Detection and Analysis Algorithms of punctual road events from vehicular data in the search for the optimal route

An Overview of the AMNAM platform

Jihene REZGUI, Pascal DALLY-BÉLANGER and Philippe RIVEST
Laboratoire Recherche Informatique Maisonneuve (LRIMa)
Montreal, Canada
jrezgui@cmaisonneuve.qc.ca

***Abstract*—Beaconing in vehicular networks (VANETs) is a greatly developed research field able to support safety applications like warning the drivers of potential dangers. Nevertheless, service messages, that uses different channels than beaconing, are also a core part to improve worldwide driving conditions and to minimizing traffic. This paper provides data analysis tools and algorithms for the detection and the analysis of road events. In particular, we develop a Java platform called "AMNAM" [1]. Its purpose is transposing and obfuscating raw simulation data into realistic information to, ultimately, detect the real world position of these events e.g. potholes, roadblocks, etc. Additionally, AMNAM computes the optimal route according to the detected events. Through extensive simulations, we demonstrate the efficiency of our proposed algorithms for the detection of punctual events. We also propose a new heuristic called "DirectionnalWeight" (DW) which adds a supplementary weight to the edges guiding the search towards the destination. Moreover, AMNAM's modified A* algorithm enforced by DW outperforms classic pathfinding algorithms found in the literature in terms of required iterations for its completion.**

Keywords—VANET; Global Perception; Shortest Path; Vehicular Data; DSRC; Virage Simulation; OPTICS; AMNAM.

I. INTRODUCTION

VANETs bring a whole new spectrum of functionalities to the traditional vehicle. The latter can be used to save energy, time, and to lower the victims of car crashes by preventing accidents. In VANETs, vehicles can communicate with each other (V2V) or to equipment part of the infrastructure (V2I). These networks use the DSRC/802.11p protocol: a wireless communication system using the 5.85-5.925 GHz spectrum. DSRC provides seven 10 MHz channels, all of which have a separate function (cf., figure 1).

As shown in the next figure, the orange channels are the Service Channels (SCH) and are split in two parts: the medium range SCHs and the short-range SCHs. These channels are designed for extended data transfer for V2I and V2V communications. SCHs allows connected vehicles to easily fetch and send information on their surroundings to others without affecting periodic and alert message transmission.

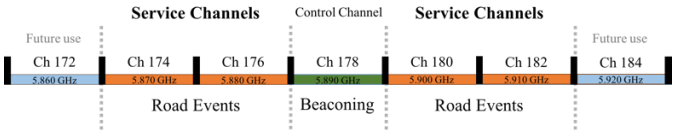

Fig. 1. DSRC's Service and Channels along with an example application

With the SCHs, VANETs can be used to reduce traffic by gathering collective data. In this paper, we use the notion of road events (or simply "events") defined as something happening on the road. The average daily travel time is greatly affected by the increasing number of vehicles on the roads of the globe. Rush hours significantly magnify the congestion up to 75% as in Los Angeles [2]. In Montreal, Canada, where the congestion levels peaks at 50% [3], such slowdowns cost approximately 1.4 Billion Canadian Dollars in 2010 [4]. The computation of a global perception of the road network fuelled by road events sent by DSRC-enabled vehicles could greatly improve the world's traffic flow.

Our contributions, in this paper, can be summarized as follows: (1) We introduce our Java platform, called "AMNAM," containing multiple data analysis tools for road events; (2) We modify the usual behaviour of "Virage Simulation"'s (VS) simulators (VSIM) [5] to obtain raw data from our vehicular simulations to, then, use it as input for AMNAM; (3) We propose algorithms to transpose and obfuscate raw simulation data into realistic information; (4) We adapt the "Ordering Points To Identify the Clustering Structure" (OPTICS) [6] algorithm to regroup the realistic vehicular data of the previous step into clusters for the detection of real-world events; (5) We demonstrated that the computation of the shortest route influenced by the detected events using the A* algorithm [7] coupled with our "DirectionnalWeight" heuristic (DW) is more efficient than Dijkstra's algorithm [8].

