**UAV Interception**

*ECE-6254, Statistical Machine Learning*

*Final Project Proposal*

*Georgia Institute of Technology*

*March  23, 2017*

Benjamin Sullins, Eric Davis, Hyeon Ki Jeong, Rish Ananthan

**Project Summary**

In recent years the number and availability of small, maneuverable autonomous aerial vehicles has grown significantly. As a result, concern around privacy, safety, and monitoring of these devices has become a hot topic. Their small size and generally low flight altitude means that identification methods must be capable of tracking and classifying targets amongst cluttered scenarios where the object of interest may exist on a similar spatial scale to many other objects in the image. The similarity of tracking other small aerial objects such as missiles in a war zone, or birds around an airport, gives any solution to this problem even broader applicability.

Our proposal is to apply the methods of statistical machine learning to the tracking and identification of small aerial vehicles in challenging synthetic video scenes. Our plan is to start with synthetic data, which we intend to generate as a deliverable of the project, allowing us to increase the image complexity as we develop and train the algorithm. In this way we will have direct control over the dataset, and can develop and debug our tracking/classification algorithms on incrementally more cluttered environments.

The plan for the project can be broken down into four general sections, some with further subsections. These sections are background scene generation, target model generation/animation, maximum target likelihood algorithm, and target position estimation algorithms. Each of these can be further broken into subgroups, beginning with initial research through a sequence of steps where the complexity of the problem is increased. A detailed description of the project plan is given in the following section, with an itemized list of tasks located at the end of the paper.

Figure 1: Ordinance UAV Figure 2: Surveillance UAV

**Detailed Description**

**Background generation** will consist of an initial simple design and eventually increment to a more complex implementation.The simple design will begin by establishing synthetic images with induced Gaussian noise as seen in Figure 3. Generic parameters for background noise characteristics shall be adjustable by the user to simulate Low, Moderate, and High complexity benchmarks for overall algorithm assessment. The adjustable parameters shall include but not be limited to noise mean, noise variance, salt-pepper noise inclusion, and variable spatial noise complexities.

After the initial algorithm design has sufficiently succeeded in tracking targets against High complexity benchmarked scenarios, enhanced complex background generations will follow. The complex design will begin by acquiring images from an online database and perform Gaussian noise injection on the computed power spectral densities as seen in Figure 4. The allowable noise injections shall be subject to the requirements needed for generating Low, Moderate, and High complexity benchmarks for overall algorithm assessment.

The critical importance of the Background Generation task is to establish the synthetic backgrounds used for the underlying target environment upon which the tracking algorithms will be running. Severity of the noise injections will strain the tracking performance and establish quality metrics upon which to compare. Potential challenges include proper selection of noise injection severities for benchmark results. These benchmark selections can be overcome by establishing baseline estimates from academic standards on target tracking in cluttered environments.



Figure 3: Simple Gaussian Noise Figure 4: Complex Noise Induced on PSD

**Target generation** will consist of a number of sub programs, loosely divided into target model creation, target motion generation, and background image integration and post-processing. The goal is to design a set of scripts which will automatically create target shapes and trajectories based on parameters inputs by the user. The target images and motion information can then be combined with different background images to build the complete training and test sets.

Initial target modeling is planned around the use of a two dimensional Gaussian blob. The Gaussian shape can then be translated to different positions in the image plane according to a time series of coordinate information, resulting in a video like progression. This progression can be saved as a set of images with an alpha channel to be laid over the background as a separate step, or generated directly in the background scene without the additional step. The former being preferred, as this will allow the development of code which can be reused with more complex models at a later stage. As time permits, a second phase will source more realistic models which can replace the simple Gaussian blob. These models will be from openly available images online, 2d images generated from 3d models, and/or 3d images rendered directly in the scene.

Initial target positions will be generated using standard kinetic models for ballistic motion, parameterized according to starting velocity and bearing. Additionally, to model more complex motion such as that of a drone or airplane, a stochastic model will be generated to randomly alter the trajectory of the target in a realistic manner at random points in time. In this case, the physical laws dictating possible motions of a mass bearing target will be accounted for, in order to maintain real world applicability of the results. Note that the coordinate time series created to position the target image will also be saved off and used as the true position for training and testing of the tracking algorithms.

