Engineering 10573-080
Image Tracking Project

The "Image Tracking" Project once again has very everything to do with programming structure and style, instrumentation of experiments, and display of results. Of course, you're always encouraged to go deeper and farther.

You will be finding the moving circular dots in an image sequence and placing a visual mark on the center of them. The dots represent moving targets in the EO video stream and finding them and their center locations in the image represents finding the location of the targets whose geographic locations in the scene could then be determined through associated camera parameters. The image sequence has been delivered to you as an AVI file and the final answer you produce should be an AVI file with the marks placed on the targets. The imagery has added Gaussian noise as well as Shot noise which is a simple model of the expected error that may result from real world implementation. The imagery must be cleaned and processed in such a way as for the automatic determination of the center of the circular target.

Have fun and DO NOT wait until the last minute to start the project … You will have problems.

What to Hand In

1. Zip-file uploaded to the TCU Online website with
   - your code, M-files
   - a Microsoft Word file with your Write-up.
   - a single AVI-file which contains the new movie with the marks on the circular targets.
2. A paper copy of your Microsoft Word Project Write-up. (This is what will be graded)

Image Tracking Concept

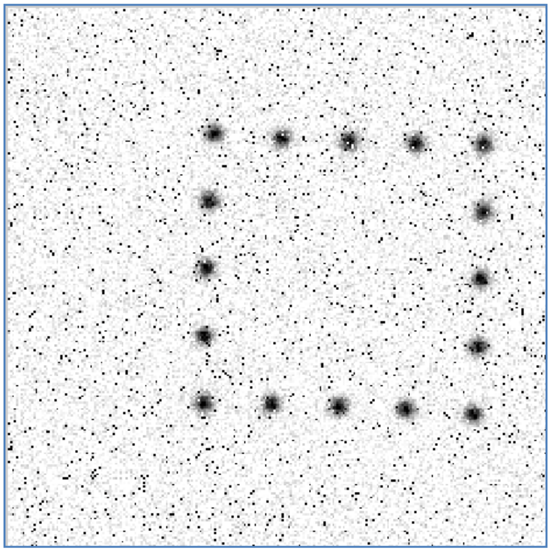
Suppose we have an imaging system that views a scene and there exists physical objects that appear within that scene and we are to design a system that finds, locates and tracks those objects within the temporal sequence of images. The objects can be treated as rigid body so their locations can be represented by a single three-dimensional coordinate. Instead of representing a complex rotating rigid body, we generalize the target as a single point with height and breadth, and since the objects have been generalized to a point, the center of the objects can be used as the location of the object. It is this center location that is to be used and tracked.

To obtain the center location of an object, pixels within the image must be determined to be either target/object pixels or background pixels. This allows for each pixel to be one of two values, ON or OFF; one value to represent a target/object pixel and one value to represent a background pixel. This forms a binary image. The center of the object is calculated as the Centroid of the target pixels whose location is the average of all of the row pixels and the average of all the column pixels. Since each pixel is the same size in the 2-dimensional plane, the size of the object can be estimated by the total number of target pixels. This is also termed the Area of the object.

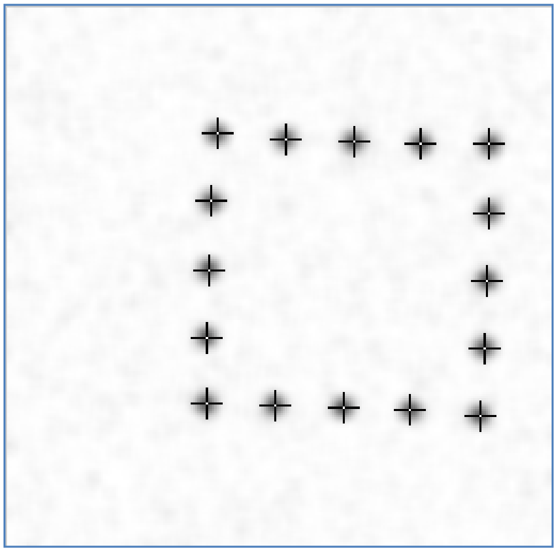
Once the location of the object is determined in sequential images, association of the object locations is performed. This association places the objects in a sequential order. In the basic data association algorithms, the k+1 location is predicted from previous trajectory locations and the next object location that is closest to that predicted location is used as the next object location input. The objects within the scene will also move in a trajectory in three-dimensional space but the image sequence captures their movement in a two-dimensional planar space. The objects move in a semi-predictable trajectory governed by the laws of physics so their movement in the 2D image plane should be semi-predictable as well. To perform any type of error analysis in this project, the student must address the problem of data association so that the error between the calculated position and the truth position is consistent. The given image sequences are temporally sufficiently close enough together to allow for a "closest object" algorithm to be implemented. If data association is performed then the tracks can be estimated and then drawn to show the trajectory pattern in the image.

