Deniz Kartal (deniz001) - Public Page

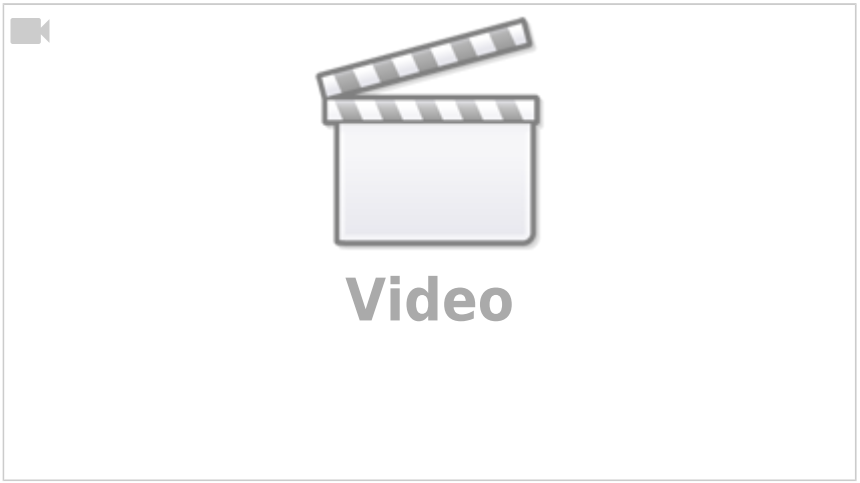
Drone Tracker

Background

Components

Component	Brand, Type	Datasheet	Link
Pan and Tilt Unit	FLIR PTU-5	PTU-5-USER-MANUAL FLIR PTU-5 DATASHEET	PTU-5 @ FLIR
Jetson Xavier NX	NVIDIA	Getting started with Jetson Xavier NX developer Kit	Embedded Computing with Jetson
Camera			

Pan and Tilt Unit



Source: FLIR Systems Inc. YouTube Channel.

Thread: 1/4-20 UNC THRU (from mechanical drawings in the user manual)

0. Introduction

On this page, I would like to introduce an object tracking system. Main idea is to follow a drone indoor but the system can be trained to track any object.

The system consists of a pan and tilt unit, a camera, and software.

Software is implemented to:

- detect the drone object or let the user select a bounding box around the drone in the first frame.
- find/track the object in the future frames.
- control the PTU(Pan and Tilt Unit) using a PID controller to put the drone object in center of the

frames.

There are two options in order to have a working system that tracks a drone. The first option is to basically select an initial bounding box and, the second one is to train a neural network that does the manually bounding box selection step automatically using the trained CNN. Both of those mentioned solutions are implemented in software.

The idea is very simple, we first “detect” or “select” a bounding box around the drone object, then use a tracking algorithm to track the drone object in each frame, and then control the PTU using the PID controller to put the drone object into the center of the frames.

All the source code for this project and usage demo can be found here:
https://gitlab.com/poseidon42/object-tracker

1. Data Collection

Data is very important if we use the second option that we train a neural network. There are a lot of resources to load datasets but I could not find an efficient one for drone images therefore I have written a python script that downloads a good number of user specified object images from google to create my own datasets.

2. Image Augmentation

Data augmentation aims to increase the accuracy of the training by creating different versions of the each original image that we gathered using the python script to collect our data. This way we can use the original images in the validation step and use the augmented images in the training step.

3. Object Detection

In object detection we aim to find an instance of an object of a certain class in an image or a frame of a video stream and output the bounding box coordinates of that instance. In our case, one of our drone from our laboratory is an instance of the drone object.

Object detection can be achieved using machine learning and deep learning based algorithms. In machine learning approaches we need to feature engineer in order to define features. While in deep learning, we do not need to manually define the features but rather the CNN(Convolutional Neural Network) finds its way to define the features. In conclusion, the deep learning approaches do not require feature engineering and give better results, therefore I decided to use deep learning in the object detection step.

Example of Machine Learning approaches:

- Viola-Jones object detection framework based on Haar features
- Scale-invariant feature transform (SIFT)
- Histogram of oriented gradients (HOG) features

Example of Deep Learning approaches:

- R-CNN(Region-based Convolutional Neural Networks)
- Fast R-CNN
- Faster R-CNN
- SSD (Single Shot MultiBox Detector)
- YOLO (You Only Look Once)

In this project, I have decided to try out both YOLO and SSD, but in the future faster R-CNN could be implemented as well.

Faster R-CNN:

YOLO:

YOLO is a real-time object detection algorithm. YOLO divides each image into a grid of S x S and each grid predicts N bounding boxes and confidence. The confidence reflects the accuracy of the bounding box and whether the bounding box actually contains an object(regardless of class). YOLO also predicts the classification score for each box for every class in training. In total S*S*N boxes are predicted. However, most of these boxes have low confidence scores and if we set a threshold say 30% confidence, we can remove most of them. Image is run through the CNN only once at run time. Framework for YOLO is also YOLO.

