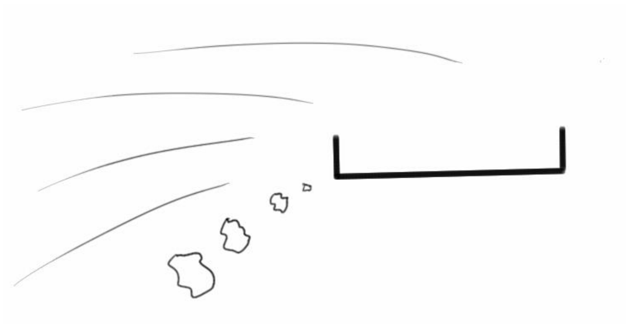


```
 _       ____    _ __                                       ____  _ __      __ 
| |     / / /_  (_) /____  _________  ____ _________       / __ \(_) /___  / /_
| | /| / / __ \/ / __/ _ \/ ___/ __ \/ __ `/ ___/ _ \     / /_/ / / / __ \/ __/
| |/ |/ / / / / / /_/  __(__  ) /_/ / /_/ / /__/  __/    / ____/ / / /_/ / /_  
|__/|__/_/ /_/_/\__/\___/____/ .___/\__,_/\___/\___/    /_/   /_/_/\____/\__/  
                            /_/                                                
```

HSR CHALLENGE 2015

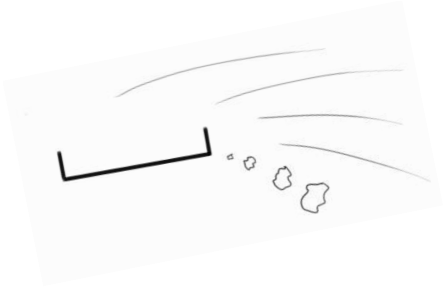
Fast & Furious

Presentation 06.11.15

Agenda

- Team / Organisation
- What we did
- Problems
- Next steps
- Questions / discussion

Whitespace Team:



- Stefan Kapferer

- Roberto Cuervo

Team Organisation

- Both part time students
- Collaboration only possible on sundays and mondays
- Pair Programming

Dev-Env Setup day

- Projects forked
- MVN changes (build without local dependencies)
- CI Setup (Jenkins)
- Own artefact repositories created, were snapshots/releases can be deployed
- Snapshots automatically deployed to the SNAPSHOT-Repository after each commit and successful build

Research:
How can we recognize the track?

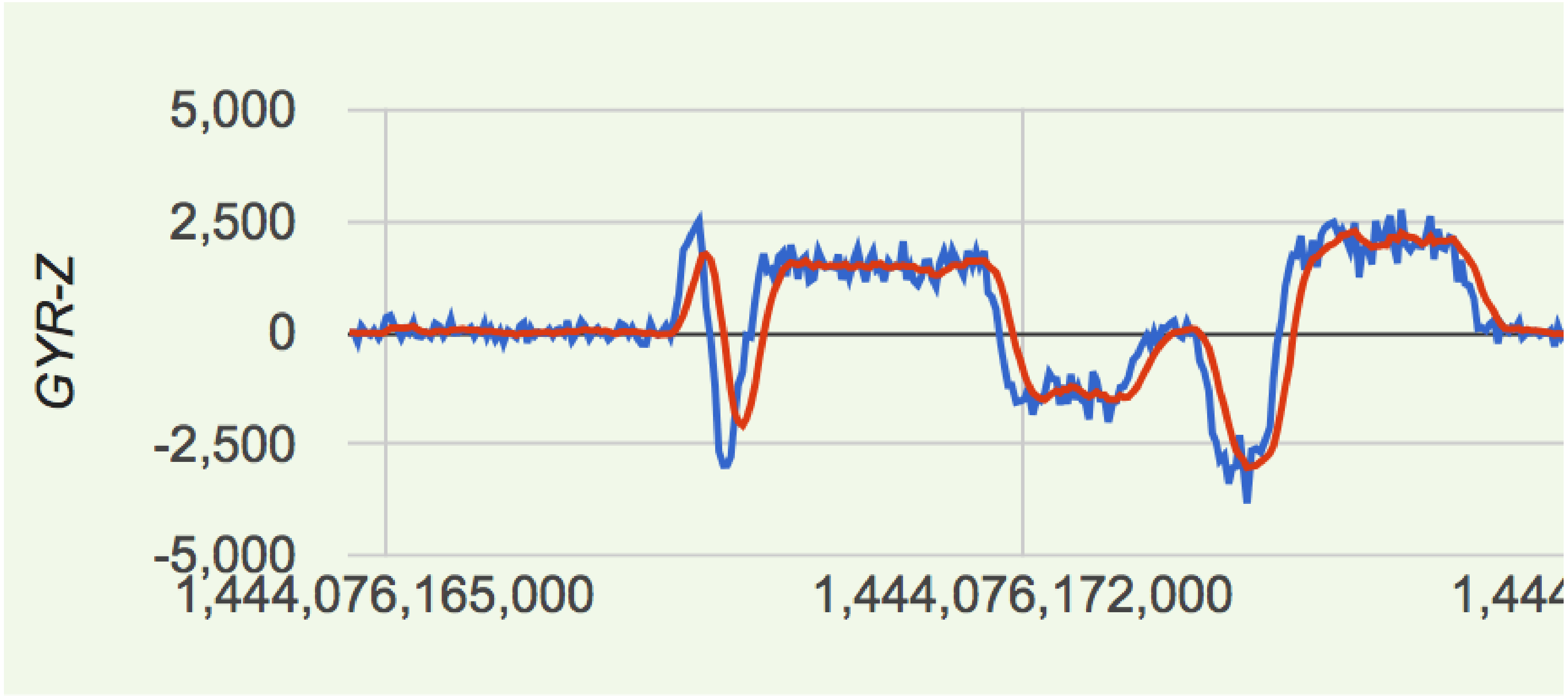
- Gyro Z sensor Data
- Basic idea :
  – Build a function graph.
  – Find the period of the graph.
  – If the signal repeats (period), we know that we passed one round. (it should be possible to recognize this after 2-3 rounds)

