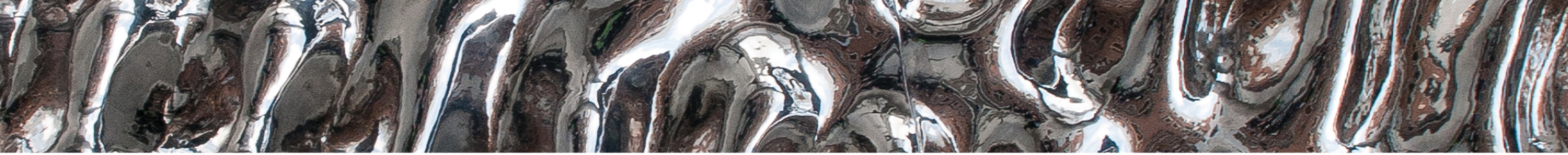


UiO : Department of Informatics
University of Oslo

Digital Services based on Vehicle Data

SCOTT Vehicle Data Hackathon

<Team Name=" White Hats "

Members="Toktam Ramezanifarkhani, Peyman Teymoori"

Affiliation="University of Oslo"

Email="{toktamr, peymant}@ifi.uio.no"

/>

Outline

- Introduction
- Proposed services
  - fuel consumption optimization
  - driving style
  - road statistics
  - security (authentication)
  - optimum placement of gas/charging stations
  - car safety
  - alerts/information
- Conclusion

Data is Gold!

- Vehicle CAN bus data provides a rich dataset
- Driving can be safer and more economic
- Information derived from cars can flow faster to stakeholders with an information system
- More meaningful decisions can be made

Stakeholders

- Drivers
- Road maintainers
- Car companies
- Insurance companies
  - insurance based on the driving region/behavior

Proposed Services

1. Fuel consumption optimization
2. Driving style
3. Road statistics
4. Security (authentication)
5. Optimum placement of gas/charging stations
6. Car safety
7. Alerts/information

1.1. Deceleration Alert

- No acceleration is usually needed while approaching red lights, roundabouts, roads with a lower speed limit, etc.
- The above limiting elements can be obtained from GPS map data, online information (if the car is enabled with 4G for example)
- Car can show an alert to the driver not (no need) to accelerate in these situations
- This can optimize fuel consumption

1.1. Deceleration Alert: Description

- Can be implemented in the car computer system
- Actor: driver
- Description:
  - get location and direction info from GPS
  - if approaching one of the aforementioned elements, and acceleration is not negative, then show/hint decrease speed signal

1.1. Deceleration Alert: Demonstration

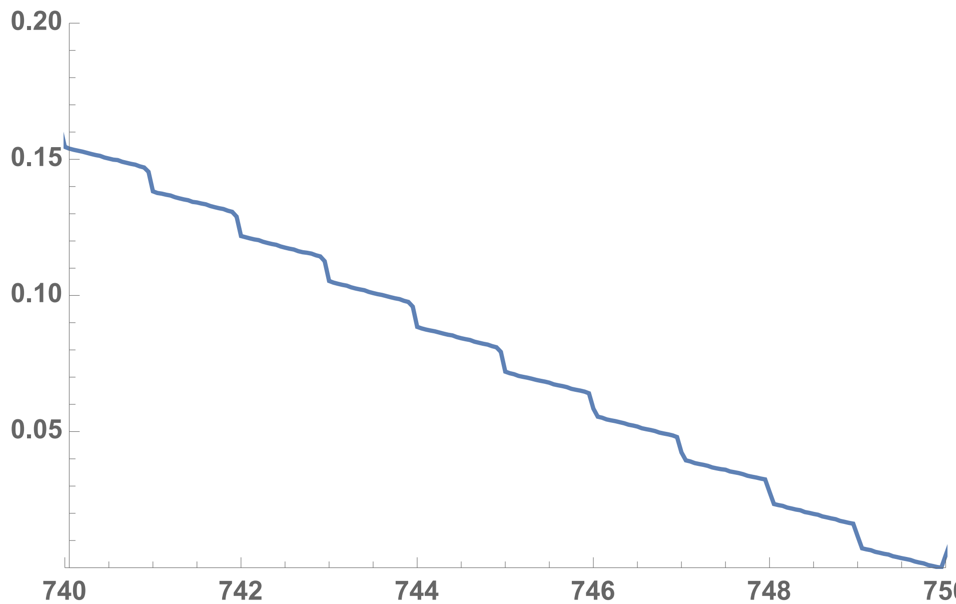
- Code in Mathematica [*]:

```
CleanSlate[];
ClearAll["Global`*"];
ac = Import["Downloads/20181116_Driver1_Trip3.hdf", {"Datasets", "/GPS/Acceleration"}];
la = Import["Downloads/20181116_Driver1_Trip3.hdf", {"Datasets", "/Math/Latitude_IMU"}];
lo = Import["Downloads/20181116_Driver1_Trip3.hdf", {"Datasets", "/Math/Longitude_IMU"}];
n1 = 740; n2 = 750; t = 750;
c1 = la[[t*20]][[1]]; c2 = lo[[t*20]][[1]];
d[p1_, p2_, i_] := Sqrt[(p1 - la[[i]][[1]])^2 + (p2 - lo[[i]][[1]])^2];
s[a_] := If[a > 0, 1, 0];
l = Table[{la[[i]][[2]], d[c1, c2, i]}, {i, 1, Length[la]}];
ListPlot[l, Joined → True, PlotRange → {{n1, n2}, {0, 0.2}}]
ListPlot[Reverse[ac, 2], Joined → True, PlotRange → {{n1, n2}, Automatic}]
l = Table[s[ac[[i]][[1]]], {i, 1, Length[ac]}];
ListPlot[l, Joined → True, PlotRange → {{n1, n2}, Automatic}]
```


[*] Wolfram Research, Inc., Mathematica, Version 11.0, Champaign, IL (2017).

