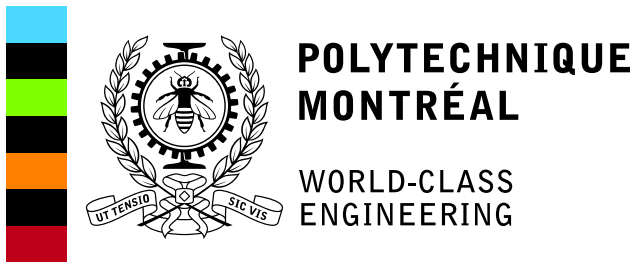
# Open source tools for trajectory data analysis

## ITS Canada 15$^{th}$ Annual Conference

Nicolas Saunier
nicolas.saunier@polymtl.ca




June 13$^{th}$ 2012

# Outline

Transportation Data

Sample Applications

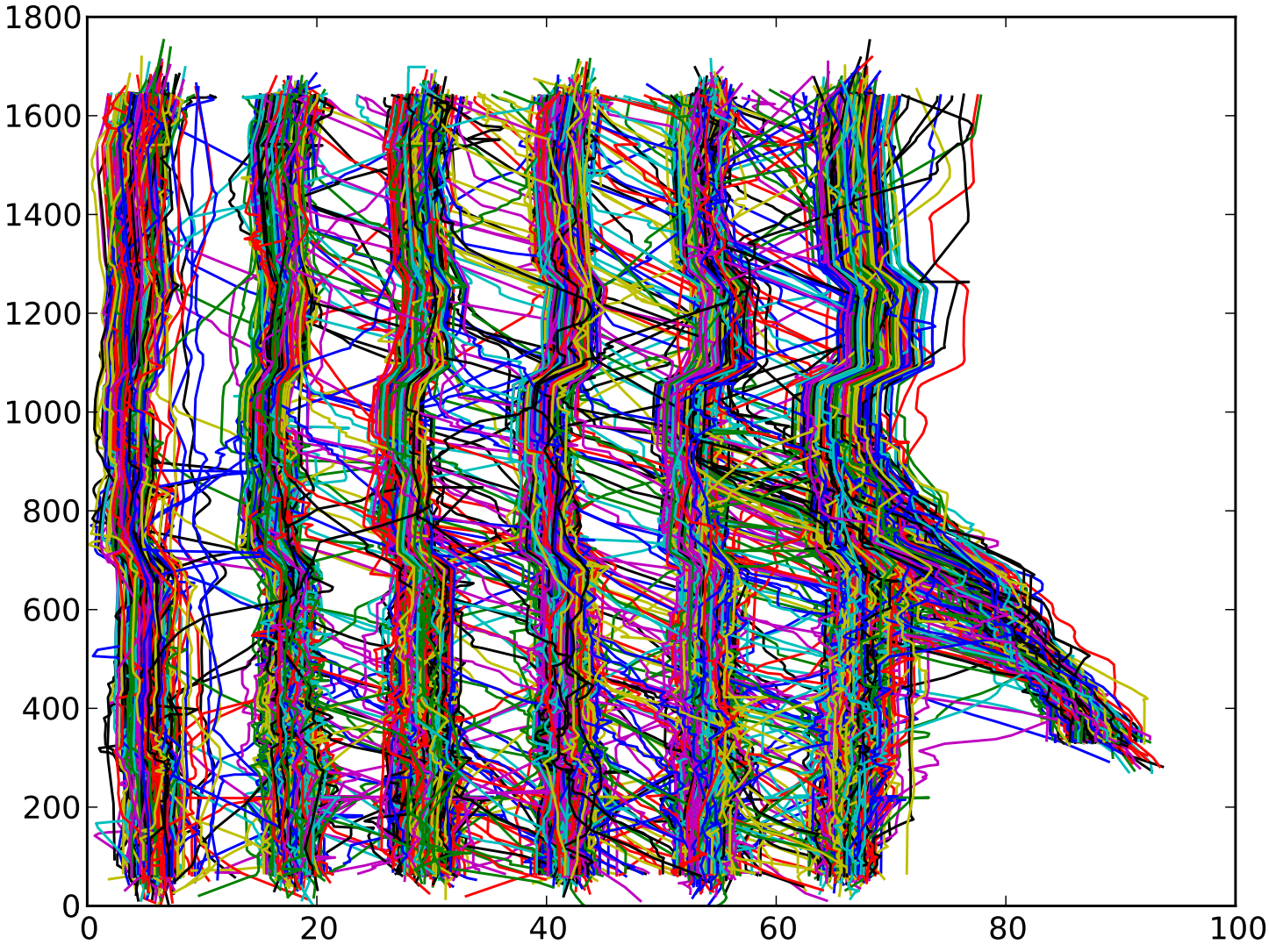
Open Source Software

Conclusion

# What is transportation?

- People and goods at different places at different times
- Sets of locations at given times: trajectories are one of the most important transportation data types
- More and more easily available, at different spatial and temporal scales
  - GPS data, vehicle probes, Automatic Vehicle Identification sensors (Bluetooth, Automated License Plate Readers)
  - Video-based tracking
- Need to compare trajectories for high-level analysis, e.g. mobility patterns

# Example: 2052 Trajectories (15 min)



# Trajectory Data

- Processing trajectories raises the following issues:
  - different sampling rates/speeds
  - outliers and noise
  - different lengths: trajectories cannot be processed in fixed-size tables (e.g. spreadsheets), re-sampling loses information, actual positions
  - efficiency: tradeoff between accuracy and computing cost
- More suitable techniques exist
  - use the right data structure:
    $[(t_1, x(t_1), y(t_1)), ..., (t_n, x(t_n), y(t_n))]$
  - use suitable similarity and distance measures, e.g. the longest common subsequence similarity (LCSS), that may leave some elements unmatched

