resulting movements were then used to fit an ellipse to feed the motion estimate of the physical model.

2.3.2 Feature Matching

Feature matching attempts to match extracted geometric features to known templates or training data. With representative templates of the drogue basket such as a CAD model or real annotated videos/images, characterizations of the 3D pose can be found for large range of poses.

That calculated ground truth data can be used in many different ways to then calculate the pose estimate of the object. Direct methods include fitting the data via template search or feature fitting. Machine learning approaches could use the ground truth data to train a SVM or a neural network.

3. Approach

The intended technical solution was to implement the processing pipeline as shown in **ERROR! REFERENCE SOURCE NOT FOUND.** below. The goal was to try a novel approach to the problem by instead of trying to perform both pose estimation and tracking within same algorithm, I would separate the tracking and pose estimation into separate processes. The tracker would first provide an estimated window (or bounding box) for where the drogue is thought to be. The pose estimator would then be able to calculate the relative position and attitude of the drogue found within the bounding box. By having providing an estimated window, more computationally intensive pose estimation approaches can be practically used now that the entire frame does not need to be processed.

3.1. DRN Tracker

The neural network tracker was to be leveraged from the Stanford open sourced GOTURN project by Held, et al [8] for the neural network tracker. At the beginning of the project it was thought that the extensive documentation and instructions on training a pre-trained model would make this code extendible and readily integrated to the drogue basket example. For suitability for this project, its proven performance at 100 FPS on a NVIDIA GPU mitigated performance concerns for the pipeline. The tracker was intended to output a defined bounding rectangle and pixel position of the drogue basket that will be used in the pose estimation step as shown in Figure 3. As explained in section 4 below, I was unable to leverage the GOTURN project as I ran into compiling and setup issues of the GitHub code base and its dependency Caffe onto my development platform, the NVIDIA TX1.

3.2. Pose Estimation Algorithm Selection

The two pose estimation general approaches explained in section 2.3, feature detection and feature matching would both be acceptable for this problem. In fact, feature matching would be better suited for a machine learning based approach by extending the GOTURN neural network similarly to *Su et al* [9].

However, due to constraints on my current knowledge base and that the bulk of CS231a examined coursework focusing on feature detection, pursing a feature detection approach seemed more feasible for this project. It would also hopefully demonstrate the advantages of the pipeline approach by combining the analytical element with the machine learning element.

3.3. Hough Circle and Ellipse Estimation Methods

Based on the reality that this project was aimed to solve the problem from the receiver aircraft reference frame such that the dragged drogue basket end would appear circular, circle or ellipse estimation methods seemed like a good starting place for developing the pose estimation algorithm.

An initial rough approach employing Hough circle methods was implemented on the publicly available YouTube videos. For quick prototyping purposes, the HoughCircle method from the open sourced OpenCV 2.4 [10] library on Python 2.7 was used. Parameters used in the frame shown in FIGURE 4 were tuned manually for good results:

- minimum distance between centers = 10 pixels
- canny-detection parameter = 100
- accumulator threshold = 50
- minimum radius = 10 pixels
- maximum radius = 80 pixels

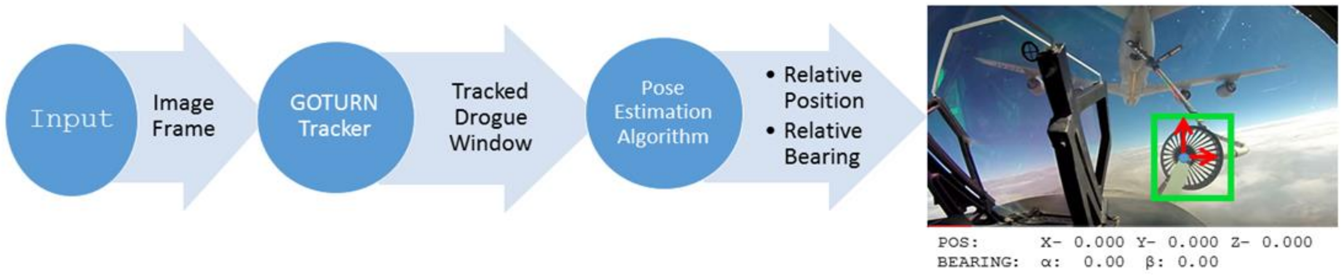
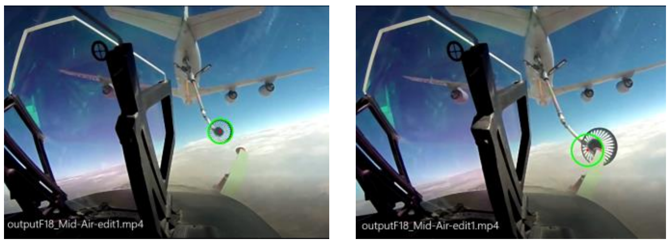


Figure 3: Proposed processing pipeline



(a) (b)

Figure 4: Hough Circle processed frames from video. (a) Shows an illustrated circle correctly tracking the drogue. (b) Shows the method with current constrained parameters is not very accurate at close range and with occlusions.

Hough circle method demonstrated assuming a circular or elliptical shape from the receiver aircraft point of view was sound and could be built on.

Unfortunately, as explained in section 4, issues with setup of the GOTURN project prevented me from working on the any further refinement of the pose estimation code. Dealing with the tracker issues occupied my time as a single person project.

3.4. Proposed Future Path

Given the success of the initial rough Hough circle method, feature extraction using Hough space and probabilistic voting (or another more advanced circle/ellipse finding algorithm) looks to be a valid path forward to develop the pose estimation algorithm. If the proposed tracker were to deliver a confined search window, then algorithm parameters could be relaxed to avoid further false negatives from the results.