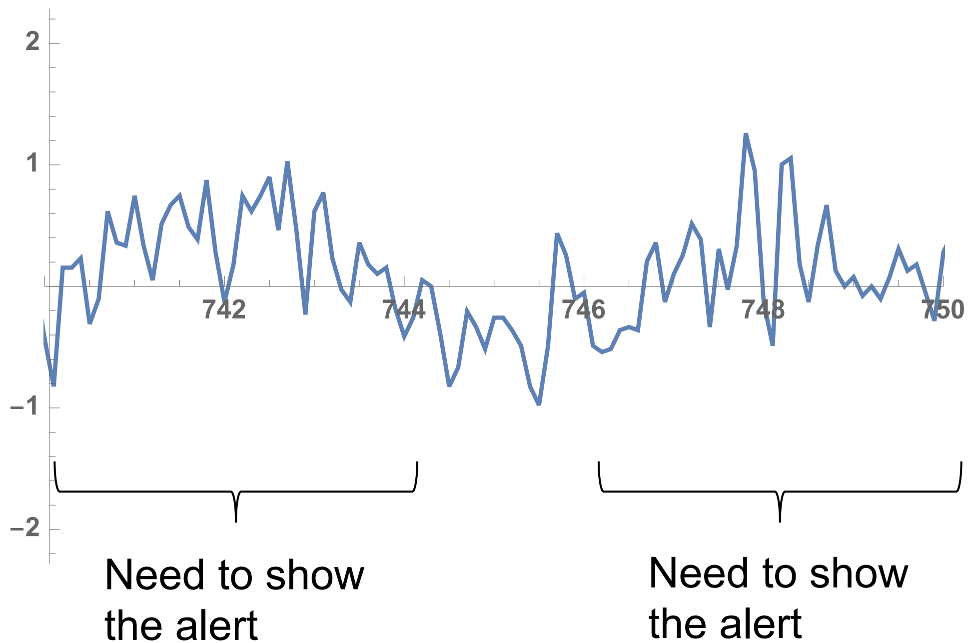
1.1. Deceleration Alert: Demonstration

- Assume that driver 1 reaches a limiting element in trip 3 at second 750
- The code first plots the Euclidean distance to the limit:



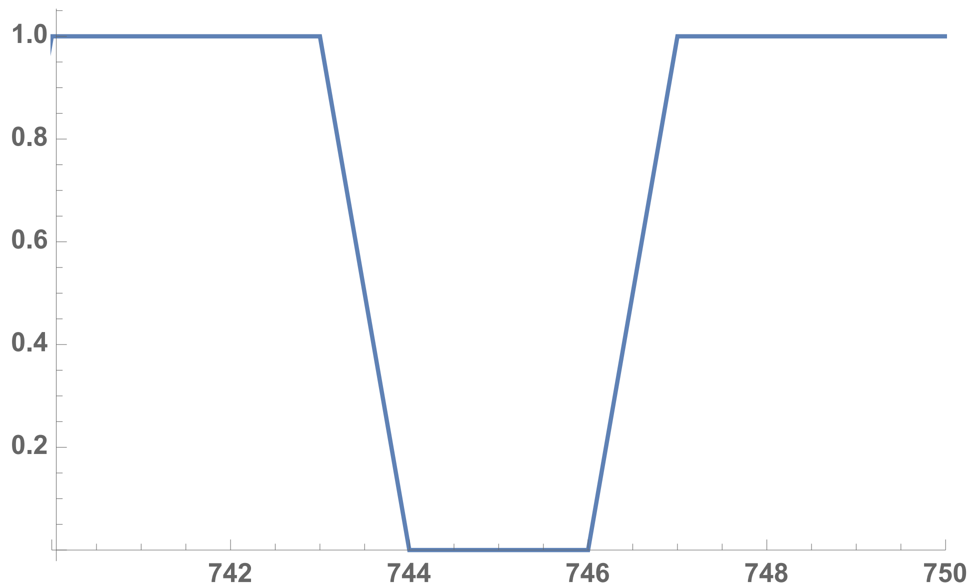
1.1. Deceleration Alert: Demonstration

- It then obtains acceleration values and plots them:



1.1. Deceleration Alert: Demonstration

- If acceleration is positive, then a signal is produced
- The signal plot: (can be shown as a light in the car dashboard)



1.2. Economical Driving Mode

- When driving downhill, usually there is no need to use more fuel consumption modes
- The gravity can help accelerate!
- The car can automatically switch to a more economical mode
- The modes can be defined with more granularity
  - depending on the slope, the most appropriate one is used

1.2. Economical Driving Mode: Description

- Can be implemented in the car computer system
- Actor: driver
- Description:
  - if the route is known (e.g. the driver is following a route on GPS map) and the car is driving downhill, or
  - if the route is unknown and the altitude is decreasing in the past, for example, 10 seconds, then
    - switch to the more economical mode
  - the mode can be dependent on the slope in case there are more fine-grained modes defined in the car

