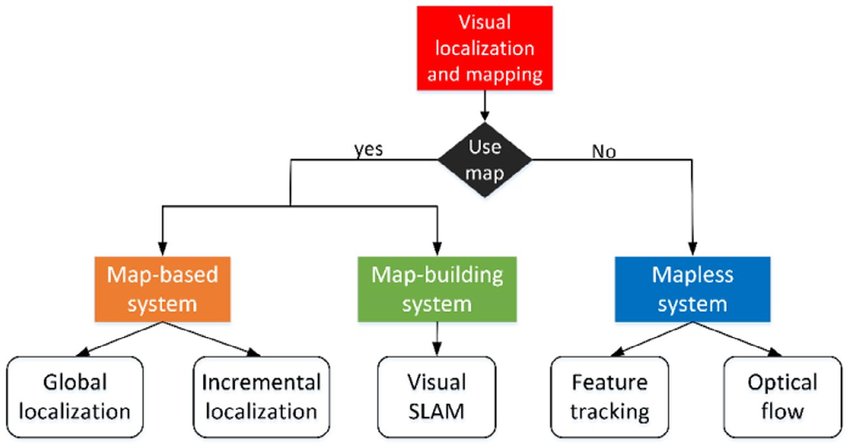
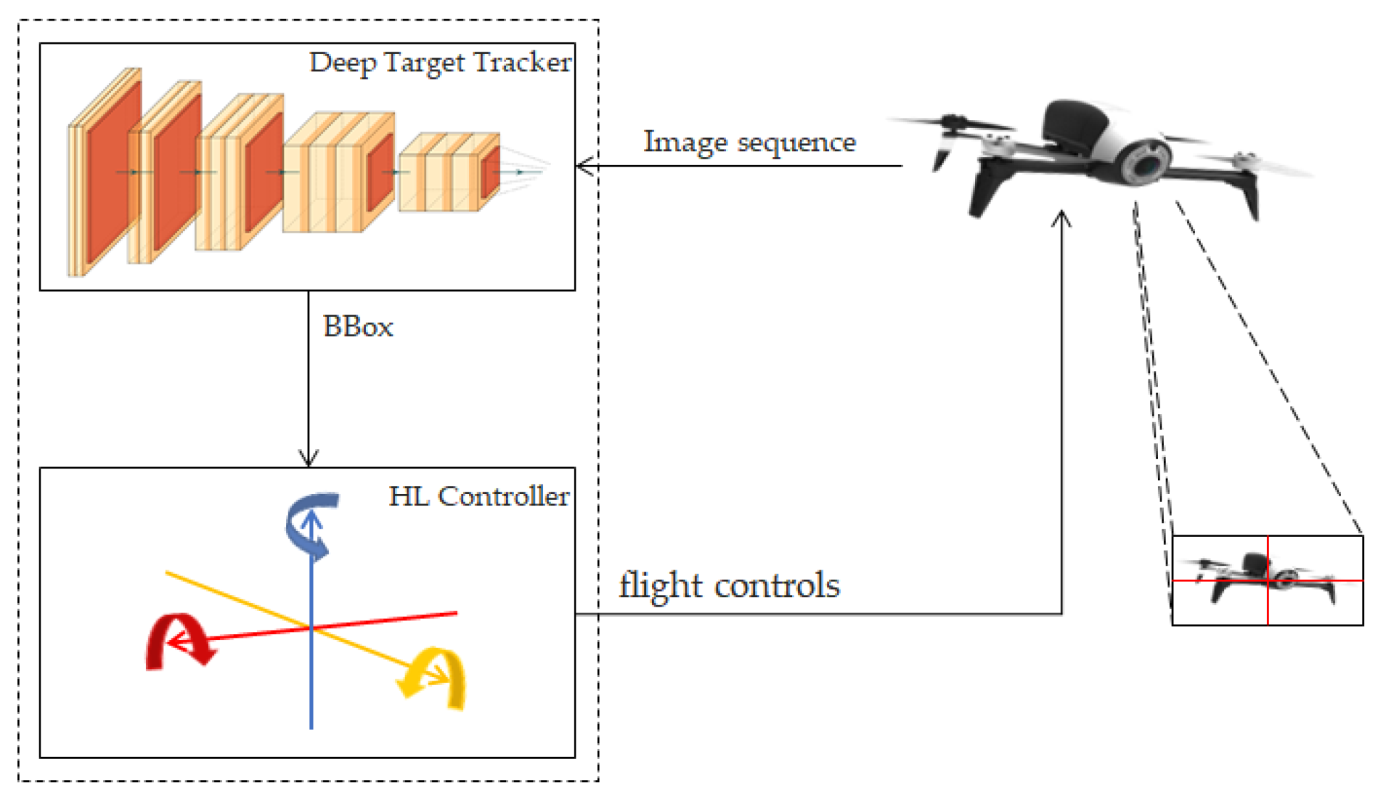
**PROPOSAL AND IMPLEMENTATION:**