# The LCSS

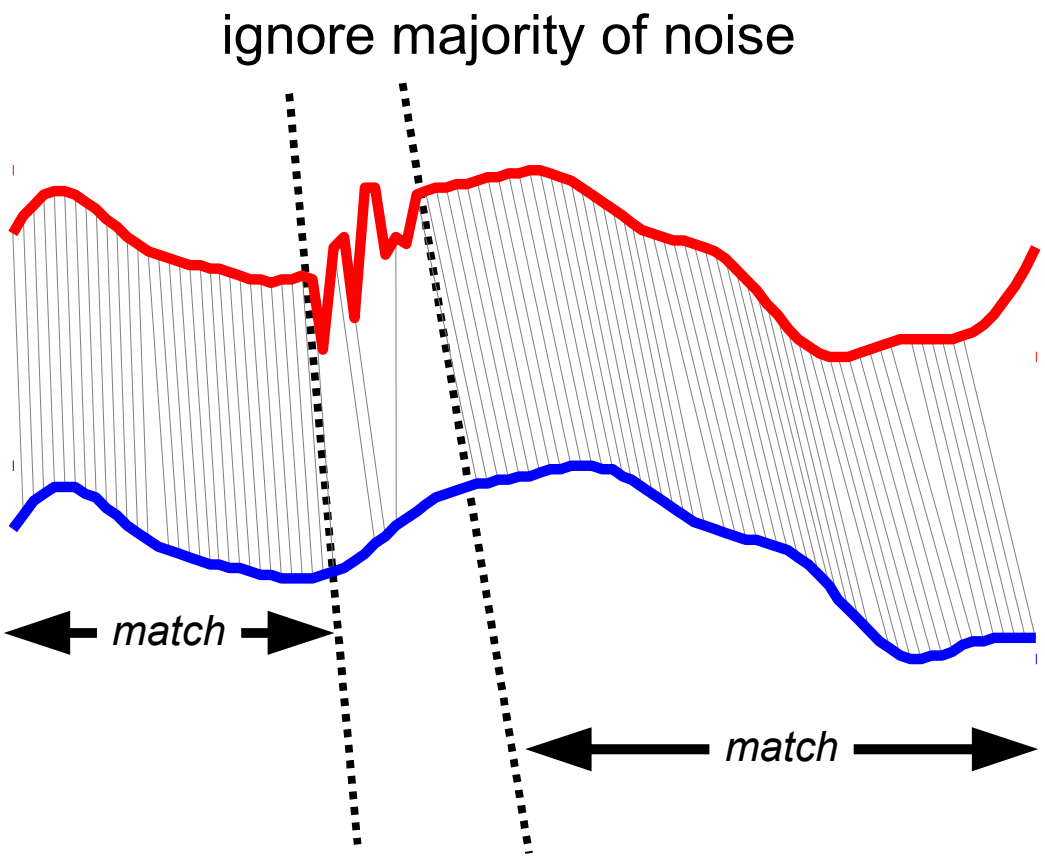
- The LCSS is a modified edit distance (used for spellchecking, handwriting recognition, DNA sequence matching, etc.)
- The LCSS is robust to noise
  - sequences are matched by allowing them to stretch, without rearranging the sequence of the elements, but allowing some elements to be unmatched
- The LCSS is very flexible
  - similarity is subjective and depends on the application

[Vlachos et al., 2005]

# The LCSS

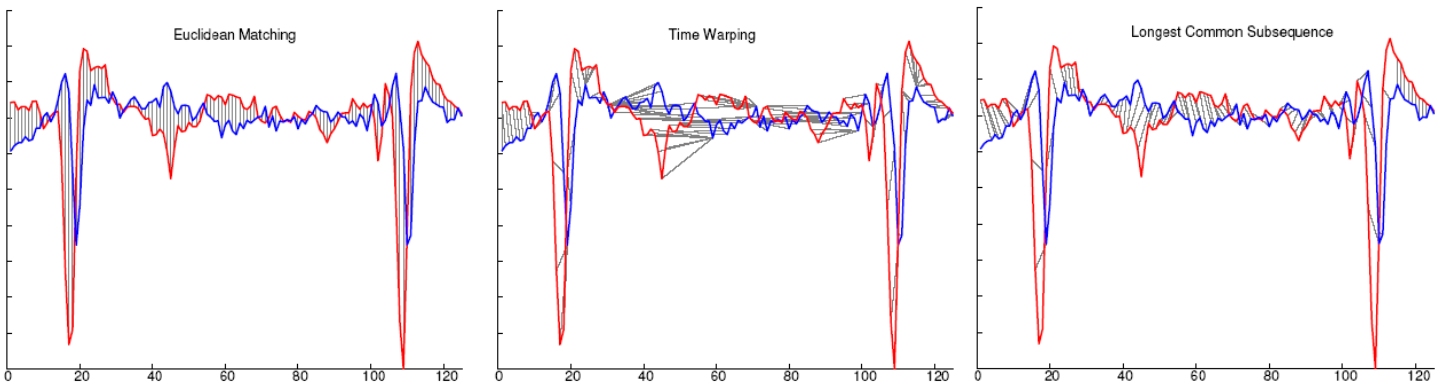
- Trajectory at regular time-steps: $P_i = \{p_{i,1}, ...p_{i,n}\}$ where $p_{i,k} = (x_{i,k}, y_{i,k})$
- $Head(P_i) = \{p_{i,1}, ...p_{i,n-1}\}$
- With a threshold $\epsilon > 0$, $P_i$ and $P_j$ two trajectories of lengths $m$ and $n$, $LCSS_\epsilon(P_i, P_j)$ is defined as
  - 0 if $m = 0$ or $n = 0$
  - $1 + LCSS_\epsilon(Head(P_i), Head(P_j))$ if the points $p_{i,n}$ and $p_{j,m}$ match
  - $max(LCSS_\epsilon(Head(P_i), P_j), LCSS_\epsilon(P_i, Head(P_j)))$ otherwise
- Example matching: $p_{i,k_1}$ and $p_{j,k_2}$ match if $|x_{i,k_1} - x_{j,k_2}| < \epsilon$ and $|y_{i,k_1} - y_{j,k_2}| < \epsilon$
- Metric $DLCSS_\epsilon(P_i, P_j) = 1 - \left(\frac{LCSS_\epsilon(P_i,P_j}{min(n,m)}\right)$