Research: How can we recognize the track?

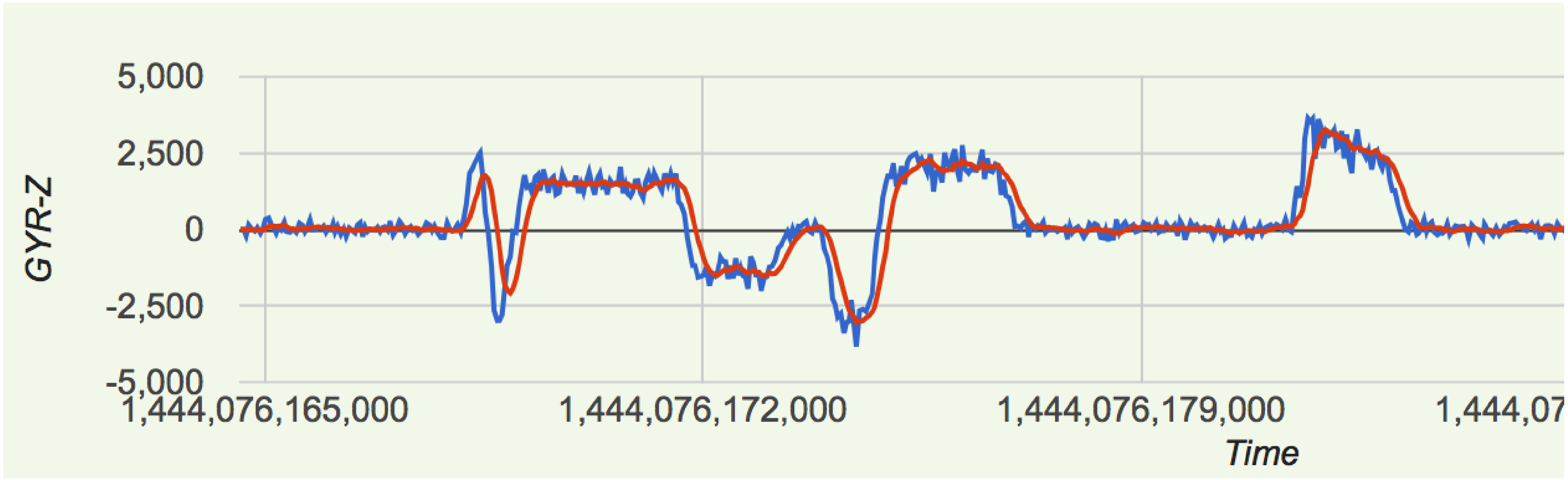
- First idea:
  - Interpolate data to get a function
  - Integrate function to compare regions
- Conversation with Prof. Augenstein
  - Our mathematical approaches were too complex
  - Moving averages to cope with the signal noise
  - How find the track:
    - regularly split the track into two parts and compare them. If they are equal it's maybe a candidate for a match.
  - Just use the signal in order to find out if we drive straight, left or right
  - Pattern like “left-straight-left-left-right…”
  - Search for a repetition in that pattern