Interesting enough, the success of the Hough circle method opens the door for pursuing either of the pose estimation methods explained above. Features extracted from an ellipse method could then be applied to the analytical model using a process similar to *Wilson et al* [6]. [It is also of note, the pose estimation can be ignored entirely as in *Chen et al* [11] where features extracted were only used to calculate the capture radius for image based visual servoing (IBVS). However, that was not the point of this project and was ignored as an approach.]

On the other hand, pose feature matching could be implemented after the probabilistic voting step in Hough space. The flattened histogram of the drogue in many different orientations could be placed in a training set. That training set could be used to train a network (or extend the GOTURN trained network as mentioned in **3.2 above)**, used as a state vector machine or through other methods such as defined *Reinbacher et al* [12] who implemented an efficient search for a silhouette matching approach.

4. Experiment Difficulties

Project output was limited to the simple Hough Circle demonstration as I was unable to integrate the GOTURN project into a feasible video processing pipeline. Due to my struggles, I was left with little time to devote towards the pose estimation side of the project. The goals of the project proved too ambitious in the time frame.

4.1. Hough Circle Videos

Video output and the rough Hough circle OpenCV python code are included in the GitHub location here: https://github.com/gkravit/CS231A-Final-Project. No performance information was collected as it was not representative of the envisioned pipelined process.

4.2. Development Environment Setup Issues

I chose to implement the project on the NVIDIA Jetson TX1 Development board due to the project subject matter being geared to an embedded context on an aircraft, the lack of a personal Linux workstation with a useful NVidia GPGPU (GOTURN was written for Ubuntu and used with a powerful GPU), and the ample processing capability to handle deep learning and computer vision processing tasks with its quad core ARM processor roughly 256 CUDA cores combination.

4.2.1 Caffe Installation Issues

One of the dependencies of the GOTURN project is Caffe. Caffe is an open source deep learning project sponsored by the BVLC lab at UC Berkeley and has accelerated extensions for NVidia proprietary neural network accelerations called CuDNN. I initially did not see this as a risk as I found plenty of documentation online of successful installs [5] [6] of Caffe on the TX1. However, I ran into a number of issues as the existing install instructions and available install scripts did not match the exact locations that the GOTURN cmake files were expecting. As I had little prior experience in source compilation on Linux/ARM architectures, this was quite the learning curve.

After a number of times reflashing the hardware and switching between Caffe forks offered both by the main BVLC GitHub and NVidia’s NVCaffe with FP16 support, I finally was able to work through all the compile and library linkage issues as of 06/07/2017 leaving very little time to complete the project as envisioned. I had reported on the midterm report that this problem was solved. However, after the midterm report when I started to try to integrate the tracker into a pipeline I discovered I had not compiled it properly leaving me further delayed in approaching the other parts of the project.

5. Conclusion

This project attempted to create a pose estimation processing pipeline by combining a neural network based tracker with a pose estimation algorithm in order to be utilized for the autonomous aerial refueling problem. Unfortunately, due to underestimating the amount of work required to adapt the neural network and my present capabilities in compiling source code on embedded platforms, I was severely delayed in executing the project aims and ultimately unsuccessful.

On a positive note, I was able to demonstrate conceptual understanding of the problem with the Hough circle extraction video demos. The relative success of the Hough circle methods with the circular dragged refueling drogue basket certainly made clear the possibilities going forward to achieve the pose estimation solution.

Despite my struggles on the project, I still believe the processing pipeline approach is a valid approach and should be further studied and implemented. I believe this paper can form the basis of a future student project or future engineering project.

References

[1] Wikipedia. "3D Pose Estimation" https://en.wikipedia.org/wiki/3D_pose_estimation

[2] *NASA Video - F/A-18 Automated Aerial Refueling (AAR) Phase 1.* YouTube. https://www.youtube.com/watch?v=jES-Q6inE0o

[3] *F18 Pilot Refueling Midair*. YouTube. https://www.youtube.com/watch?v=1vhHs7k1zwg

[4] Spencer, James H. "Optical Tracking For Relative Positioning in Automated Refueling." http://www.dtic.mil/dtic/tr/fulltext/u2/a469498.pdf

[5] Xu, Yan et al. "Pose Estimation for UAV aerial refueling with serious turbulences based on extended Kalman filter" *Elsevier GmbH, 2013.*

[6] Wilson, Daniel et al. "Drogue Motion Estimation Using Air-to-Air Observations." *Proceedings of the Australasian Conference on Robotics and Astronautics* , 2-4 Dec 2014

[7] Valasek, John et al. "Vision-Based Sensor and Navigation System for Autonomous Air Refueling" *Journal of Guidance, Control and Dynamics*. Vol. 28, No. 5, Sept-Oct 2005

[8] Held D, Thrun S & Savarese S. "Learning to Track at 100FPS with Deep Regression Networks" *European Conference on Computer Vision (EECV)*, 2016. http://davheld.github.io/GOTURN/GOTURN.pdf

[9] Su, Hao et al. "Render for CNN: Viewpoint Estimation in Images Using CNNs Trained with Rendered 3D Model Views." *arXiv: 1505.05641v1*, 21 May 2015

[10] OpenCV 2.4 Documentation. "Feature Detection: HoughCircles". Online. http://docs.opencv.org/2.4/modules/imgproc/doc/feature_detection.html?#houghcircles

[11] Chen, Chao I et al. "Autonomous Aerial Refueling Ground Test Demonstration –A Sensor-in-the-Loop, Non-Tracking Method" *Sensors 2015*, no. 15, 10948-10972

[12] Reinbacher, Cristian et al. "Pose Estimation of Known Objects by Efficient Silhouette Matching." *Pattern Recognition (ICPR), 2010 20th International Conference on*. 23-26 Aug 2010.