1.2. Economical Driving Mode: Demonstration

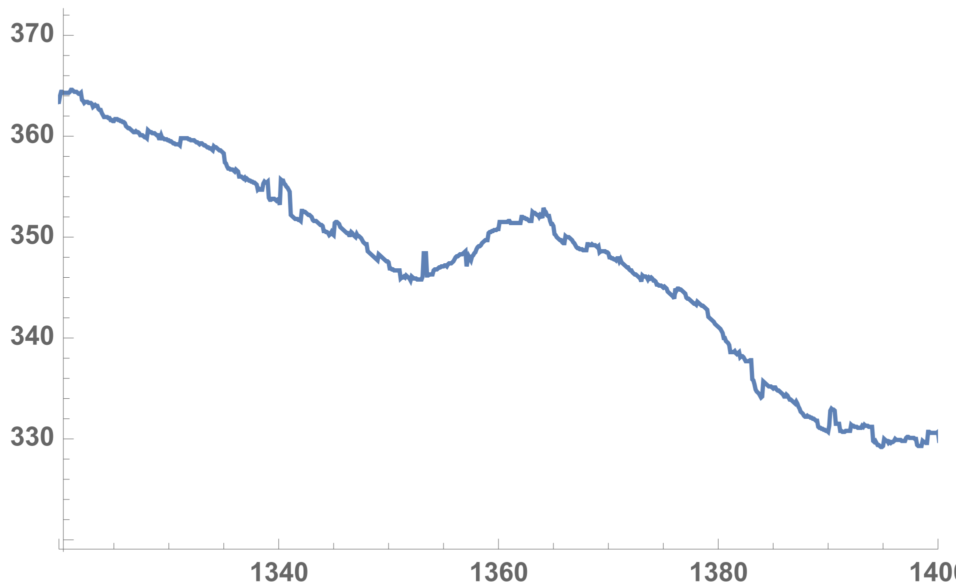
- Code in Mathematica [*]:

```
CleanSlate[];
ClearAll["Global`*"];
n1 = 1320; n2 = 1400; w = 10;
z = Import["Downloads/20181116_Driver1_Trip3.hdf", {"Datasets", "/GPS/Z"}];
y = Reverse[z, 2];
v = MovingAverage[y[[All, 2]], w*20];
l = Table[{y[[i + w*20 - 1]][[1]], v[[i]]}, {i, 1, Length[v]}];
ListPlot[y, Joined → True, PlotRange → {{n1, n2}, Automatic}]
ListPlot[l, Joined → True, PlotRange → {{n1, n2}, Automatic}]
s = Table[If[l[[i]][[2]] - l[[i - 1]][[2]] < 0, {l[[i]][[1]], 1}, {l[[i]][[1]], 0}], {i, 2, Length[l]}];
ListPlot[s, Joined → True, PlotRange → {{n1, n2}, Automatic}]
```


[*] Wolfram Research, Inc., Mathematica, Version 11.0, Champaign, IL (2017).

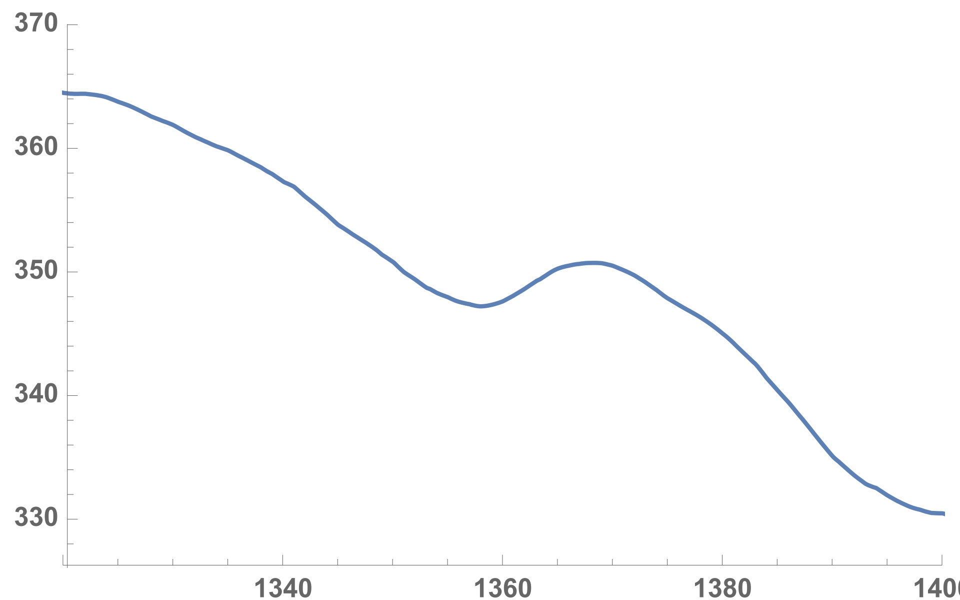
1.2. Economical Driving Mode: Demonstration

- The code obtains and plots altitude from GPS:



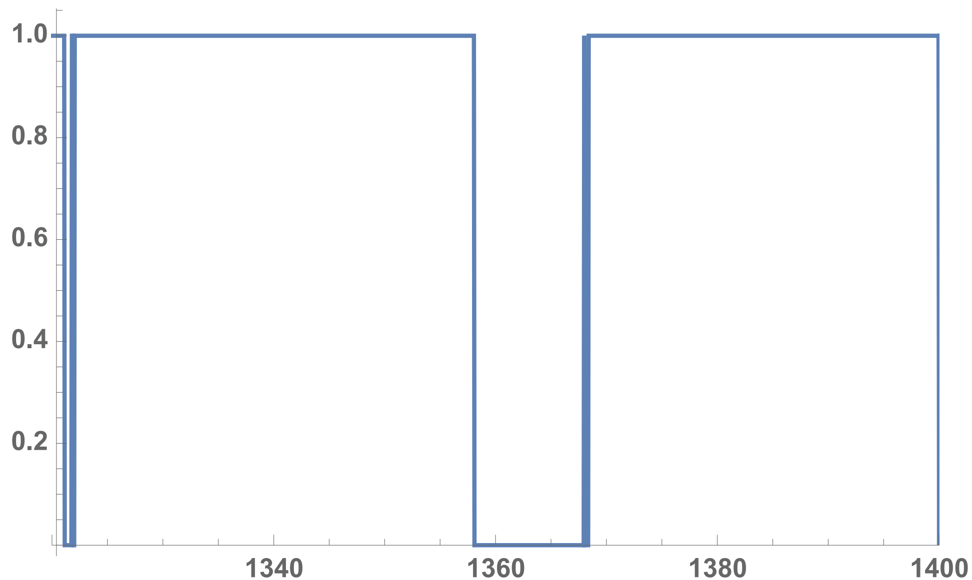
1.2. Economical Driving Mode: Demonstration