# The LCSS



# The LCSS

| Method | Complexity | Elastic Matching | One-to-one Matching | Noise Robustness |
|---|---|---|---|---|
| *Euclidean* | *O(n)* | × | ✓ | × |
| *DTW* | *O(n*δ)* | ✓ | × | × |
| *LCSS* | *O(n*δ)* | ✓ | ✓ | ✓ |

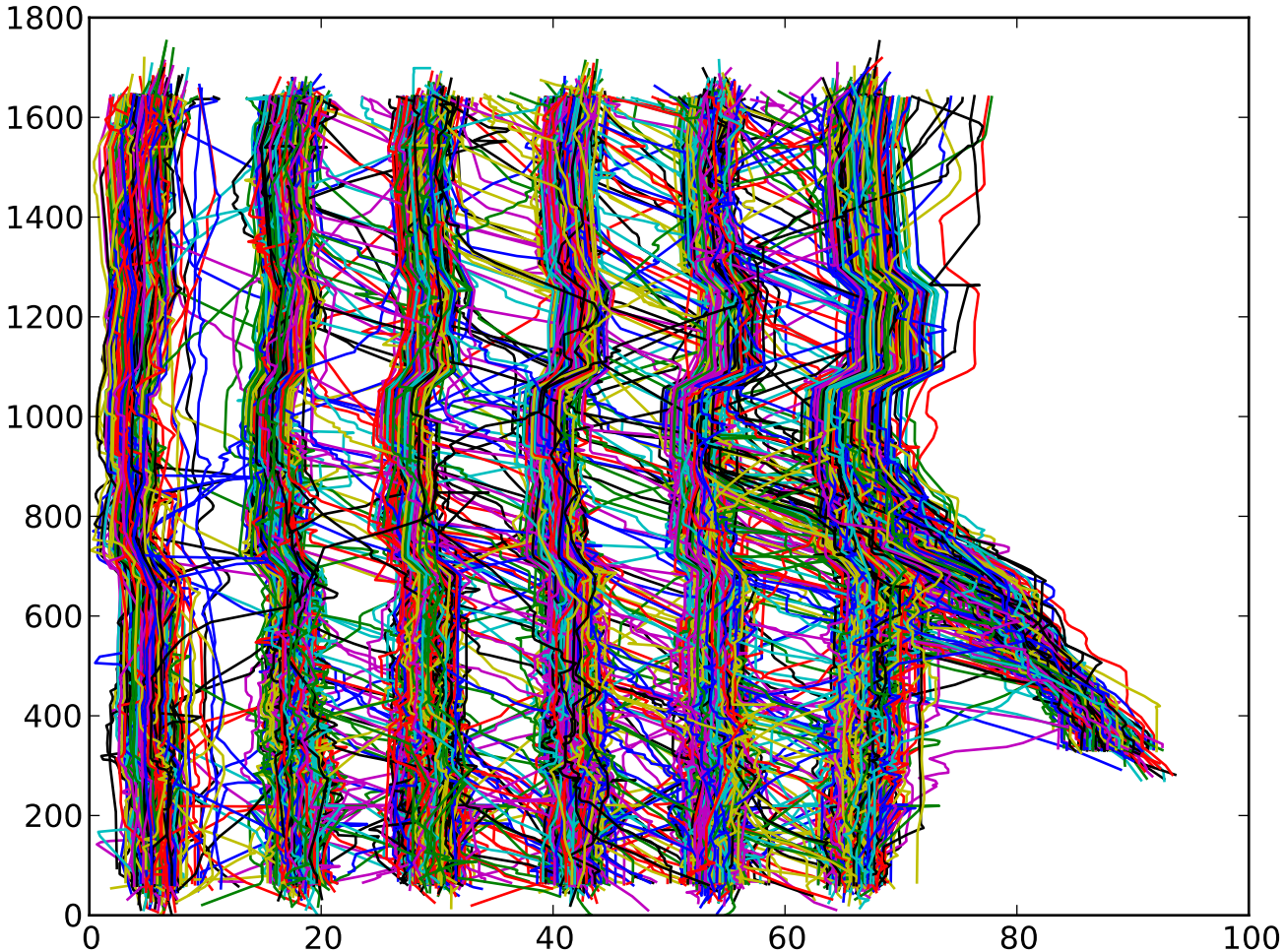



(Vlachos 2005 Tutorial)

