ECE 275

Spring 2020
02/06/2020

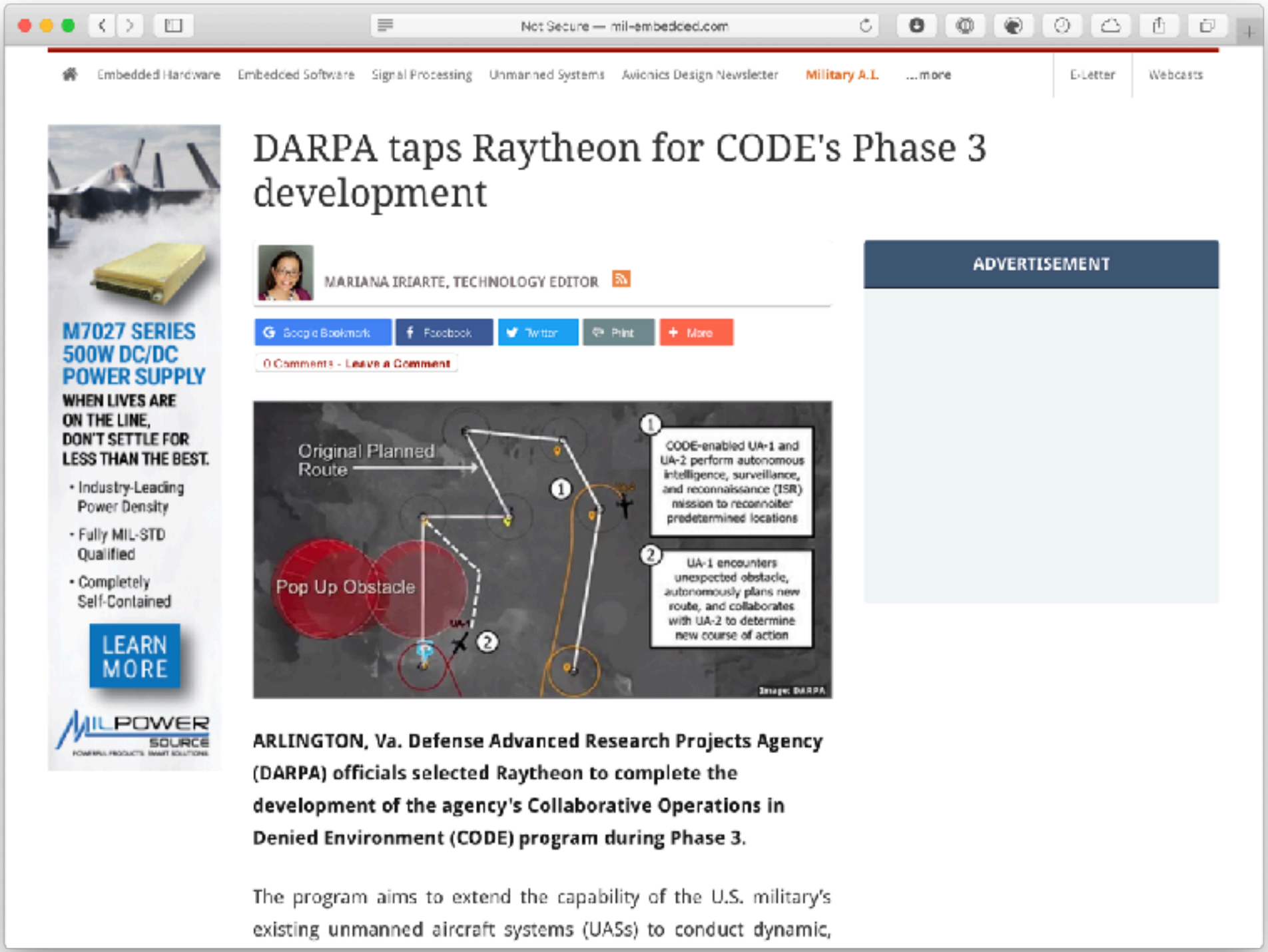
THE UNIVERSITY
OF ARIZONA

Today's Plan

Class Plan 02/06/2020: ECE 275

Activity	Time	Time Left
introduce Project 1	25	50
Introduce Structures	0	50
^ ZyBooks 16.11	10	40
^ ZyBooks 16.12	10	30
^ ZyBooks 16.13	10	20
^ In class challenge (if time allows)	20	0
^^ zyDE 16.12.1: Monetary change program.	^	
^^ 16.12.2: Structs and functions.	^	
^^ zyDE 16.13.1: Modify the TV watch program.	^	
^^ 16.13.2: Structs and vectors.	^	