Image sensors operate in the real world and accurate representation of their output is not a perfect representation to the targets and the background. The image sequences have varying forms of noise embedded in the data. Noise either affects the pixel values or the pixel locations within the image grid. We will assume for this project that the pixel values given represent their true locations with the image and no geometric noise is present. However, the pixel data for the project image sequences have added Gaussian noise and added shot noise.

Sample image frames are shown in the following figures.



Original Image Frame



Cleaned Image with Added Marks

Assignment:

Write program scripts with any supporting functions to implement your own image target tracking algorithms. Clean the noise from the image frames, locate each target center and place a mark on the image denoting that center location. The processed imagery should be written out into a single AVI file. Use the provided functions as an example of how to read and write AVI movie files. Attempt to characterize the performance of the processing algorithms by presenting the characterization parameters in a table or a graph. It is up to you as to determine how best to characterize the performance of your algorithms. If you graph any performance characteristics, remember to appropriately label the axes and title the plot. You should be thoughtful and inventive in the determination of any performance characteristics. Spend some time presenting the data in unique ways. The write-up for this project should be a quality work since sample code has been provided.

> There **may** be a supplemental assignment to this tracking exercise that utilizes a Kalman filter as a tracker and the subsequent center object locations as measurement input. The Kalman filter code is delivered as `classdef point_tracker` and the assignment will be to process an additional set of input images where there are frames where one or more of the targets disappears and then reappears. This simulates a loss of measurement data to the imaging sensor such as a target that cannot be found or acquired from the sensor. This would serve as extra credit.

Implementation:

Matlab has several functions available that are able to perform the functions required to process the imagery sufficiently for the project. They are all available through the Image Processing Toolbox or any of the other associated Toolboxes. Suggested processes are to

- Remove the Shot noise,
- Remove the Gaussian noise,
- Threshold the grayscale image to produce a B&W binary image,
- Remove spurious pixels in the binary image so that only the larger targets remain,
- Calculate the center of the target, *(i.e., Centroid)*
- Limit the size of the target by the number of pixels on the target,
- Draw a mark on each target in the image,
- Draw the historical tracks on the image frame, and
- Characterize the performance of the algorithm using the truth file given.

We will discuss these functions further in class as well as the Kalan tracker.

The movie will have to be opened and the image frame data copied into a local image for subsequent processing. The suggested reading and writing procedures are to use the functions in the Video objects. Matlab Classes such as `VideoReader(·)` and `VideoWriter(·)` should be used. The associated functions return a structure array whose size is `1×N` where N is the number of frames in the movie. Each structure in the array contains an image and a colormap. The colormap is just a 256-bit grayscale linear map.

Once the image is processed to the student's satisfaction, the image should then be copied into an output movie frame of a separate output movie object. When this movie is closed the data is then written to file. One aspect of Matlab that is currently in error: If the written code has an error in it and Matlab stops the processing between the time the movie is created and the time the movie is closed, Matlab continues the hold a handle to the movie and will not release it to the operating system. The next time the student runs the Matlab code after attempting to fix the offending error, Matlab is not able to use the same filename since the file has been locked. You can use the command, `clear mex` and `close all` to force the Matlab program to release the file handle.

The code for the reading an AVI movie file, copying the frame data to an image, copying the image data back to an output movie frame and then saving the output movie has been provided as an example. You need to provide the image processing code needed to find the targets and place a mark on the target in the image. A function to place a mark on a given image location is also provided. There are also functions provided that read and write a binary file containing the track data called `'input_truth_2024.bin'`. The file is (#f x #pts 16 x 2) which is the (number if image frames, number of points, rows & columns in pixels) of the centroid of the points. To compare your tracking data with the truth data and perform error analysis, you will have to perform the data association tasks so that the points in the file line up with the points you have found. They most likely will not be calculated in the same order. Use `associate_tracks(·)` to reorder them to align the two data sets.

Provided Code/Data:

Functions

```
function Iout = draw_mark_on_target(Iin, stats, obj, crs, color)
function success = write_track_data(filename, tracks)
function [success, tracks] = read_track_data(filename)
function assoc_tracks = associate_tracks(true_tracks, in_tracks)

classdef point_tracker (This may be uploaded later)
```

Input Data

```
image_tracking_input_2024.avi
input_truth_2024.bin
Image Tracking Project.pdf (This document)
```

Coding Examples

```
run_movie_image_example
```

Delivered Code/Results:

```
function tracks = image_target_tracking(in_fname, out_fname)
studentsname_output.avi
function [mu, sigma] = calculate_statistics(x)

     Provide a script that sets the proper data parameters,
     calls the required methods and any needed routines.
```

The function `image_target_tracking(·)` should open the input and output files, `image_tracking_input_2024.avi` and `studentsname_output.avi` loop through the input image frames, process each frame, mark the image, and then write the output movie file. You may use any Matlab code available to you that you deem necessary. You do not have to utilize the functions `draw_mark_on_target(·)` and `associate_tracks(·)` but it is provided to you if you so desire. You may also draw the tracks using the `plot(·)` function on a movie if you desire. Use the functions `imshow(·), image(·)` or `imagesc(·)` to create a figure for copying the imagery to your Word document.