The remainder of the paper is organized as follows. Section II, provides a brief overview of related work and compare them with our platform. Next, the section III displays "Virage Simulation" our road simulator. Then, we present the AMNAM platform in section IV. Finally, in section V, we

show AMNAM's simulation results and conclude this paper in section VI.

II. Comparison With Other Tools

A. Google Maps

Google Maps ("GM") is a free online mapping platform allowing the user to obtain the route from the users' position to a given destination. For this paper, we will compare our platform to the GM's smartphone application. One main feature of the latter is the computation of the optimal route using its available traffic data. As AMNAM, GM's data is crowdsourced meaning that acquiring information is done using the users' devices. What distinguishes our approach to Google is that the data is natively handled by DSRC-enabled vehicles; our platform automatically sends relevant data without any smartphone. Also, road events are sent via DSRC which improves speed, reliability, and security over 3G/LTE connectivity. In addition, our platform can handle a wider range of events detected by vehicles like potholes, a tree blocking a lane, etc.

B. Waze

This smartphone application focuses more on road events via crowdsourcing. Firstly, Waze asks the user for a destination then displays the path along with diverse facts about the upcoming road. Traffic, roadwork, potential dangers and more are notified to the user as he progresses along the itinerary. Data used to detect events like these are entered manually on the application by the users. For example, a driver will notify others with Waze if he sees a car parked on the side of the highway. Our platform is fully automated and doesn't require any human interaction for the collection of road data which improves its precision and veracity. As a matter of fact, the proportion of vehicles or computers sending false information is significantly lower than malicious humans. The road safety is improved with our platform since the driver doesn't have to signal any event. Also, with DSRC, communication delays can be as low as 100 ms [9] which is notably lower than Waze who relies on human reactions.

III. "Virage" Vehicular Data Simulation

Our platform relies on data generated by VSIMs. VS has a partnership with LRIMa in order to supply the information required to test and evaluate the effectiveness of our platform and our algorithms. We modified the usual behaviour of VSIMs to obtain files containing raw data from our simulations. We added a periodical vehicular data log system to export each instant.

Our simulation scenario is based on an urban road network containing about a hundred roads forming a small city. This fictional city is surrounded by a highway. The proportion of vehicles having DSRC capabilities is configurable; they will drive along in an autonomous manner perfectly obeying road codes and laws.

VSIMs allow an operator to drive an "interactive" simulation car to explore and investigate the behaviour of the autonomous virtual vehicles.

Fig. 2. Virage Simulator in action

For our purposes, obstacles can be placed anywhere on the map of a scenario. We define obstacles as objects whose sole purpose is to disturb driving conditions. Three different categories of obstacles were implemented in VSIMs: potholes, blockages and variators. While potholes are solely for detection reasons and don't impact the vehicles in anyway, blockages, as their name implies, completely or partially obstructs the road they're placed on. Variators reduces the maximum speed limit of all vehicles in a sphere of influence with a given radius. All these obstacles have a configurable detection radius which allows DSRC-enabled to identify them within a given area. Unfortunately, since obstacles aren't truly natively supported by VSIMs' core infrastructure, some functionalities cannot be observed. The blockages are mostly avoided by the autonomous vehicles and the variators can't effect of the latter because their behaviour is final.

When the user presses on the horn of the interactive car, the simulator will log various information at regular intervals; the amount of time VSIMs save data per second is called the poll rate and is measured in Hertz (Hz). The latter can range from 0.2 Hz (one poll per 5 seconds) to 10 Hz (10 polls per second). The presence of a vehicle in the detection radius of an obstacle causes an event to be logged at each poll. Additionally, for every poll—regardless of the presence of an event—the vehicle's raw data like its speed, position, current road, etc. are logged in a separate file.

The data specified above is given as input for the AMNAM platform described in the following section.

IV. AMNAM ("AMNAM N'EST AUCUNEMENT MAPS")

A. Platform's Input

After a recording session, VSIMs output 32 "Comma Separated Values" files (.CSV) containing the raw data of the simulation. There are three types of CSVs in those files:

1) Initialization File: The simulation initialization file logs information at the beginning of a recording. It contains the number of vehicles, their ID, their brand and model, and specifies if they are capable of DSRC communication.