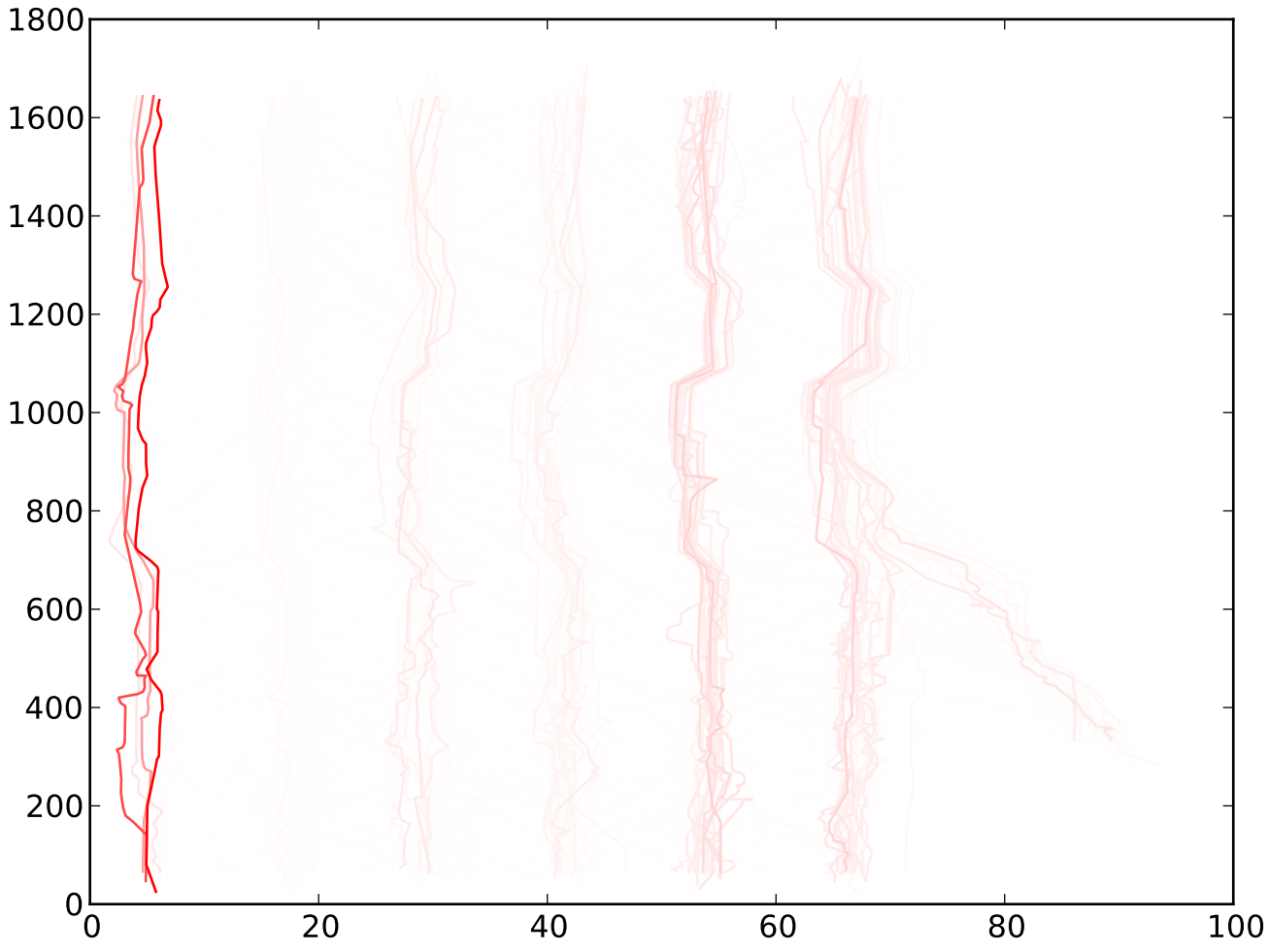
# Sample Applications

- Need to compare trajectories for
  - activity / travel behaviour monitoring and modelling
    - detect “abnormal” behaviour, e.g. infractions
  - road safety diagnosis
- Algorithms
  - clustering, e.g. k-means algorithm [Saunier et al., 2007]
  - (dis-)similarity query
- Ongoing work
  - trajectory management and analysis library
  - video-based road user tracking tool