Research: How can we recognize the track? (II)

- Data visualization
- Easier to recognize possible patterns



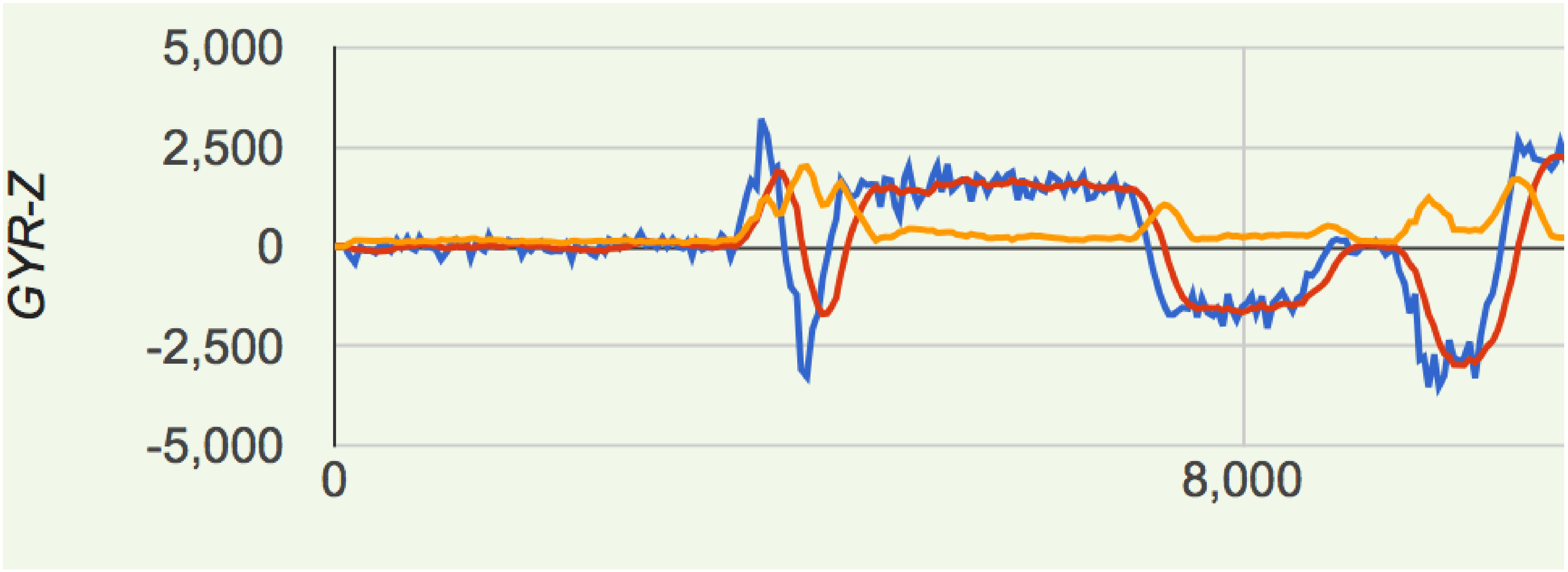
Research: How can we recognize the track? (II)



- Raw data, a.k.a the signal with noise
- In order to recognize the track, it’s necessary to remove the big signal variations
- Moving averages
    - This curve is softer, without big peaks or jumps

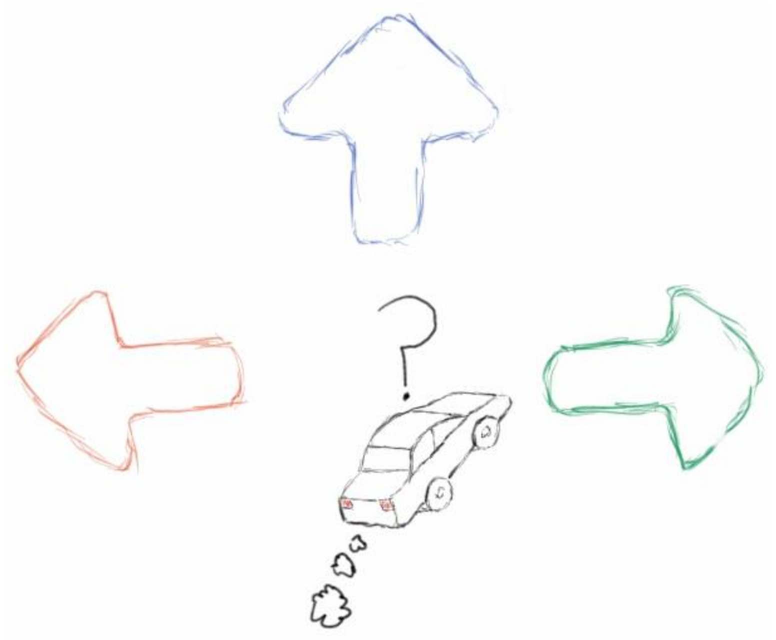
Right-Left-Straight...