2) Event Detection Log File: Each time a DSRC-enabled vehicle is inside the detection radius of an obstacle, an entry is added to the event detection log file. These entries are structured the following scheme:

TABLE I. EVENT FILE STRUCTURE

#	Info	#	Info
0	Detector Vehicle ID	4	Position Y (m)
1	Detected Obstacle Type	5	Position Z (m)
2	Simulation timestamp (s)	6	Road ID
3	Position X (m)	--	—

3) Periodic Files: Depending on the configured poll rate, VSIMs will record specific information about the vehicles and log in what we call a periodic file. Each vehicle has its own file. The interactive car is also saved in a periodic file. Its structure is described in the table below:

TABLE II. PERIODIC FILE STRUCTURE

#	Info	#	Info
0	Vehicle ID	10 to 13	Acceleration X/Y/Z/Norm $(\frac{m}{s^2})$ [c]
1	Simulation timestamp (s)	14	Friction coefficient
2 to 4	Position X/Y/Z (m) [a]	15	Current Road ID
5 to 9	Speed X/Y/Z/Norm (m) [b]	16	Speed limit (Km/h)
9	Orientation ($[-\pi, \pi]$)	--	—

[a] The position is measured in reference to the world.

[b] The speed is relative to the vehicle: the Y coordinate vector is the speed forward or backwards; the X coordinate vector is the speed on the side present in drifting; the Z coordinate vector is vertical movements like ramps or speed bumps.

[c] Acceleration has the same frame of reference as speed.

4) AMNAM Files: All the data harvested from VSIMs are stored in an AMNAM file. Based on the zip architecture, the latter allows us a better manipulation of data files along with the ability to save the output of our different analysis.

B. Differences between a real world implementation and simulator data

1) Real World Implementation: The methodology of our platform, in the physical world, is described by the following steps:

i. The detection of an event by a vehicle;
ii. The transmission of the latter to the central infrastructure via DSRC's service channels;
iii. The analysis of the pool of events resulting in the creation of a global perception of the road.
iv. The broadcast of the new data contained in the calculated perception to the vehicles.

The first step, the detection of an event from a vehicle's point of view, differs from the global detection scheme proposed in this paper. We suppose that vehicles are able to reliably (but not necessarily perfectly, see section 3.C) identify events within their vicinity. The detection at a vehicular level (which implies research in sensors, computer vision, AI, etc.) is beyond the scope of this project. Once an event is detected, it is sent to the platform for analysis using the DSRC protocol. We once again suppose that the data transfer is sufficiently reliable, but it is not required to be perfect. Then, this event is put together with all others and analyzes them, attempting to extract useful information for the vehicles on the road network. The goal is to send events to the infrastructure to inform others. A real world implementation of the platform would have as sole purpose the analysis of those events reported by vehicles. However, due to the fact that we use a simulator instead of realistic vehicular data, additional steps must be added to the process.

2) Implementation in simulations: There are two main differences between a real world implementation of the platform and our platform developed for simulations. The first relates to the structure of the process. The second relates to the analysis itself.

During a simulation, additional steps must be executed in order to fit the real world model described in the previous section. As such, the transposition and obfuscation steps are required. The former allows events to be in the same format as one would expect from a car and the latter adds error to the data in order to mimic the uncertainty and errors that can happen. (see IV C. 1 and IV C. 2)

The analysis is also handled differently due to technical restrictions. As simulations cannot be run over a large period of time (15 minutes for VSIMs), the analysis is performed all at once, disregarding the fact that events are not simultaneous. In the real world, some events may come and go as time goes on, e.g. traffic and roadblocks. As the simulator does not allow dynamic creation, creation or displacement of obstacles as the simulation is running.

C. AMNAM's analysis process

1) Event Transposition: The first step in the analysis process, transposition, consists of taking the raw data generated by the simulator and formats it to obtain a better representation of what vehicles would truly send.