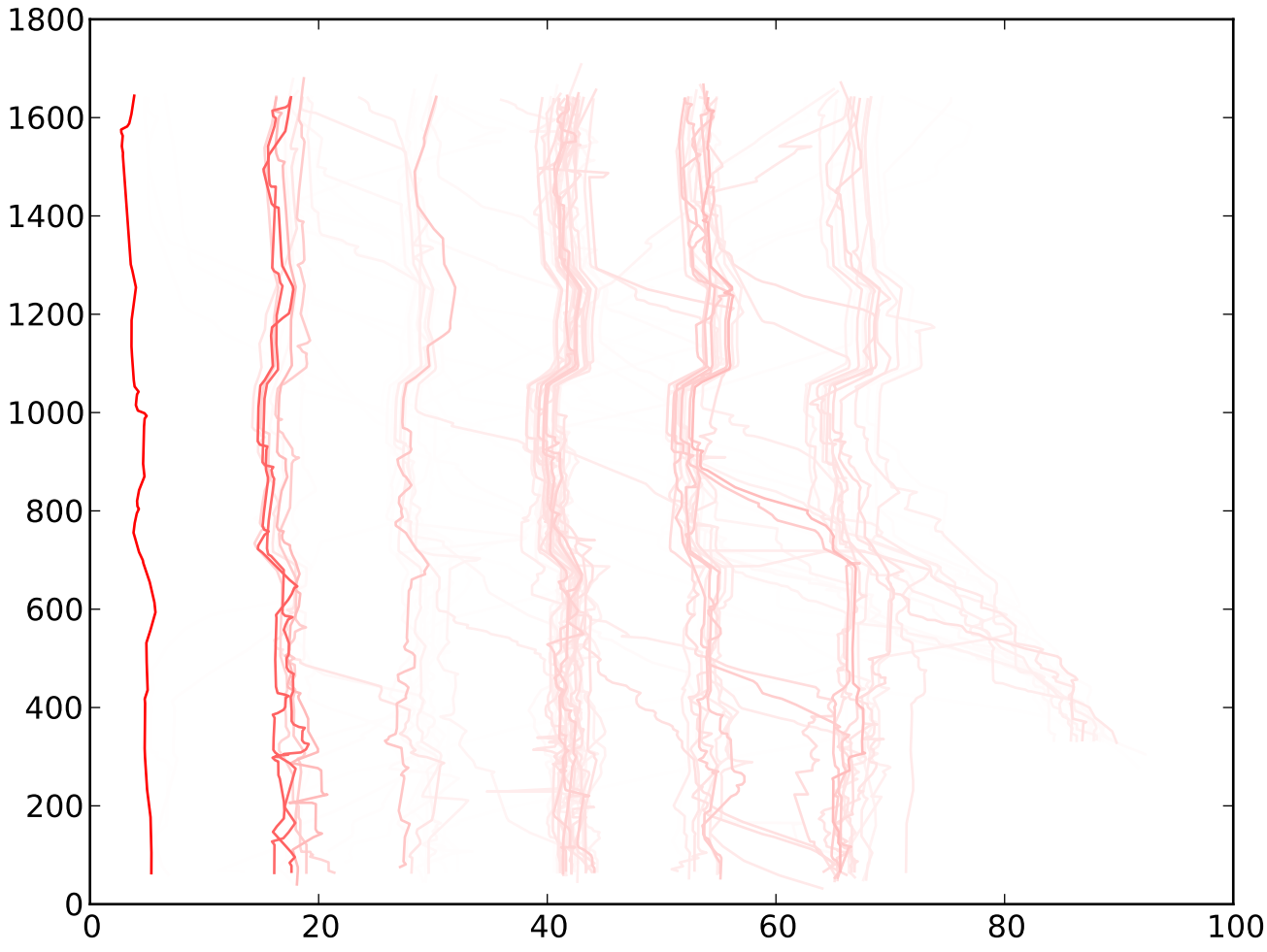
# Clustering Examples: NGSIM Dataset (2052)



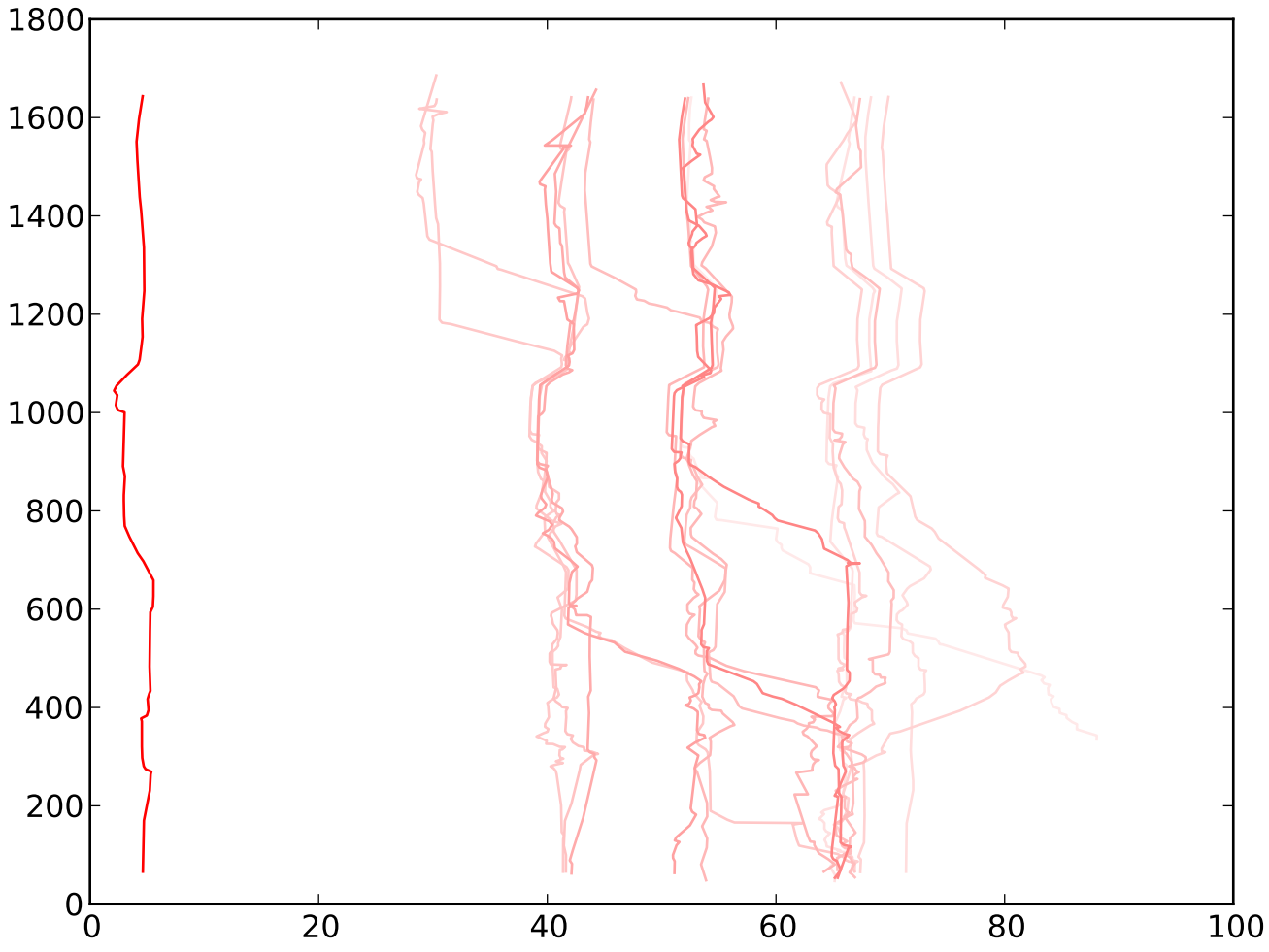
# Clustering Examples: NGSIM Dataset (333)