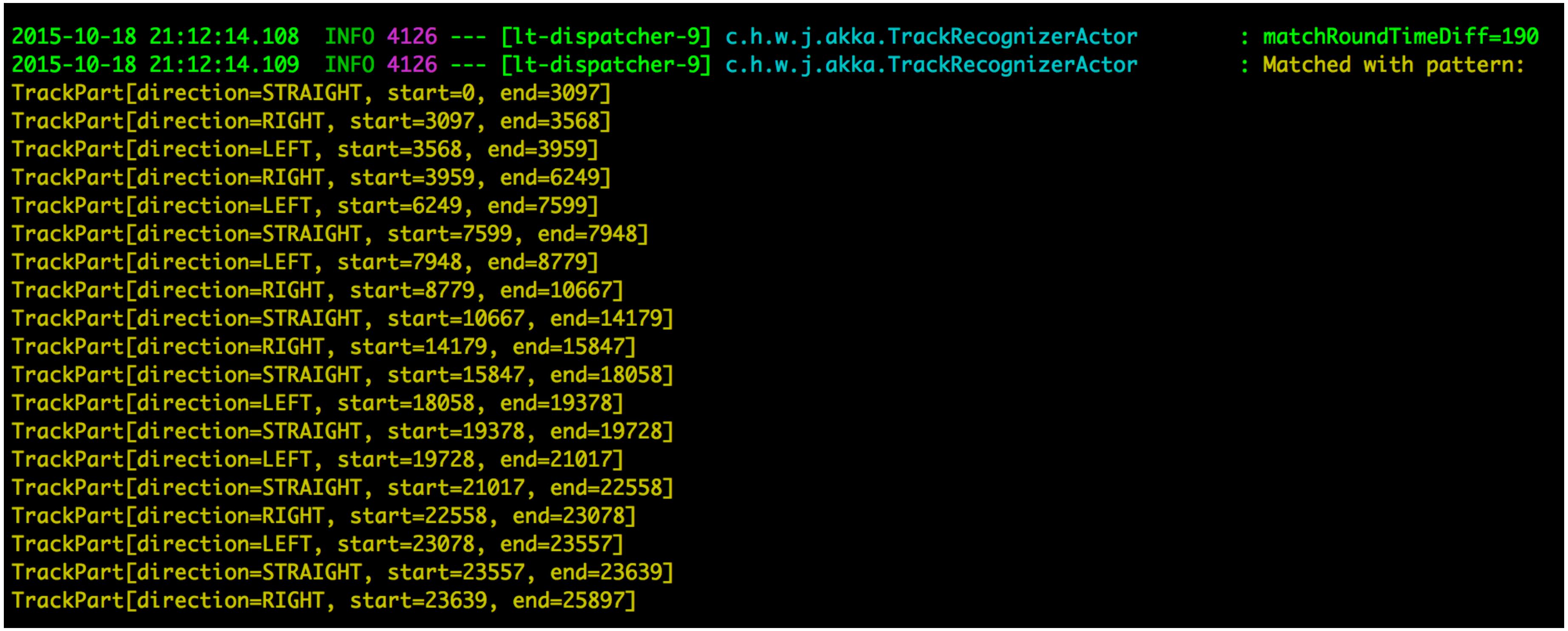
- Standard deviation of the gyro z value
  - Better straights detection



Right-Left-Straight...

