Exercise 2: Decoding CAN data

Alexander Rasch
alexander.rasch@chalmers.se

September 9, 2021

Introduction

In this assignment you will apply your knowledge on the Controller Area Network (CAN) bus protocol to decode real-world data. Specifically, this assignment will guide you into decoding some signals (collected with different sampling rates from different CAN frames) and synchronize them. In this assignment you will be using MATLAB. You will be working with different types/classes of MATLAB variables (e.g. table, cell). An overview of the fundamental MATLAB classes can be found here: `https://se.mathworks.com/help/matlab/matlab_prog/fundamental-matlab-classes.html`.

Note: You are encouraged to discuss your solutions with other groups but you must write and submit your own code implementation and answers in your group.

Deadline: September 23, 2021 (23:59)

Learning objectives

After having performed this assignment, you shall be able to:

- Read a CAN data log into MATLAB
- Select data frames
- Identify the value of a specific signal within a data frame
- Reconstruct a signal time series from different data frames
- Synchronize signals from different data frames sent at different frequencies

Preparations

1. Download the material from Canvas. `CANdata.mat` is the file containing the CAN data to decode. `Exercise.m` is the template script to solve the exercise.

2. Read the additional information given in the material to understand the CAN bus data format.
3. Open `Exercise.m`, fill in your group number in the header of the file, and start the tasks below.

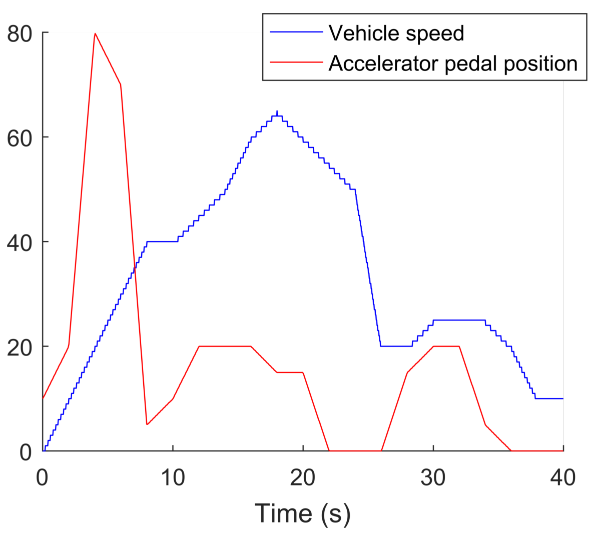
Tasks

1. Import the data from `CANdata.mat`.
2. Decode the signal for vehicle speed and accelerator pedal position. To do so, you would need to (also see Table 1 in the additional information document):
   (a) Select the "right" CAN frame
   (b) Identify the "right" bytes
   (c) Convert the data from hexadecimal to binary
   (d) Identify the "right" bits
   (e) Convert the data from binary to decimal
   (f) Apply a gain factor and an offset if necessary
3. Plot both signals with respect to their timestamps. Include labels and legend. If successful, you should obtain something like Figure 1a.
4. Then, synchronize the signal for vehicle speed and accelerator pedal position and resample them at 1 Hz. Finally, plot the results, together with the original (i.e., not down sampled) signals. If successful, you should obtain something like Figure 1b. To highlight the effect of down sampling the signals, limit the x- and y-axis of the figure to [28, 36] and [15, 30], respectively.
5. Imagine a simple active safety system that issues a warning to the driver when the driver is about to exceed the speed limit of the road. Use the synchronized speed and time signals and implement a system that warns a driver of exceeding the given speed limit (50 km/h) by 10 km/h (tolerance), i.e. a system that shall prevent the driver of having greater speed than 50 + 10 = 60 km/h. Take into account the driver reaction time of $\Delta t_\mathrm{R} = 1.0$ s to predict the speed of the driver at any time index k, given a constant acceleration model, i.e. the acceleration is assumed constant during the reaction time:

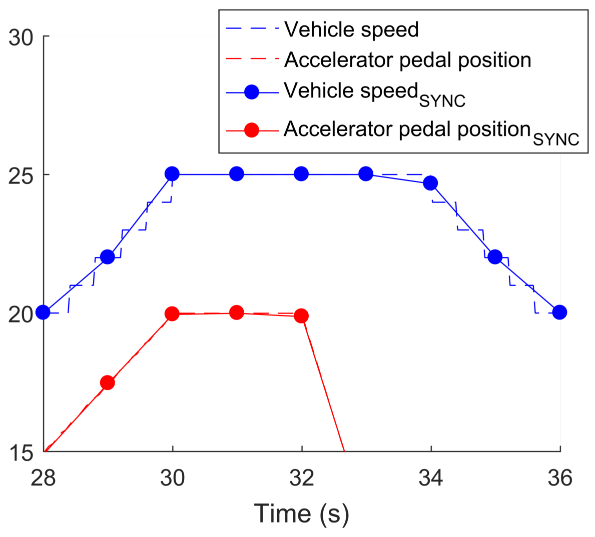
   $$v_{\mathrm{pred},k} = v_k + a_k \cdot \Delta t_\mathrm{R}, \tag{1}$$

   where v_pred is the predicted speed, v, a speed and acceleration at the current time index k, respectively. At what time would the system activate the warning ($v_\mathrm{pred} > 60$ km/h)?

   Hint: Derive the acceleration signal simply by discrete differentiation of the speed signal (you can just use the `diff()` command in MATLAB as sample time is 1 s), and prepending 0 as the first measurement. Also note that v and a in Eq. (1) need to be in m/s and m/s^2, respectively.



(a) Decoded signals



(b) Synchronized signals

Figure 1: Vehicle signals obtained from CAN data.

Submission

1. Run and submit your code in MATLAB Grader to verify that the output is correct (recommended, not mandatory).
2. Submit your solutions (`Exercise.m`) in the assignment in Canvas. It is sufficient if at least one group member submits your solution in Canvas. See deadline above.