SSD:

SSD runs a convolutional network on input image only once and calculates a feature map. SSD uses anchor boxes at various aspect ratio and learn the off-set rather than learning the box. In order to handle the scale, SSD predicts bounding boxes after multiple convolutional layers. Since each convolutional layer operates at a different scale, it is able to detect objects of various scales.

Why not using tracking by detecting in each frame?

- There can be multiple objects entering and exiting the view of the camera over time in frames, in that case there is no possibility to match or connect the objects in the current frame with the previous frames that the camera was recording in the past.
- The object may suddenly go out of the camera's view in the next frame then another same instance of the object(drone) may get in the frame, in this scenario there is no way that the system can figure out which object that the system was actually tracking. To sum up, there is no way for the system to have an idea about object's current and past movements but in this project the purpose is to track a unique object so that we can calculate where the object goes and create a motion map.
- There can be blur or noise in the frames due to the motion of the object or camera, that is why the object may look very different so the detection would fail again.
- The object may have a various viewpoints that trained neural network was not prepared for that
- The object may go far away from the camera so that the distance between the camera and the object would be so much, in this case as there will be a huge change in the scale of the object this would most likely cause a failure in detection.
- Low resolution, the number of pixels

- Tracking algorithms are faster than object detection algorithms, as detection algorithms aims to learn all the detailed information of the object ,while trackers do not intend that.(High computational usage in object detection.)
- There can be obstacles where the object ,that the system suppose to track, may hide behind another object. In that case, trackers can estimate that the object may be behind this obstacle ,however any object detector cannot do that unless there is a system giving the detector help for this speculation.

4. Object Tracking

By using an object tracking algorithm we can uniquely identify an object instance, and locate a moving object by estimating the location of the target object in the future frames and check if the object in current frame is the same as the one which was in the very previous frame.

Tracking process goes by first initially defining a bounding box of the target object.

A good tracker must model the motion, and appearance of an object and detect the motion space to localization the object in the future frames using the knowledge from the past frames.

Motion modelling

Any object do not randomly move in the space but rather they have moving characteristics and patterns which can be modeled.

Therefore, a successful object tracker must understand and model a movement estimation model which remembers how the object moved in the past frames in order to predict the next possible location space of that the object can be present. This will also make the algorithm faster by reducing the size of the region of interest that the tracker needs to scan for that object.

Appearance Modelling

An instance of an object has also an appearance characteristics. A good tracker must understand the appearance of the object that the tracker tracks by using the previous frames to train the appearance model and also they must learn to differentiate the object from the background which is in the image.

To sum up, if the tracker has efficient models about the object's look and behavior, it can then use this knowledge to find the exact location of the target object in that current frame.

Type of object trackers:

Offline learning trackers are used when we have a recorded media, in that case we also use the future frames to make tracking predictions.

Online learning trackers train itself to learn about the object which is inputted to the tracker for learning by drawing a bounding box around that object. Those trackers use an array of frames, starting from the initial frame until the frame that is one before the current frame.

A decision has to be made:

1. Use an online tracker that could train itself.
2. Use an offline tracker that has been already trained.
3. Train an offline tracker to identify only the drones.
4. Train an offline tracker to identify drones and many other objects.

Offline trackers do not need to learn anything during the tracking process, that sounds faster but training is not an easy task because we can never train a CNN for every possibility. However, online learning trackers may just learn about the object that we are interested in at that moment, for example the object may be red and the background may have no red color, in this case it will be so easy to track the object, in the opposite case it may be very challenging. This is not a physics problem that we can explain and formulate using mathematics but rather an engineering problem that requires experimenting and many trackers have its advantage in different cases, therefore, I have decided to implement several tracking algorithms which the user can decide what to use in different scenarios.

Most of the traditional trackers that are available in OpenCV are not based on deep learning CNNs and my favorite algorithm is KCF.

CNN(Convolutional Neural Network) based offline trackers: GOTURN CNN(Convolutional Neural Network) based online trackers: MDNet(Multi domain network) best DL based

Tracking algorithms available in this system:

- **Boosting Tracker:**
- **MIL Tracker:**
- **KCF Tracker:**
- **MEDIANFLOW Tracker:**
- **GOTURN Tracker:**
- **MOSSE Tracker:**
- **CSRT Tracker:**
- **MDNet Tracker:**
- **ROLO Tracker:**

5. PID Controller

PID controller is a feedback control loop, its output is fed back to the system as an input to reach the target.

The output of the tracking algorithm is a bounding box that represents the location of the object that we track, that is the drone object. Using the output of the tracker the error, that is the distance between the center of the current frame and the center of the drone object in the current frame, is calculated and this error is the input to the PID controller which tells the PTU(Pan and Tilt Unit) in which direction to move in order to put the object in the center of the current frame.