- Recognize curves and straights
- Split the track into simple parts
  - TrackPart Class
- Read the values, when they reach certain thresholds, create a new instance of the TrackPart class in which we save the direction, start and end times.
- When the values reach another threshold, we create again the corresponding instance.
- Use the standard deviation in order to avoid the detection of a STRAIGHT-Part between every curve





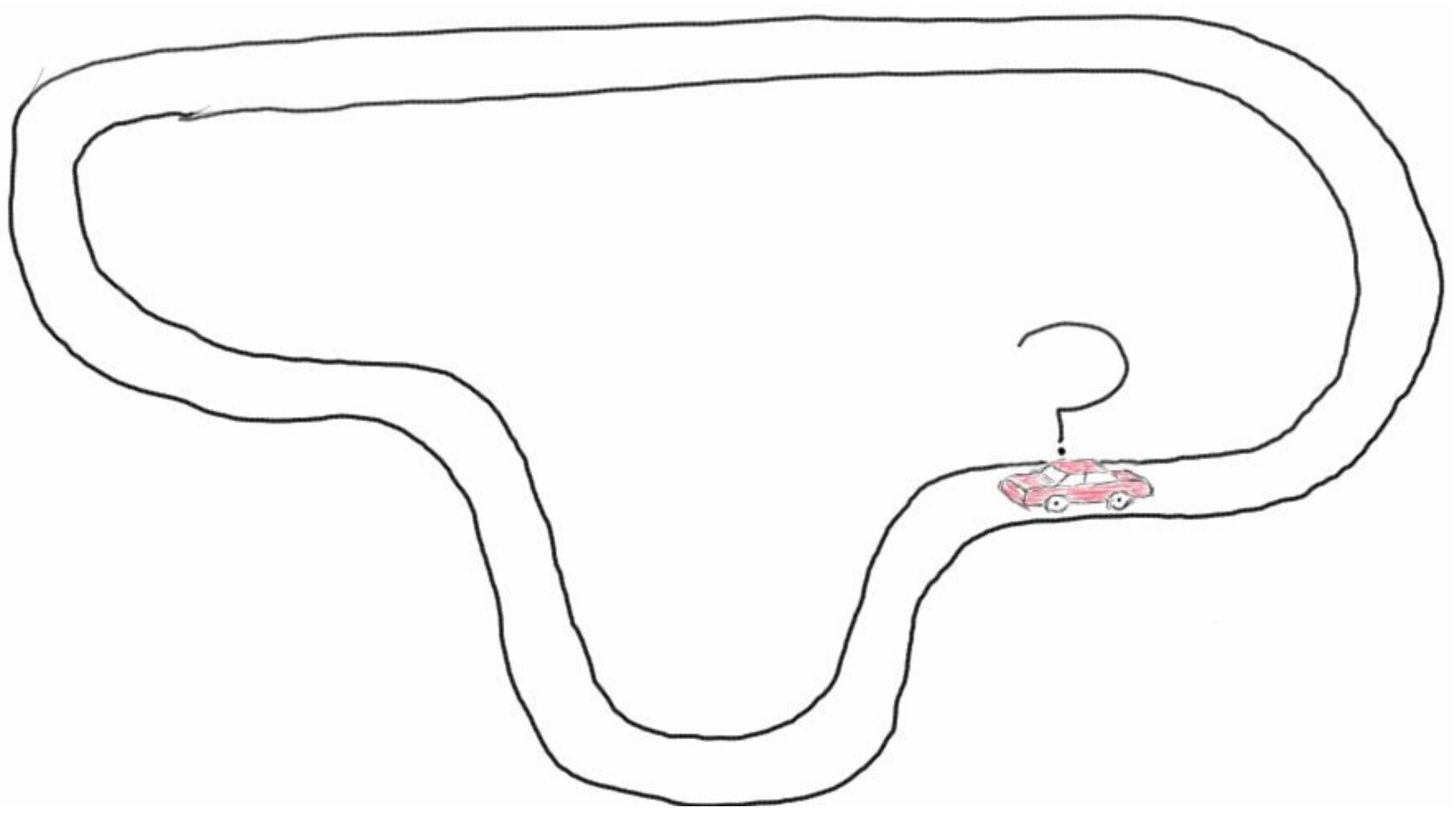
```
2015-10-18 21:12:14.108  INFO 4126 --- [lt-dispatcher-9] c.h.w.j.akka.TrackRecognizerActor        : matchRoundTimeDiff=190
2015-10-18 21:12:14.109  INFO 4126 --- [lt-dispatcher-9] c.h.w.j.akka.TrackRecognizerActor        : Matched with pattern:
TrackPart[direction=STRAIGHT, start=0, end=3097]
TrackPart[direction=RIGHT, start=3097, end=3568]
TrackPart[direction=LEFT, start=3568, end=3959]
TrackPart[direction=RIGHT, start=3959, end=6249]
TrackPart[direction=LEFT, start=6249, end=7599]
TrackPart[direction=STRAIGHT, start=7599, end=7948]
TrackPart[direction=LEFT, start=7948, end=8779]
TrackPart[direction=RIGHT, start=8779, end=10667]
TrackPart[direction=STRAIGHT, start=10667, end=14179]
TrackPart[direction=RIGHT, start=14179, end=15847]
TrackPart[direction=STRAIGHT, start=15847, end=18058]
TrackPart[direction=LEFT, start=18058, end=19378]
TrackPart[direction=STRAIGHT, start=19378, end=19728]
TrackPart[direction=LEFT, start=19728, end=21017]
TrackPart[direction=STRAIGHT, start=21017, end=22558]
TrackPart[direction=RIGHT, start=22558, end=23078]
TrackPart[direction=LEFT, start=23078, end=23557]
TrackPart[direction=STRAIGHT, start=23557, end=23639]
TrackPart[direction=RIGHT, start=23639, end=25897]
```