# Clustering Examples: NGSIM Dataset (96)

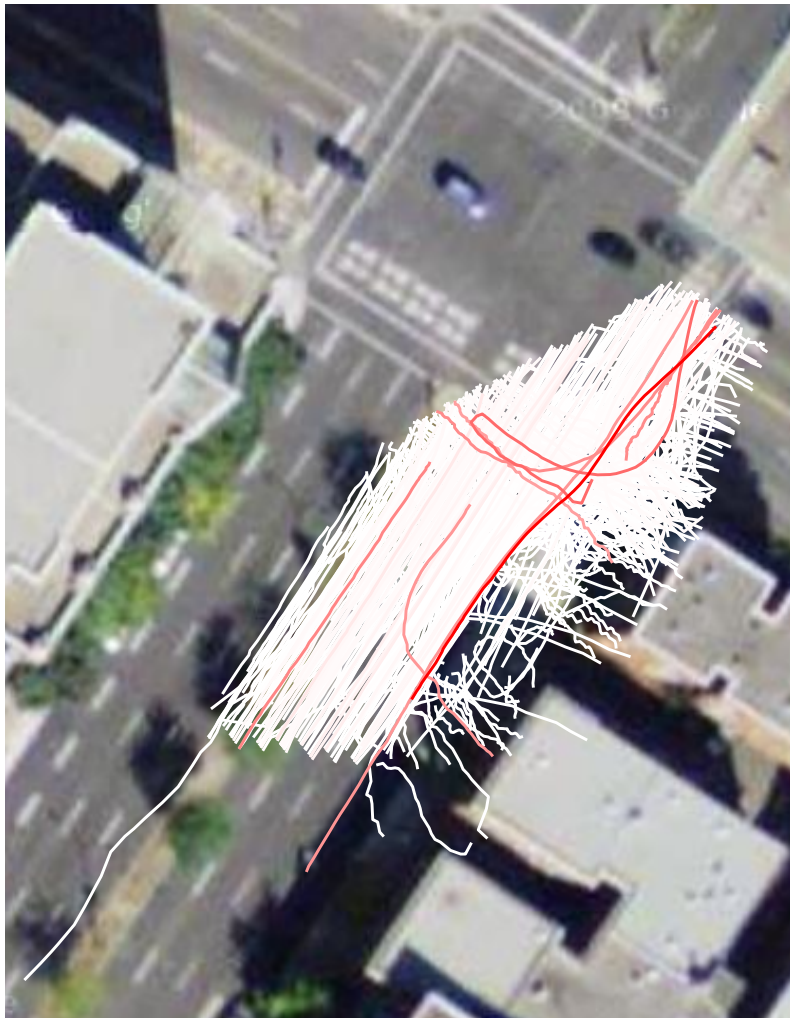


# Clustering Examples: NGSIM Dataset (19)

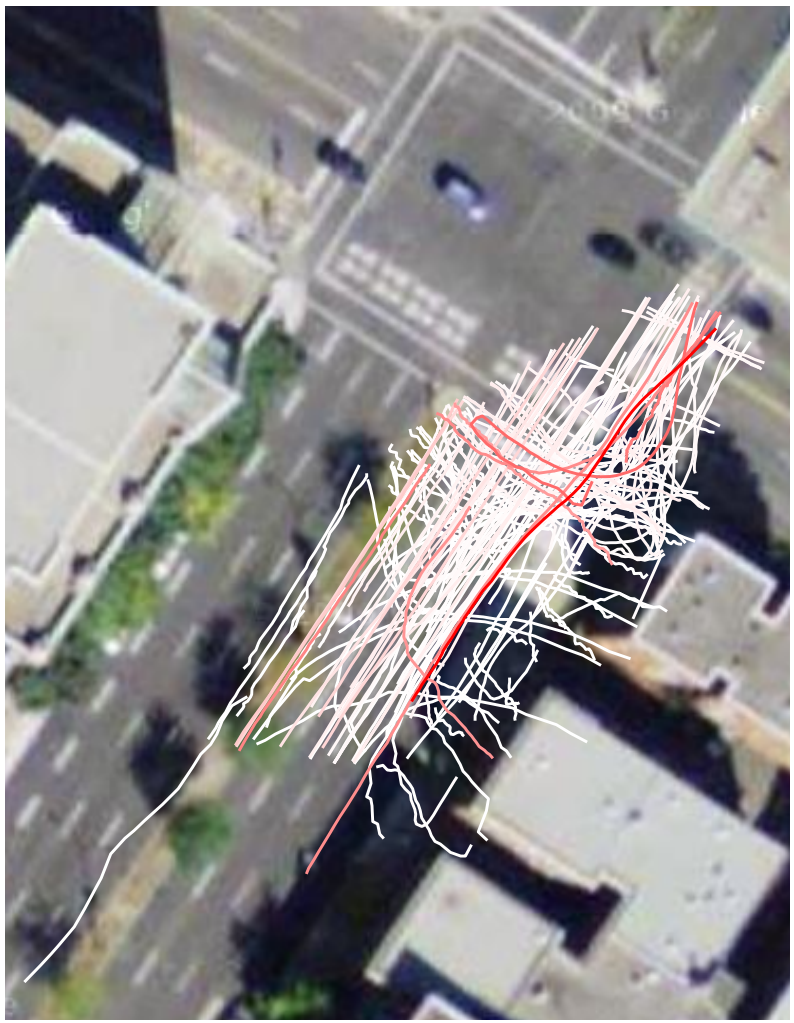


# Clustering Examples: Montréal