Software IRL



Project 1: “Lazah Beam”

- Due 02/19/2020
- ZyBooks Section 9.21
- *Don’t wait until the last minute*
- All concepts needed to complete the project will be introduced before the project due date. However, I encourage you to read ahead in ZyBooks and begin as early as possible
    - 02/11 Preparation introduces user-defined functions

- Your program must be capable of utilizing a command line argument to specify the input log file and output file with the filtered data and analysis results.
    - `./project1 inputFile outputFile`
- Your program must ensure the user has correctly provided the required command-line arguments and display a usage statement if the provided arguments are incorrect.
- Create a program that will produce a smoothed curve of timestamped distance data, with annotations to denote significant changes in the angle of detection, or distance to detected object. The resulting timestamped data (with annotations) should be placed in the outputFile.

Input Data

- The data in the inputFile are simplified information from a front-mounted LIDAR sensor on board an autonomous vehicle. All data presented in test files are collected from the CAT Vehicle Testbed at the University of Arizona.

- Data are collected at a frequency of 75 Hz, and each datapoint consists of 181 samples of distance information that scan an arc of pi radians, with time information that indicates the computer time at which the computer recorded the data from the sensor. Each pulse of the laser returns a distance (in meters) of any reflection it encounters; if no reflection is detected, then the maximum range of the sensor is returned---81m.

- A separate software component has already analyzed this hemispherical scan, and found the smallest distance value for each sample, and its index in the vector of length 181. Through the index number, the angle (in radians) of the nearest sample is returned.

- Through the use of this distance, angle, and timing information, the distance between the CAT Vehicle and its nearest obstacle can be inferred; differentiating this data provides velocity and acceleration information---but unfortunately the noise in the data prevents the direct application of a differential operator. Thus, **the goal of this project is to smooth the data, so that it can be more suitable for estimating relative velocity**.

Data structures

The following data structure and enumerated type should be used for storing and filtering data.

```
// This enumerated type provides entries for each of the
// potential labels attached to a sample
typedef enum FilterStatus { UNDEFINED=-1, VALID, FILTERED, ANGLE_RESET, DISTANCE_RESET }
FilterStatus;

// This struct is used to print out the filtered data
// (and can also be used to store data from the read files)
// N.B. a previous version of this struct used float rather than
// double values: if you had trouble with truncating data points,
// change ALL of your variables that used to be float to be double
typedef struct obstacleDataSample_struct {
    double timestamp;
    double distance;
    double angle;
    FilterStatus status;
} obstacleDataSample;
```

You may define other structures and types as you see fit.

Filter Algorithm

- We will filter the distance data only
- The filter is the average of all distance points across the width of the filter
    - We define, at compile time, the width to be 11 points
    - The filter will be run ONLY if the full length of the window is available
        - Else, the existing data is used
- Angle data is NOT filtered, but are used to determine if the sample is valid

- A sample is valid if
    - The relative distance is strictly less than the tolerance (i.e., 0.1 meters)
    - The relative angle is less than 15 degrees
        - Angle data is given in radians. Possibly need to covert to degrees, or visa versa (to check if within tolerance). Consider using M_PI constant from cmath library

- The first element is always labeled as UNDEFINED
- If the sample has the default angle then label as UNDEFINED
- If the sample can be used for filtering, but is not, then label as VALID
- If the angle between two points exceeds the angle maximum tolerance, then label as ANGLE RESET
- If a sample is within angular tolerance but the distance between two points exceeds the distance maximum tolerance then label the second point as DISTANCE RESET
- If a sample is not within either distance or angle tolerance of a pervious sample and is not maximum distance or default angle, then label as ANGLE RESET

Output File Format

- The output file should be formatted the same as the input file where each line of text is followed by a new line
    - Each additional line will have one additional entry indicating the status of that point
        - VALID, FILTERED, ANGLE RESET, DISTANCE RESET, UNDEFINED
    - Given as text outputs as they appear
- Numerical values should be printed using a set precision of 6, i.e., `setprecision(6)`.

Project 1 Notes

- I will not require the TA or ULA to help unless you bring @ minimum a flowchart

- I will ask you to draw a flowchart before assisting if you show up for help without one

ZyBooks Lecture: Structures