- Compare the given round time with the added time of all our track parts
- If this difference is smaller than a threshold, we have a best match:

Pattern:

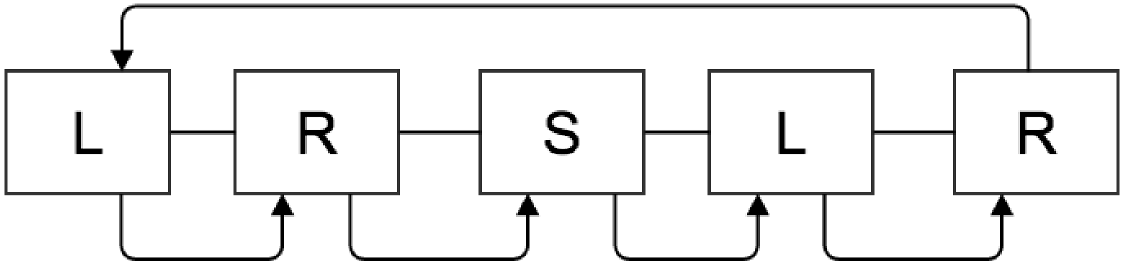
STRAIGHT- RIGHT- LEFT- RIGHT- LEFT- STRAIGHT- LEFT...

Position



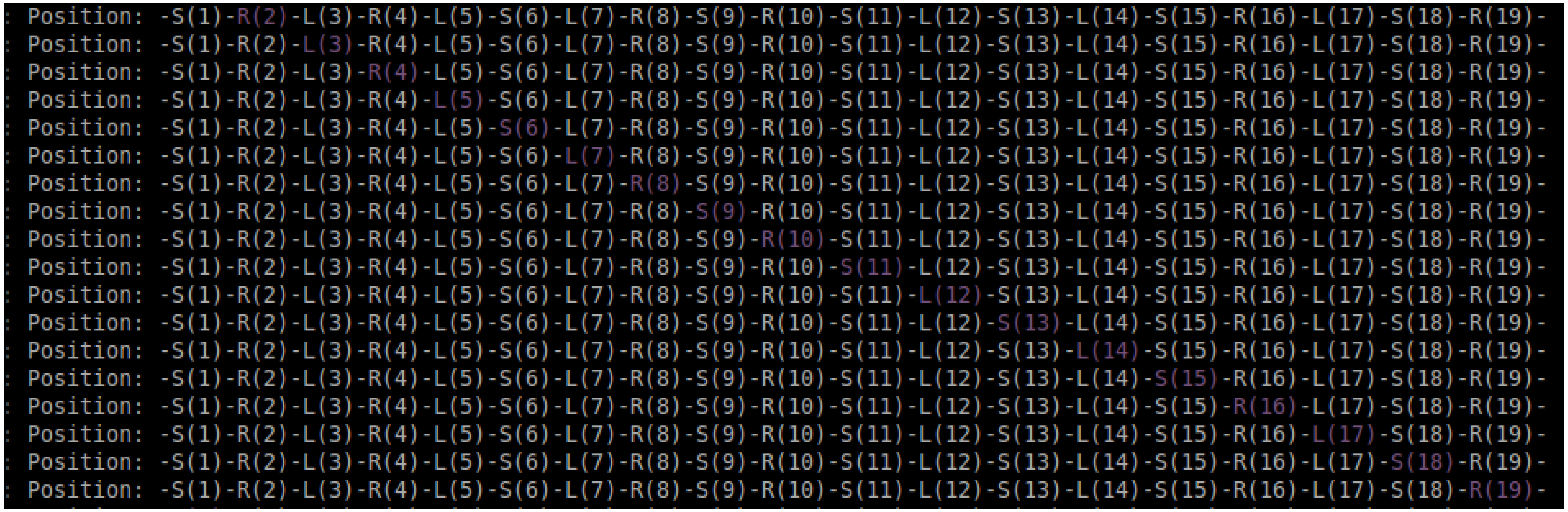
Position

- PositionDetector actor is created as soon as the track is matched
  - Keeps the track model
- Similar as the TrackRecognizer, detects direction changes on the basis of the gyro values
- Based on these direction changes, it updates the current car position on the track
- Updates the start and end times of each track part