Intersection (6777)

# Clustering Examples: Montréal Intersection (587)

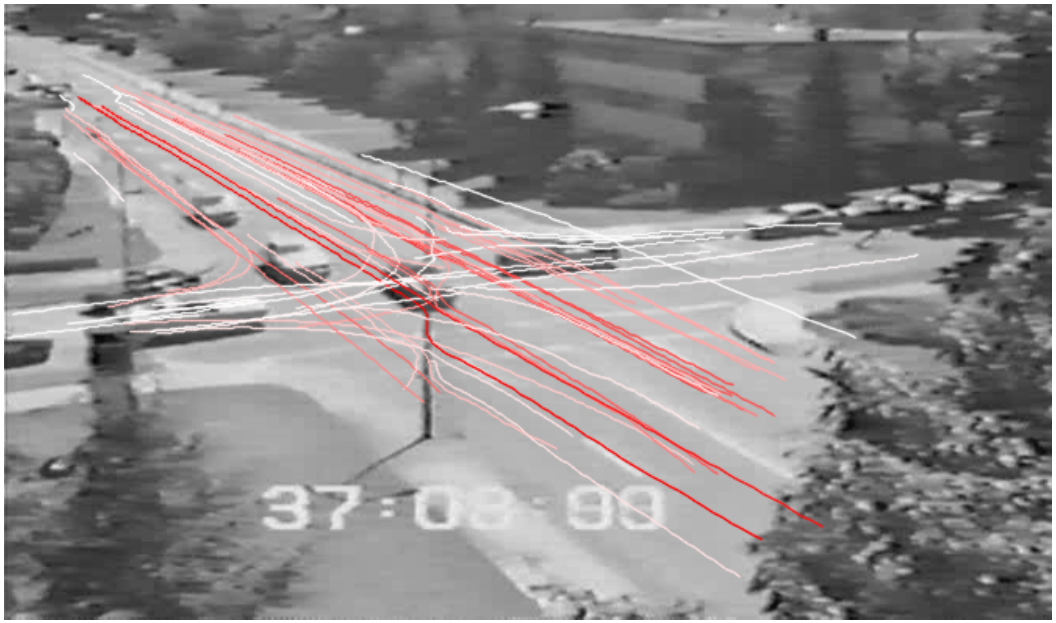


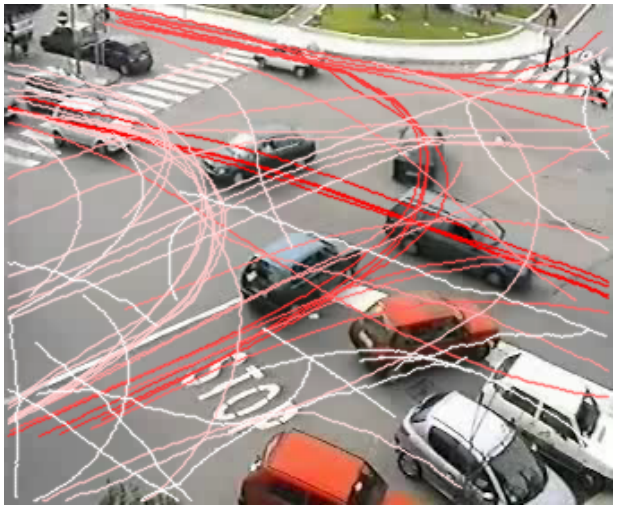
# Clustering Examples: Montréal Intersection (168)