- Since variation is not small in the values, the moving average of altitude is calculated:



1.2. Economical Driving Mode: Demonstration

- The switching to a more economical mode signal:
  – "1" means showing the signal to the driver or automatically switching to a more economical mode



Proposed Services

1. Fuel consumption optimization
2. Driving style
3. Road statistics
4. Security (authentication)
5. Optimum placement of gas/charging stations
6. Car safety
7. Alerts/information

2.1. Acceleration Distribution

- Acceleration (including both positive and negative values) is a combined metric of pressing the gas and brake pedals
- Can reflect the driving style
- Can be used as a means of identifying the driver
- Identify the risk level of the driver for insurance companies

2.1. Acceleration Distribution: Description

- Can be implemented in the car computer system
- Actor: driver identification module, insurance companies, etc.
- Description:
  – obtain acceleration values
  – calculate statistical values of the data, e.g. mean, variance
    - the distribution can also be obtained

2.1. Acceleration Distribution: Demonstration

- Code:

```
CleanSlate[];
ClearAll["Global`*"];
z = Import["Downloads/20181113_Driver1_Trip2.hdf", {"Datasets", "/GPS/Acceleration"}];
z = Reverse[z, 2];
Print[Min[z[[All, 2]]], "\t", Mean[z[[All, 2]]], "\t", Max[z[[All, 2]]], "\t", Variance[z[[All, 2]]]]
z = Import["Downloads/20181116_Driver1_Trip3.hdf", {"Datasets", "/GPS/Acceleration"}];
z = Reverse[z, 2];
Print[Min[z[[All, 2]]], "\t", Mean[z[[All, 2]]], "\t", Max[z[[All, 2]]], "\t", Variance[z[[All, 2]]]]
z = Import["Downloads/20181114_Driver2_Trip1.hdf", {"Datasets", "/GPS/Acceleration"}];
z = Reverse[z, 2];
Print[Min[z[[All, 2]]], "\t", Mean[z[[All, 2]]], "\t", Max[z[[All, 2]]], "\t", Variance[z[[All, 2]]]]
z = Import["Downloads/20181114_Driver2_Trip2.hdf", {"Datasets", "/GPS/Acceleration"}];
z = Reverse[z, 2];
Print[Min[z[[All, 2]]], "\t", Mean[z[[All, 2]]], "\t", Max[z[[All, 2]]], "\t", Variance[z[[All, 2]]]]
z = Import["Downloads/20181114_Driver2_Trip3.hdf", {"Datasets", "/GPS/Acceleration"}];
z = Reverse[z, 2];
Print[Min[z[[All, 2]]], "\t", Mean[z[[All, 2]]], "\t", Max[z[[All, 2]]], "\t", Variance[z[[All, 2]]]]
```

2.1. Acceleration Distribution: Demonstration

- Results:

		Min	Mean	Max	Variance
Driver 1	Trip 2	-57.61	0.0003	28.78	0.75
	Trip 3	-8.46	0.0014	7.94	0.62
Driver 2	Trip 1	-24.31	0.0020	35.21	1.34
	Trip 2	-34.36	0.0036	30.82	2.23
	Trip 3	-27.28	0.0005	37.96	0.46

Driver 1: More stable in acceleration

Driver 2: More variations in acceleration

2.1. Acceleration Distribution: Demonstration

- Finding a distribution function fitting the data:
  - with the FindDistribution command

		Function
Driver 1	Trip 2	StudentTDistribution[0.0177689, 0.242434, 1.45387]
	Trip 3	LaplaceDistribution[0.00125023, 0.519172]
Driver 2	Trip 1	MixtureDistribution[{0.171996, 0.828004}, {LogisticDistribution[-0.0101865, 0.492662], CauchyDistribution[-0.025938, 0.21569]}]
	Trip 2	MixtureDistribution[{0.15552, 0.84448}, {LaplaceDistribution[0.742762, 1.52849], CauchyDistribution[-0.016463, 0.152987]}]
	Trip 3	MixtureDistribution[{0.263411, 0.736589}, {LogisticDistribution[0.0400912, 0.182406], CauchyDistribution[-0.100337, 0.37527]}]

2.2. Hard Brakes

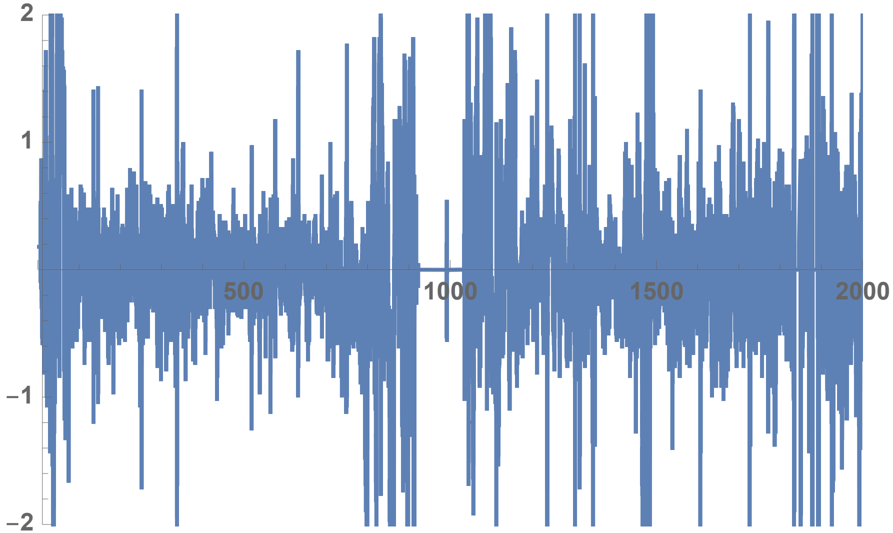
- When (negative) acceleration is smaller than a threshold
  - several threshold values can be defined: e.g. hard brakes, very hard brakes
- Might lead to a full stop!
- Can reveal
  - the driving style
  - problems with the road design, road constructions, wrong placement of road signs, etc. (when combined with GPS locations)
- Can be used as a way of identifying the driver