Position

- Model extended in order to save the light barriers
- PositionDetector uses them to correct the position when lost

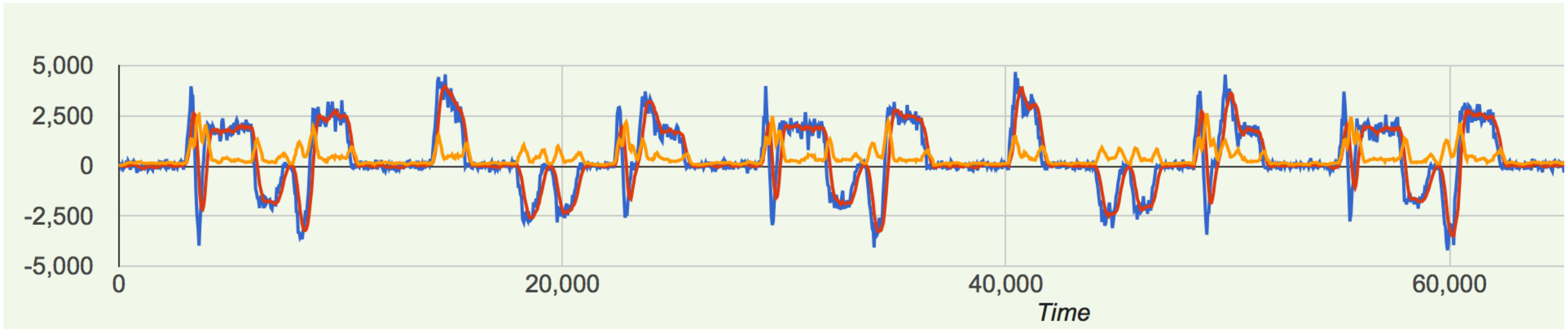


```
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
: Position: -S(1)-R(2)-L(3)-R(4)-L(5)-S(6)-L(7)-R(8)-S(9)-R(10)-S(11)-L(12)-S(13)-L(14)-S(15)-R(16)-L(17)-S(18)-R(19)-
```

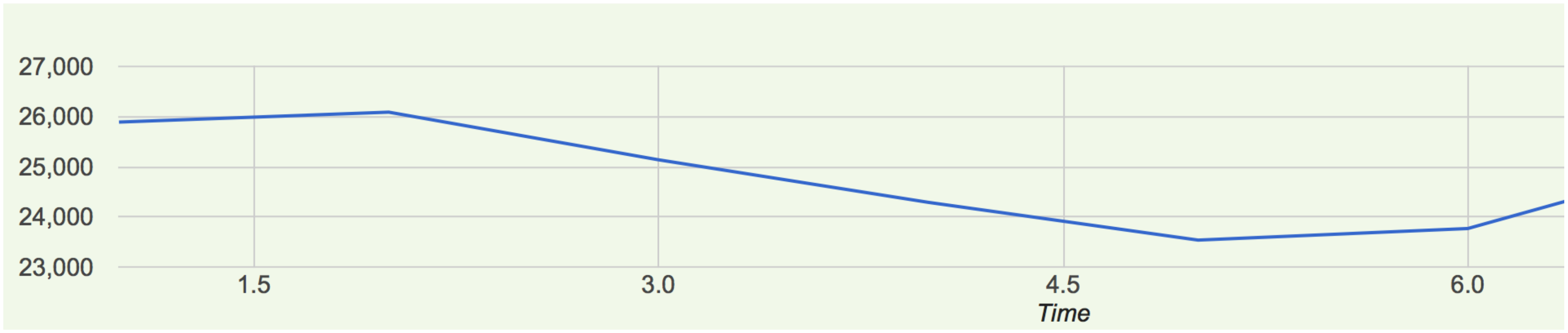
First driving

- First approach:
  - Start speeding up as soon as we are in a straight
  - Brake down before getting into the curve
  - Problem: how much should we reduce the power?
- Current Approach:
  - Speed up in straights until we get penalties
  - In the curves we keep the power from the previous straight
  - As soon as we receive a penalty, we map the penalty to the track part and we save the power value
  - Next round, we drive with less power in this track part