VSIMs collect logs for the obstacles' detection in a way that every vehicle in the detection radius of an event will raise a "reported event," regardless of the previous simulation ticks. This means that the same event will likely be reported multiple times over several consecutive ticks. For example, a pothole can be "detected" 28 times over a period of 2.3 seconds. In reality, a vehicle would only raise an event once after a detection. In summary, this step trims the data received in input. First, events are separated by sources and by type into different lists. Then, those lists are sorted chronologically. If a reported event is sufficiently close temporally and spatially to another reported event in the same list, they are

grouped together (Further details in section V). For every group, a reported event is created having as position and timestamp the mean position and timestamp of the list of events in the group. This process is repeated until no more reported events can be merged. The lists are then put back together to form the transposition's output: a single list containing all reported events with the unwanted repetition trimmed down. However, these reported events are still far from representing our world. The next step, the obfuscation, will remedy this issue.

2) Reported Event Obfuscation:

a) Event Obfuscation to Mimic Real Vehicles: The output of the previous step regroups every detection VSIMs' vehicles make into fewer but concise reported events. In real life applications, these events would've been sent to the infrastructure for further processing. Unfortunately, the vehicles in VSIMs are programmed to act perfectly in every way. It is fundamentally impossible to harvest data close to reality from VSIMs because perfection doesn't exist in real life. The perfect detection hardware of the simulation vehicles, their perfect road behaviour and their lossless vehicular communication environment are the primary elements rendering the simulation data too precise to be useful. Hopefully, we developed the obfuscation system to morph data into input. The latter introduces errors and uncertainty into the transposed information outputted by VSIMs; inaccuracies gives reported events a realistic character. To obtain these imprecisions, we manipulate several values contained in events with different mathematical models as the one presented in the following subsection.

b) Gaussian or Normal Distribution: In nature, according to the central limit theorem (CLT), the sum of independent variables converges into a normal/gaussian distribution.To simulate error factors resembling reality, we propose to use Gaussian distributions in some aspects of the obfuscation step to give a desired amount of uncertainty to our data.

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (1)$$

Where:
x: Random variable to distribute
μ: The mean of the distribution
σ: The standard deviation of the distribution

c) Developed Obfuscation Parameters: We implemented different obfuscation parameters impacting three main components of our events: position, type and packet loss. Theses parameters allows the user of AMNAM to specify the different variables used in the obfuscation process. Here's the detailed description of the three proposed parameters:

- *Position Parameter:*

The position parameter's goal is to reproduce the precision level of the average GPS. Still according to the CLT, the sum of every little error in the positioning process results in a Gaussian-like distribution [10]. For a real-life representation of the GPS data, we will obfuscate the position with random normally distributed factors. The user specifies the standard deviation (σ) in metres of the Gaussian distribution to apply on the x and y coordinates. According to the department of defence of the United States of America, the average standard deviation is 3.9 metres [11].

- *Detection Type Parameter:*

A fault in the sensors of a vehicle can lead to a misinterpretation of an obstacle. This parameter changes the obstacle type of a reported event to another randomly selected. The user enters the desired error percentage. For example, a 5% error rate will randomly change the type of an event with a 20:1 chance.

- *Packet loss Parameter:*

Since *DSRC isn't fully implemented in VSIMs, the packet loss obfuscation parameter allows us to simulate errors in the transmission of an event from the vehicle to the infrastructure. Comparatively to the type obfuscation parameter, the user enters the desired error rate for the wireless communication. The obfuscation process will simply delete events who suffered from packet loss.*

In summary, the obfuscation step adds imprecision to our data. This way, we transform our input from VSIMs to better represent real life conditions. These transposed then obfuscated events are now suitable for analysis.

3) Event Analysis

The analysis is the step during which the obfuscated events are put together in order to generate the platform's perception of the road network. The goal of this step is to undo the obfuscation in order to retrieve the original position of the obstacles. The way this is done is by clustering the data together using the OPTICS algorithm. While this algorithm is designed to uncover the structure of a cluster, we are interested only in the cluster itself and thus discard that information. This algorithm can be applied to any subset S of a normed vector space V over the rational numbers ℚ. As such, we can use this algorithm to cluster the data points of each event, by using the (x, y) coordinate as a vector and the norm defined as the standard Euclidean norm. In our implementation of the algorithm, we use the square of the Euclidean norm instead to eliminate the time taken for the square root during the computation of the norm.

$$\|\vec{x}\| = x^2 + y^2 \quad (2)$$

However, this doesn't affect the result whatsoever.