2.2. Hard Brakes: Description

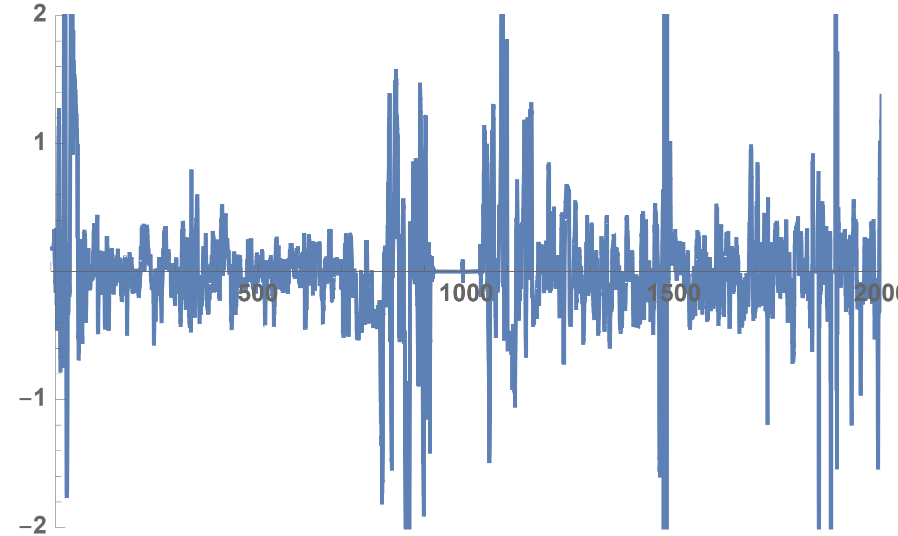
- Can be implemented in the car computer system
- Actor: driver identification module, insurance companies, road maintenance authorities, etc.
- Description:
  - obtain acceleration and speed values
  - calculate the moving average on acceleration (window size: 1 sec)
  - if (the moving avg. of) acceleration is smaller than the threshold:
    - count as a hard brake
    - if the speed becomes zero, also count as hard brake causing full stop
    - obtain the GPS location for further investigations

2.2. Hard Brakes: Demonstration

- Code is a bit large!
- Results for driver 1 trip 2:



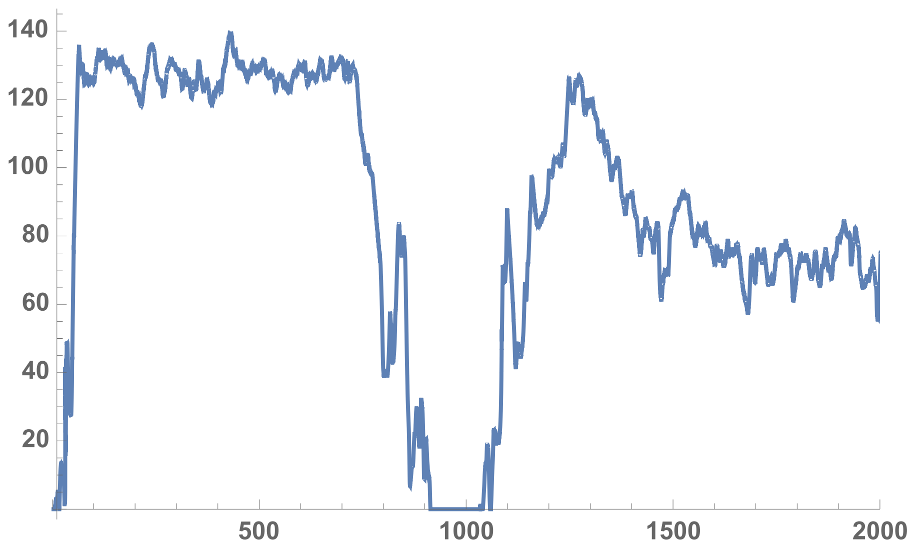
Acceleration



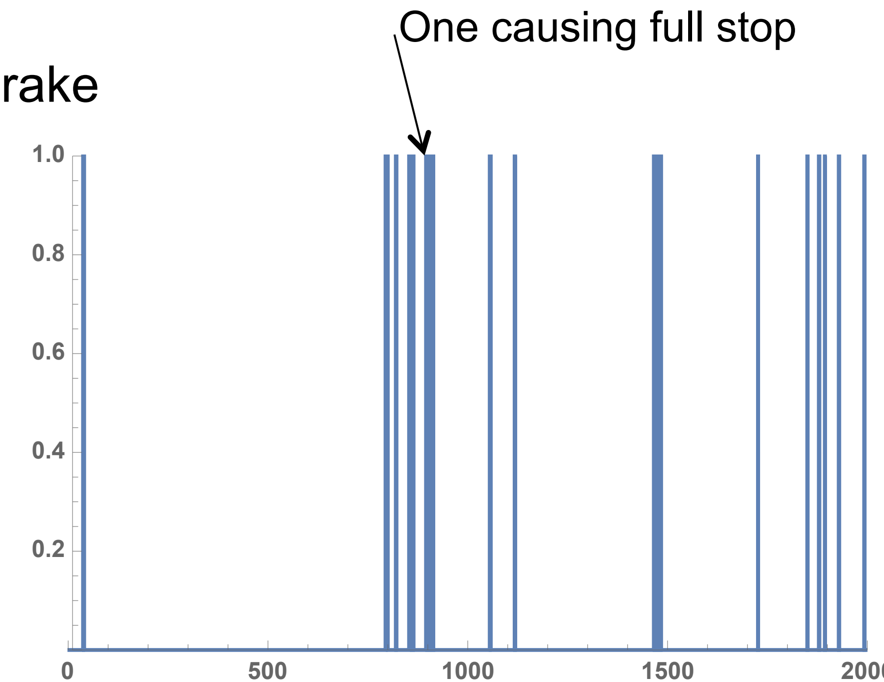
Moving avg. of acceleration

2.2. Hard Brakes: Demonstration

- Results:
  - “1” means there was a hard brake



Speed

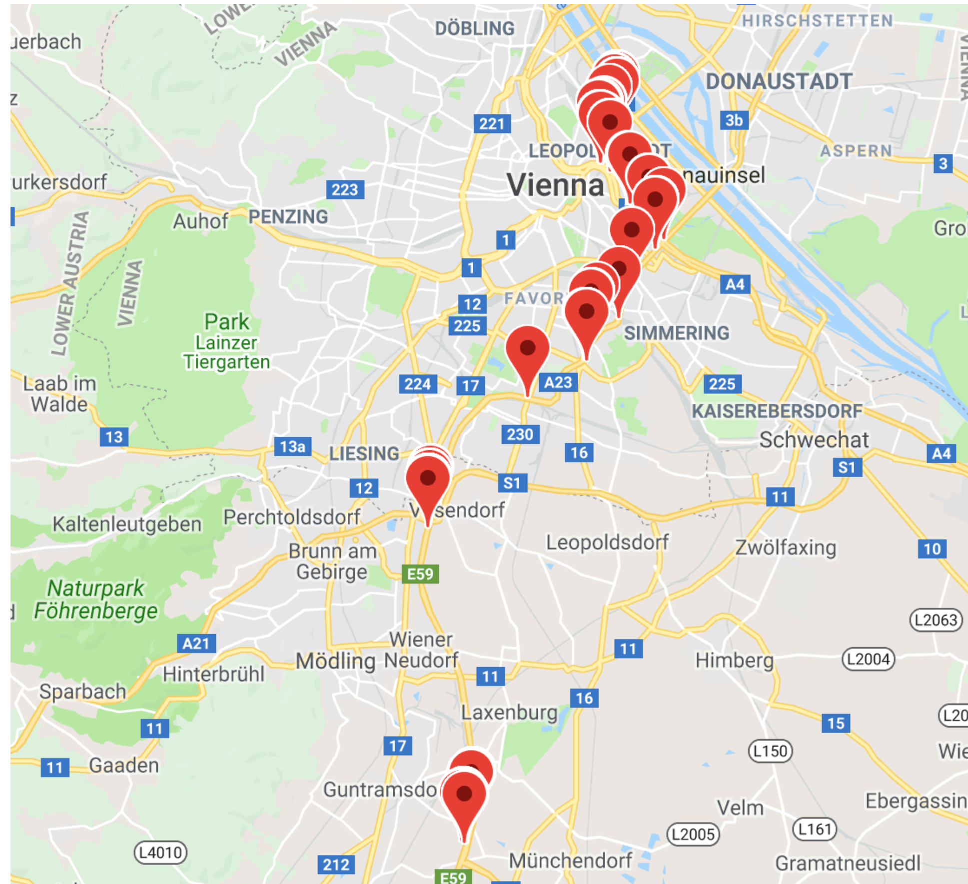


Hard brakes

2.2. Hard Brakes: Demonstration

Combining with GPS locations and integrating with Google maps showing where those happened:

zoomable for further details



2.3. Sharp Turns

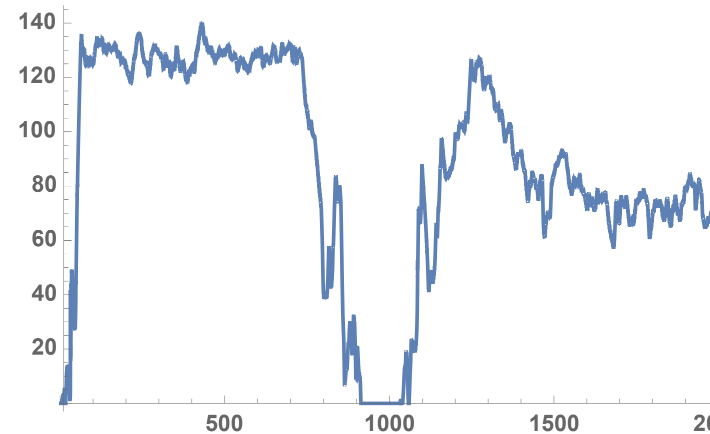
- Combining the steer angle and the vehicle speed can make another metric!
- Can reveal
  - the driving style
  - problems in the road design, road constructions, wrong placement of road signs, etc. when combined with GPS locations
- Can be used as a way of identifying the driver
- "Sharp" can be defined as a function of the steer angle and speed.
  - For example: if $\text{angle} > 540\, e^{(-0.025\ \text{speed})}$, then it's sharp!