Round Time Comparison Graph

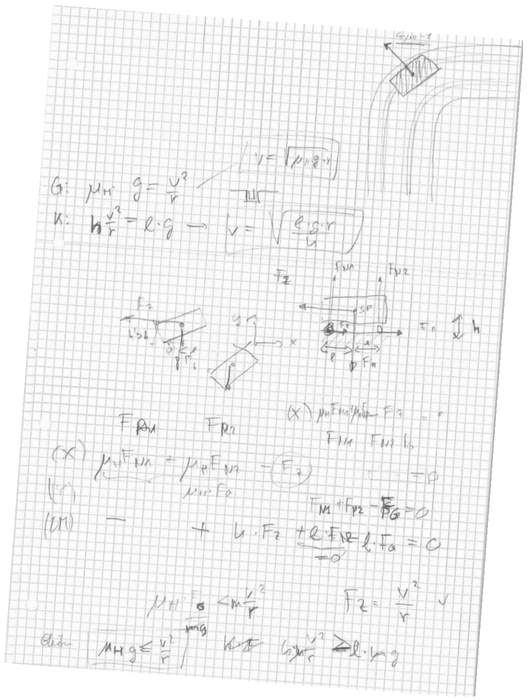


d Times



Problems

- Lost position while speeding up
  - Change thresholds during the race
  - Improved logic with light barriers for correction
- Speedup in small straight parts without barriers
  - No penalties
- Curves
  - How much power is maximum
  - No working approach till now



$v = \sqrt{\mu_H \cdot g \cdot r}$
G: $\mu_H \cdot g = \frac{v^2}{r}$
K: $h \frac{v^2}{r} = l \cdot g \rightarrow v = \sqrt{\frac{l \cdot g \cdot r}{h}}$
F_Z
F_{N1}
F_{N2}
SP
F_n
h
$h' > h$
y
x
l
F_G
F_{R1}
F_{R2}
(x) $\mu_H F_{N1} + \mu_H F_{N2} - F_Z = 0$
$\mu_H \cdot F_G$
$F_{N1} + F_{N2} - F_G = 0$
(M) $h \cdot F_Z + l \cdot F_{N2} - l \cdot F_G = 0$
$\mu_H \cdot F_G < m \frac{v^2}{r}$
$F_Z = \frac{v^2}{r}$
$\mu_H g \leq \frac{v^2}{r}$
$h \frac{v^2}{r} \geq l \cdot g$

What did we learn today 6.Nov.15?

- We cannot use the round time message for pattern matching
- We don‘t consider latency
- Use as less constants as possible
- Our logic was basically not so bad at all...

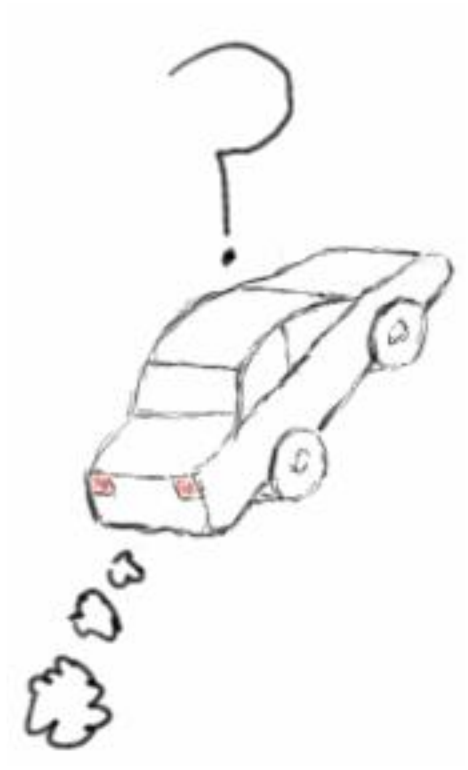
Next steps

- Reconsider pattern matching algorithm, without round time message
- Consider latency
- Configure algorithm constants dynamically
- Curve logic
- Improve speeding up in straights
- Improve error tolerance

Sources

- Prof. Augenstein (Math)
- Wikipedia (Moving Averages)
- AKKA (akka.io)
- Prof. Sourlier (Physics)

Questions?



Thanks for your attention.