As a final step, some level of post-processing will be applied to the entire video progressions. This will most likely take the form of a slight noise to be injected uniformly over the entire scene after target overlay. Additional possibilities, time permitting, could be to add alternate forms of clutter to the scene, other moving targets, or attempting to simulate shadowing as the target moves through the scene. Additionally, video of actual aircraft or missiles could be sourced off the internet and used as a final test of the algorithm, assuming they can be appropriately processed for input to the resulting program within the time allowed.

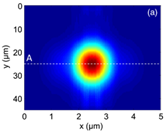
 

Figure 5: Gaussian Target (Simple) Figure 6: Commercial UAV Model (Complex)

(Propose Using This - Ben)

**Maximum Likelihood Estimation** **(MLE)** will consist of both a simple and complex design aimed at estimating/segmenting the target from the background for improved target tracking. The simple MLE design will require initial data elements from assumed background and target data points. These data points will be acquired through windowed regions on and around the assumed target location as seen in Figure 7. The MLE will use these data points to aid in selecting an optimum threshold as seen in the example Figure 8. The complex MLE design will incorporate more advanced selection criteria about the windowed region for robust MLE thresholds. Variable background window sizes will be modular to support establishing better background estimations.

The critical importance of the MLE is its capacity to optimally select a threshold for segmenting the target from the assumed background. This threshold aids in reducing error in the target location estimations following this algorithm step. Potential challenges include proper window sizes, background windows with target signal contamination, and tainted window regions which increase threshold selection errors. These problems will be address through reading academic papers which aid in overcoming the stated issues.

**Maximum Likelihood Estimation** **(MLE)** will consist of target/background estimation and segmentation for improved target tracking. MLE is a method which finds parameter values maximizing the likelihood which is used in estimating population characteristics. This method estimates particular parametric values that make the observed results the most probable based on the few data points from some sample of the overall population. The information would be normally distributed with some unknown mean and variance. The MLE algorithm requires some initial data points from assumed background and target data points. This will be achieved through windowed regions on and around the assumed target location as seen in Figure 7.

To apply the concept of MLE, there are few ground rules that must be set. First, there should be sufficient data that are ground truth and estimate both the mean vector and variance-covariance matrix of the population. Second, the distribution of the population should be a normal distribution. The probability density function of the original image and the additive noise will be defined in order to appropriately derive the maximum likelihood and map the estimates of the background. The same procedure will be done for the target estimation.

To distinguish between the background and the target, we would need to determine an optimum threshold based on minimizing the false detection rate within the scene. Multiple trial and error method will be implemented to find the best optimum threshold.

Figure X. illustrates how these methods will be applied. The orange window bounds the background region while the red window surrounds the expected target location. Understanding the underlying distribution of these two windows will allow us to disseminate between what is either. Figure X2 provides a sample likelihood Gaussian distribution of the signal and the noise (background). Our goal is to find these differences to accurately approximate the target.

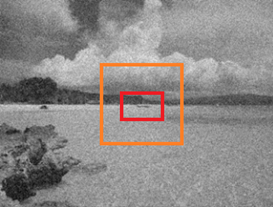
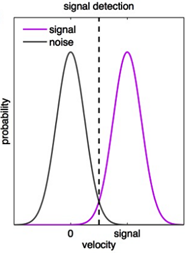
 

Figure 7: A sample image of background and Figure X2. Illustration of likelihood

target where the orange frame indicates the   distribution of noise (background)

background and the red indicates the target.   and signal (target)

Target motion tracking algorithm, Kalman filter

•    Step 1 (Simple KF):

•    Simple Design for Expecting Locations in Next Frame

•    <https://www.mathworks.com/help/vision/examples/using-kalman-filter-for-object-tracking.html>

•    <https://www.youtube.com/watch?v=GBYW1j9lC1I>

•    Code Provided in Link:<http://studentdavestutorials.weebly.com/>

•    <https://classes.soe.ucsc.edu/cmpe264/Fall06/Lec15.pdf>

•    Step 2 (Extended KF):