2.3. Sharp Turns: Description

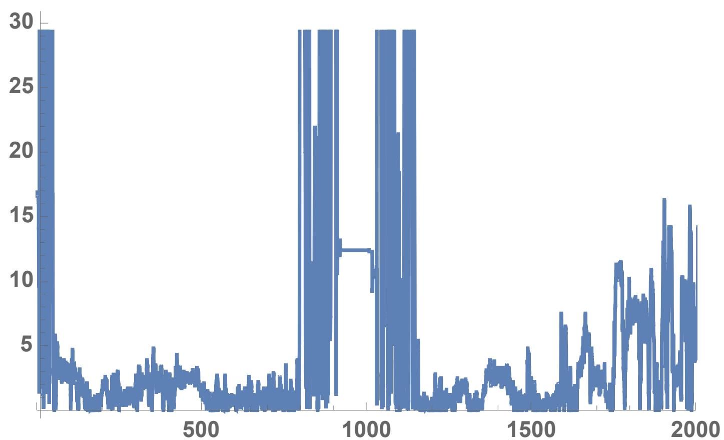
- Can be implemented in the car computer system
- Actor: driver identification module, insurance companies, road maintenance authorities, etc.
- Description:
  - obtain steer angle and speed values
  - if the function defined before returns 1 for any pair of speed and steer angle values, then mark the GPS location

2.3. Sharp Turns: Demonstration

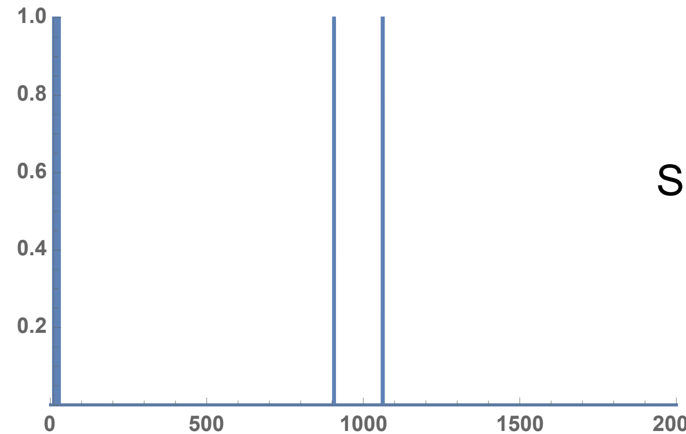
- Code is a bit large!
- Results for driver 1 trip 2:



Speed



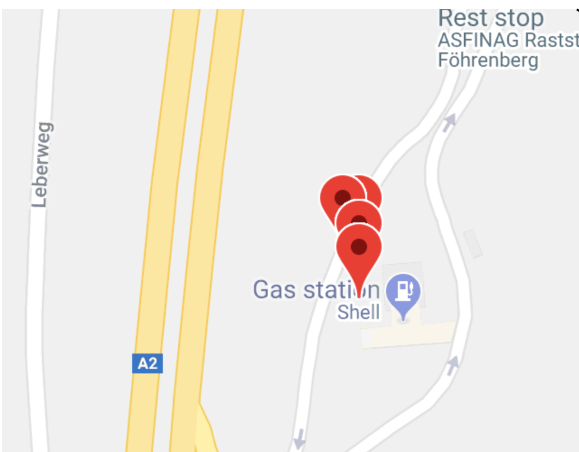
Steer angle

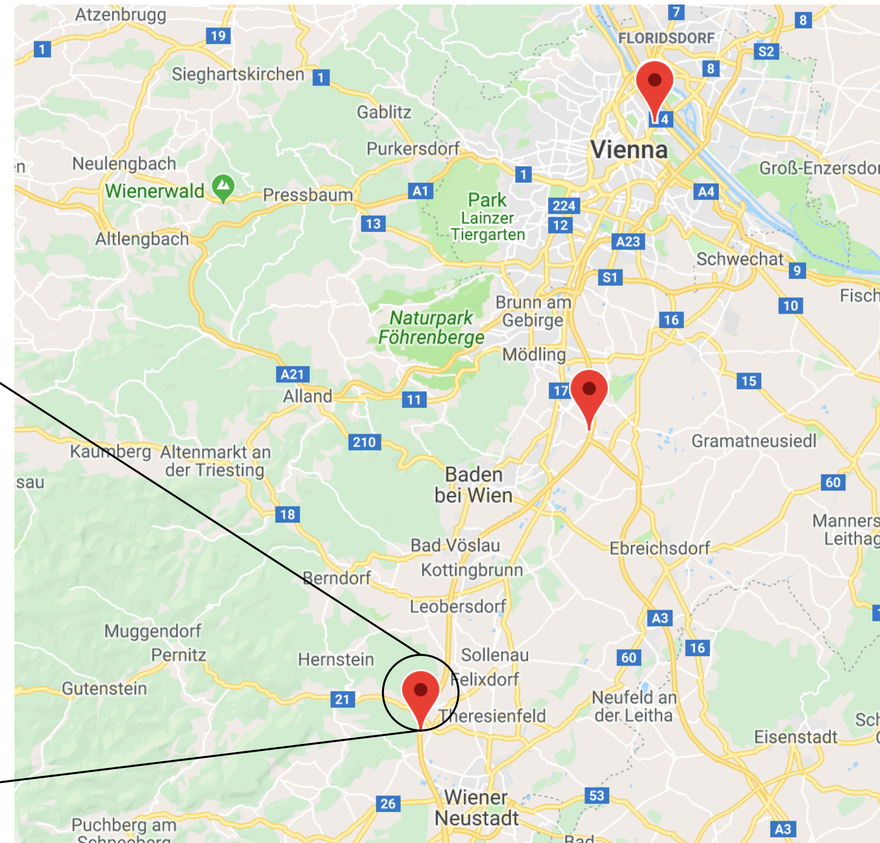


Sharp turns

2.3. Sharp Turns: Demonstration

Combined with Google maps showing where those happened





Proposed Services

1. Fuel consumption optimization
2. Driving style
3. Road statistics
4. Security (authentication)
5. Optimum placement of gas/charging stations
6. Car safety
7. Alerts/information