The algorithm takes three inputs: a pair of parameters and the set of data. The parameters describe the minimal requirements for a cluster to be generated. The first parameter is the minimal amount of data to generate a cluster (*minPts*) and the second one is the maximal distance between two data points to create a connexion between them (ε). OPTICS uses the concept of the neighborhood of a data point, that is, the set of all data points within distance ε from the point.

a. *Neighborhood of a vector:* The set of all points in S that are at most ε away from the referential vector.

I.e.

$$\{v \in S | \, \|v - v_0\| \leq \varepsilon\} \quad (3)$$

b. *Core distance of a vector:* The distance between this vector and the *minPts*-th furthest vector in S relative to this vector. Undefined if this distance is greater than ε

c. *Reachability distance of from a vector to another.* The distance between the two vectors or the core distance of the second vector, whichever is larger. Undefined if the neighborhood of the second vector contains less than *minPts* elements.

The first step is to flag every data point as unprocessed. For every unprocessed data point available, if their neighborhood contains less than *minPts* points the algorithm marks the point as a singleton. Otherwise, it creates a cluster C and adds every vector in S that is within distance ε of any data point in C and marks them as processed. The algorithm completes when every data point is flagged processed.

When the algorithm completes, all data points are organized in sets of sets, representing each of the clusters of reported events. If the clustering succeeds, there would be one event per obstacles.

D. Pathfinding Using Generated Perception

Pathfinding is a core part of our project. The goal is to suggest drivers the optimal route to a given destination. The computation of the shortest path is influenced by the detected events in the analysis process. A great majority of pathfinding algorithms proposed in the literature are based on graph theory.

Firstly, the road network of the VSIMs map must be converted into a graph in order to find the optimal route. The resulting graph must be directed and weighted. Its nodes are the intersections of the map while its edges are the roads linking them. Each resulting road node in the map's graph contains additional information like a unique ID and its three-dimensional coordinates (x, y, z). On the other hand, edges also contains a unique ID along with its source node, destination node and initial weight. Bi-directional regular streets are moralized by two edges in opposite direction and one-ways by a single edge.

Secondly, the initial weight has to be set for each edge of the graph. The weight has described in (4), is the ratio of two factors. The first ($\Delta\vec{r}$) is the Euclidean distance between two intersections or simply the length of the road link the latter. This road's speed limit (v_e) is the second factor.

$$g = \frac{\Delta\vec{r}}{v_e} \quad (4)$$

Their resulting ratio, the estimated time of travel, is the weight of the edge (g).

Lastly, to obtain the shortest path, we use the A* Pathfinding algorithm. Relying on graph theory, this algorithm is one of the most efficient of its kind. The addition of DW further improves the performance of the latter. DW, as stated in its name, guides A* in its node search by adding an extra value, the "h score" (h), to the weight of the edges (5).

$$f = g + h \quad (5)$$

The f score is the final weight attributed to the edges.

The advantage of DW is that fewer nodes have to be explored by the algorithm to find the shortest path resulting in faster computation times. The h score is computed as follows:

$$h(n) = \frac{\sqrt{(x' - x)^2 + (y' - y)^2}}{v_{max}} \quad (6)$$

Where:

n: The node to add the extra weight

v_{max}: Max speed allowed in the road network

x': The x coordinate of the destination node

y': The y coordinate of the destination node

x: The x coordinate of the node n

y: The y coordinate of the node n

Implemented in AMNAM, DW decreases the number of nodes explored by A* resulting in faster computation times.

The output of our modified A* algorithm is a list of nodes. By finding the edge linking each pair of nodes together, we're able to reconstruct the path. The final route gives in order the roads and intersections to go over until the driver reaches destination.