In GPS Denied Navigation, the localization is done using

* Feature tracking
* Optical flow



A UAV first captures the sequence of images of the target and estimates the path. In order to achieve this, two steps are involved

* Object detection and Classification
* Distance calculation

**Object Detection and classification:**

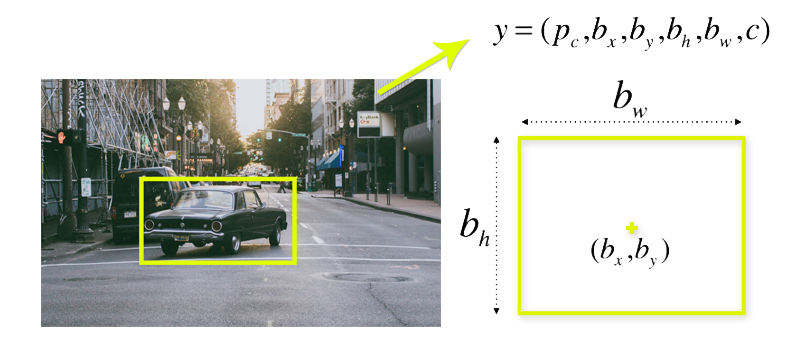
**YOLO:**

1. Algorithms based on classification – they work in two stages.

* In the first step, we’re selecting from the image interesting regions.
* Then we’re classifying those regions using convolutional neural networks.
* This solution could be very slow because we have to run prediction for every selected region.

1. Algorithms based on regression – instead of selecting interesting parts of an image, we’re predicting classes and bounding boxes for the whole image **in one run of the algorithm**.

Most known example of this type of algorithms is **YOLO (You only look once)** commonly used for real-time object detection.



The objects in the above image is detected using YOLO algorithm. The data set is fed to the algorithm and using the coordinates given, the objects are classified.

STEPS TO INSTALL YOLO:

**Requirements**

* Windows or Linux
* **CMake >= 3.8** for modern CUDA support: <https://cmake.org/download/>
* **CUDA 10.0**: <https://developer.nvidia.com/cuda-toolkit-archive> (on Linux do [Post-installation Actions](https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#post-installation-actions))
* **OpenCV >= 2.4**: use your preferred package manager (brew, apt), build from source using [vcpkg](https://github.com/Microsoft/vcpkg) or download from [OpenCV official site](https://opencv.org/releases.html) (on Windows set system variable OpenCV\_DIR = C:\opencv\build - where are the include and x64 folders [image](https://user-images.githubusercontent.com/4096485/53249516-5130f480-36c9-11e9-8238-a6e82e48c6f2.png))
* **cuDNN >= 7.0 for CUDA 10.0** <https://developer.nvidia.com/rdp/cudnn-archive> (on **Linux** copy cudnn.h,libcudnn.so... as desribed here <https://docs.nvidia.com/deeplearning/sdk/cudnn-install/index.html#installlinux-tar> , on **Windows** copy cudnn.h,cudnn64\_7.dll, cudnn64\_7.lib as desribed here <https://docs.nvidia.com/deeplearning/sdk/cudnn-install/index.html#installwindows> )
* **GPU with CC >= 3.0**: <https://en.wikipedia.org/wiki/CUDA#GPUs_supported>
* on Linux **GCC or Clang**, on Windows **MSVC 2015/2017/2019** <https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Community>