# Clustering Examples: Montréal Intersection (9)

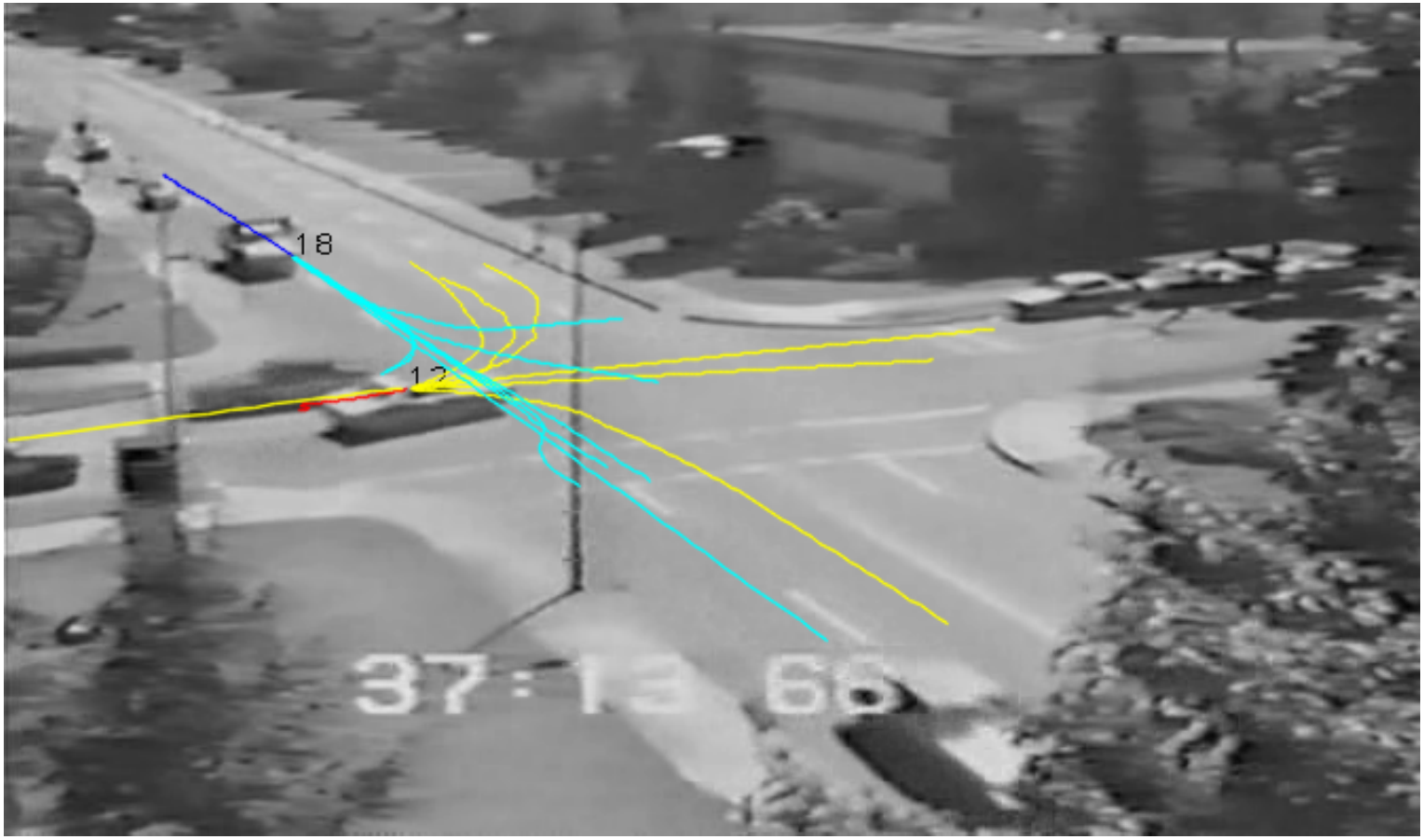
# Motion Pattern Learning





| Traffic Conflict Dataset, Vancouver | Reggio Calabria, Italy |
|---|---|
| 58 prototype trajectories (2941 trajectories) | 58 prototype trajectories (138009 trajectoires) |

# Application to Road Safety Diagnosis




Conflict data (Vancouver)

# Open Source Software (OSS)

- OSS defining characteristics (Open Souce Initiative)
  - Free redistribution
  - Source code
  - Derived work
- OSS is everywhere and you are using it daily
  - Google, Linux web servers, Android, Facebook...
- OSS often generates strong reactions: this is not about giving away software for free, or being anti-profit, but about a superior software engineering method
  - for example, The Apache foundation is suported by Microsoft, Facebook, Yahoo!, Google, IBM, HP, AMD, etc.

# Benefits of Open Source Software

1. Reproducibility of scientific results and fair comparison of algorithms
2. Uncovering problems
3. Building on existing resources (rather than re-implementing them)
4. Guaranteed access to software and tools
5. Combination of advances
6. Faster adoption of methods in different disciplines and in industry
7. Collaborative emergence of standards

[Sonnenburg et al., 2007]

# Benefits of Open Source Software

- OSS should be an obvious choice for academia (being publicly funded) and considered by industry
- Buyers should be very careful about standards and continued access to technology, and open source is an important part of the solution
- There are successful mixed business models with open source core libraries and paid graphical interfaces, technical support, consulting services, etc.

# Ongoing development

- Trajectory management and analysis library
  `https://bitbucket.org/trajectories/trajectorymanagementandanalysis`
- Video-based road user tracking tool
  `https://bitbucket.org/Nicolas/trafficintelligence`

Under BSD/MIT License

# Conclusion

- Trajectory data is everywhere and we need the right tools to process it
- Open source software is a necessary part of Open Science, i.e. doing better science
- Open source software is an attractive software engineering method for more and more companies
- Development in progress at École Polytechnique de Montréal
  - opportunities for partners
- Perspectives
  - test more clustering algorithms and metrics
  - applications to pedestrian crossing infractions, GPS data

- Acknowledgments: Piotr Bilinski (Université Sophia-Antipolis), François Bélisle (Polytechnique)
- Funding: Google Summer of Code 2010, NSERC

- Acknowledgments: Piotr Bilinski (Université Sophia-Antipolis), François Bélisle (Polytechnique)
- Funding: Google Summer of Code 2010, NSERC

Questions?

`http://nicolas.saunier.confins.net`

# References


Saunier, N., Sayed, T., and Lim, C. (2007).
Probabilistic Collision Prediction for Vision-Based Automated Road Safety Analysis.
In *The 10th International IEEE Conference on Intelligent Transportation Systems*, pages 872–878, Seattle. IEEE.

Sonnenburg, S., Braun, M. L., Ong, C. S., Bengio, S., Bottou, L., Holmes, G., LeCun, Y., Müller, K.-R., Pereira, F., Rasmussen, C. E., Rätsch, G., Schölkopf, B., Smola, A., Vincent, P., Weston, J., and Williamson, R. (2007).
The need for open source software in machine learning.
*Journal on Machine Learning Research*, 8:2443–2466.

Vlachos, M., Kollios, G., and Gunopulos, D. (2005).
Elastic translation invariant matching of trajectories.
*Machine Learning*, 58(2-3):301–334.