•    Develop Extended Method for More Complex Scenes

•    <http://www.goddardconsulting.ca/simulink-extended-kalman-filter-tracking.html>

•    <https://ags.cs.uni-kl.de/fileadmin/inf_ags/opt-ss12/lec10_opt.pdf>

Finally, we track the motion of the UAV by estimating the target and background for the future frames. Tracking any object can be broadly classified into three types [1]:

* **Point tracking:** Moving objects are represented by their feature points during tracking.
* **Kernel tracking:** These techniques involve computing object kernels from one frame to the next.
* **Silhouette tracking:**  The aim of a silhouette-based object tracking is to find the object region in every frame by means of an object model generated by the previous frames.

In this project, we will use a Kalman filter to track the motion of the UAV. Kalman filtering is a point tracking method. It is based on the Recursive Least Squares (RLS) algorithm, which estimates the position of an object based on previous estimates. However, unlike RLS, Kalman filtering is able to account for the dynamics of the object [2].

**Sources:**

[1] <https://pdfs.semanticscholar.org/25a6/c5dff9a7019475daa81cd5a7f1f2dcdb5cf1.pdf>   
  
[2] ECE 6250 Course Notes: Justin Romberg  
  
[3]

**Project Tasks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task** | **Leader** | **Deadline** | **Importance** | **Challenges** |
| **Generate Training and Test Data Sets** | Ben/Eric | 4/02 | High | - |
| ·         Collect background images | Ben | 3/28 | High | Suitable Benchmarks |
| ·         Normalized grayscale backgrounds | Ben | 4/01 | High | N/A |
| ·         Gaussian blob creation script | Eric | 3/28 | High |  |
| ·         Ballistic motion model | Eric | 4/02 | High |  |
| ·         Stochastic motion model | Ben | 4/02 | Medium | N/A |
| ·         Post processing script | Eric | 4/02 | Medium |  |
| **Target Identification Algorithm** | Hyeon/Ben | 04/18 | High | - |
| ·         Collect Research Sources | Hyeon | 4/02 | High |  |
| ·         Disseminate Research Content | Hyeon | 4/02 | Medium |  |
| ·         Design MLH Detection Algorithm | Hyeon/Ben | 04/18 | High | Proper Window Sizes |
| ·         Threshold Calibration | Hyeon/Ben | 04/18 | High | Window Contamination |
| **Post Detection Tracking** | Rish/Eric | 04/21 | High |  |
| ·         Collect Research Sources | Rish | 4/02 | High |  |
| ·         Disseminate Research Content | Rish | 4/02 | Medium |  |
| ·         Simple Kalman Filter | Rish/Eric | 04/18 | Medium |  |
| ·         Extended Kalman Filter | Rish/Eric | 04/21 | Low |  |
| **Final Project Report and Poster** | Team | 05/04 | High |  |
| ·         Poster Creation | Rish/Hyeon | 04/25 | High |  |
| ·         Final Report | Eric/Ben | 05/04 | High | N/A |

**References**

citethisforme.com/

<https://en.wikipedia.org/wiki/Maximum_likelihood_estimation>

<https://pdfs.semanticscholar.org/1dc4/1278d777404050ac226b84942bc1a8864b73.pdf>

<https://www.coursera.org/learn/digital/lecture/CWoPV/maximum-likelihood-and-maximum-a-posteriori-estimation>

<http://www.jars1974.net/pdf/12_Chapter11.pdf>

<https://www.mathworks.com/help/vision/examples/using-kalman-filter-for-object-tracking.html>

<https://www.youtube.com/watch?v=GBYW1j9lC1I>

<http://studentdavestutorials.weebly.com/>

<https://classes.soe.ucsc.edu/cmpe264/Fall06/Lec15.pdf>

<http://www.goddardconsulting.ca/simulink-extended-kalman-filter-tracking.html>

<https://ags.cs.uni-kl.de/fileadmin/inf_ags/opt-ss12/lec10_opt.pdf>

<http://cvcl.mit.edu/database.htm>