Road Statistics: Proposed Services

- The number of hard brakes at certain points
  - shown before
- The number of sharp turns at certain points
  - shown before
- The number of part failures at certain points
  - problems such as crash, flat tire, etc.
  - can reflect a problem with the road

Road Statistics: Proposed Services

- Matching the speed limit and cars’ speed in roads
  - might need to reconsider the limit, or force new prohibitory means such as speed cameras
- Implementation could be done via a cloud-based system
  - cars upload their statistics
  - data is processed and the required information is derived
- The last two services were not done in this demo since they need more information/part failure signals

Proposed Services

1. Fuel consumption optimization
2. Driving style
3. Road statistics
4. Security (authentication)
5. Optimum placement of gas/charging stations
6. Car safety
7. Alerts/information

Security: Proposed Services

- Learning the driver behavior based on the derived statistics to have an additional means of authentication
  - overcome the challenge of passive keyless entry
- Machine learning can be based on different data
  - acceleration
  - sharp turning
- Different learning techniques can be used

4.1. Driver Authentication

- We use acceleration as a means of identifying the driver
- Acceleration includes values of pressing the gas and brake pedals
- Can be used as a service in the car to learn the driver’s behavior. In case of detecting a violation from the common behavior, the car can, for example, inform the owner

4.1. Driver Authentication: Description

- Can be implemented in the car computer system
- Actor: driver identification module
- Description:
  - obtain acceleration values per driver per trip
  - calculate their statistical values such as min, max, mean, 25% and 75% quantiles.
  - choose an appropriate learning technique
  - use part of data per driver as the training set
  - use the remaining part of data as the test set

4.1. Driver Authentication: Demonstration

- Code: training using the drivers' trip 2 and trip 3
- Depending on the data, it automatically chooses a learning method, e.g. regression, SVM, Neural Networks

```
trainingset = {{Min[y2], Quantile[y2, 0.25], Median[y2], Quantile[y2, 0.75], Max[y2]} → "Driver1",
   {Min[y3], Quantile[y3, 0.25], Median[y3], Quantile[y3, 0.75], Max[y3]} → "Driver1",
   {Min[z2], Quantile[z2, 0.25], Median[z2], Quantile[z2, 0.75], Max[z2]} → "Driver2",
   {Min[z3], Quantile[z3, 0.25], Median[z3], Quantile[z3, 0.75], Max[z3]} → "Driver2"};


c = Classify[trainingset]
```

- Testing:

```
c[{Min[z1], Quantile[z1, 0.25], Median[z1], Quantile[z1, 0.75], Max[z1]}, "Probabilities"]
```

4.1. Driver Authentication: Demonstration

- Results: the test returns the probability that a test set belongs to either driver 1 or driver 2
- If we train our model and then, we feed driver 2 trip 1 to our model as the test set, it returns:

```
= ClassifierFunction[ Input type: NumericalVector (length: 5)
                      Classes: Driver1, Driver2
                      Method: NaiveBayes
                      Number of training examples: 4 ]
```

```
= <|Driver1 → 0.00226244, Driver2 → 0.997738|>
```

showing the chosen learning method, and that the trip belongs to driver 2 with probability 0.997738!

This needs more and more training and test sets. We are now working on this as a paper.

Proposed Services

1. Fuel consumption optimization
2. Driving style
3. Road statistics
4. Security (authentication)
5. Optimum placement of gas/charging stations
6. Car safety
7. Alerts/information

Optimum Placement of Gas/Charging Stations

- Using information such as Fuel Consumption in vehicle CAN, authorities can determine how much fuel/electricity has been used on average to reach point B from point A
- This can help decide where and on which side of the road to place facilities such as fuel/charging stations
- This can mitigate the charging problem of short-range electric cars

Since this data was zero in the provided data sets, the demo was not done

Proposed Services

1. Fuel consumption optimization
2. Driving style
3. Road statistics
4. Security (authentication)
5. Optimum placement of gas/charging stations
6. Car safety
7. Alerts/information

Car Safety: Proposed Services

- Ordering car parts in case of imminent failure
  - can be detected from the data
  - needs communication with the service center (e.g. with 4G)
- Ordering a service/maintenance
  - in case the car reaches some limits, e.g. mileage, duration
- Alerting/ordering tire change (getting a flat tire)
  - in case tire sensors (if present) detect a constant pressure loss, they can notify the driver or a service center

Since this data was zero/not available in the provided data sets, the demo was not done

Proposed Services

1. Fuel consumption optimization
2. Driving style
3. Road statistics
4. Security (authentication)
5. Optimum placement of gas/charging stations
6. Car safety
7. Alerts/information

Alerts/Information: Proposed Services

- The driver can be notified with a richer set of alerts/ information derived from the data
- Can be shown as messages on the car dashboard
- Examples of services:
  - alerts showing the driving is not safe
    - when there are sharp turns
      - can be combined with weather information: if it is raining, sharp turning is more dangerous, and has a new definition
    - when there is no need to accelerate
  - alerts obtaining from other drivers' behavior, e.g. alerting when approaching a point at which drivers usually have hard brakes!

Conclusion

- We demonstrated some digital services
  - optimizing fuel consumption, obtaining the driving style, finding road problems, learning the driver behavior for security purposes

- We also proposed services such as placement of stations, car safety, and car alerts
  - can be implemented if more data is provided

- A complete information system utilizing the CAN bus data can be beneficial to almost everyone!

Thanks!