V. SIMULATION RESULTS

A. Simulation Input

To obtain the results in the next subsection, the following input parameters were specified:

Our modified scenario was executed during 900 seconds (15 minutes). The map contained in the scenario is 5255 metres high and 8106 metres wide.

Six (6) obstacles of pothole, blockage and variator type were placed on the map at the coordinates in the subsequent table. Each obstacle has a detection radius of 15 metres.

TABLE III. OBSTACLES INITIAL CONDITIONS

Obstacle ID	Obstacle Type	Position (X, Y) in metres
1	Pothole	(-228.57; 321.01)
2	Pothole	(-37.07; 554.45)
3	Blockage	(-326.07; 388.92)
4	Blockage	(283.49; -7.46)
5	Variator	(-174.72; 628.46)
6	Variator	(374.26; 456.3)

Along with the obstacles, 30 vehicles, including the interactive car, where rolling along autonomously in "Simton." We used a 70% DSRC penetration rate (DSRC capable vehicles ratio) along with a simulation poll rate of 10 Hz. The initial position of the 29 other vehicles was assigned randomly in the top left quadrant of the city.

We drove the interactive car far out of the road to avoid any "interference" with the autonomous traffic. In our simulations, the latter had the tendency of escaping the city by wandering on the highway surrounding the it. The highway didn't contain any obstacle, so cars rolling on it will not report events. A low event count is the cost to pay for the ability to access every part of the network.

During transposition, we group two reported events if they are within one second of each other and if the distance does not exceed 30 meters.

For the obfuscation process, we used a 3.9m standard deviation for the position parameter; a 7.5% error for the type parameter; and a 5% packet loss.

With this input, we obtained the results stated in subsection *B*.

B. Generated Results

1) AMNAM Interface

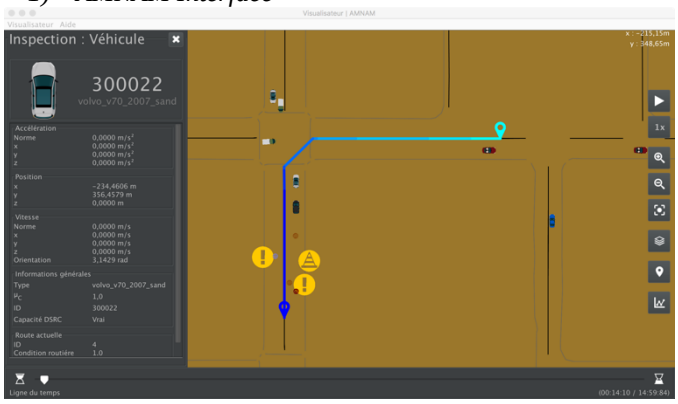

Fig. 3. AMNAM's visualization window with vehicles, a detected pothole and an example path.

2) Algorithmic Accuracy

a) OPTICS

The algorithm used for clustering the events in order to recover the original obstacles gives good result. For the following results, every data point is created by evaluating the average of a large number (at least a thousand) of repeated evaluation.

Using transposed data, and obfuscating it using different parameters, we found that the probability that clustering will be done successfully, that is, every reported event is properly associated with an event, describes a sigmoid function as shown in fig. 4, likely reminiscent of the Gaussian function used during obfuscation.

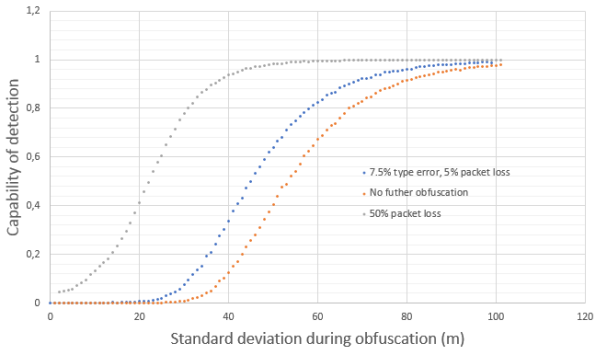

Fig. 4. Impact of the standard deviation during obfuscation on the capacity of detection

The average error can be generated similarly, as shown in fig. 5. This is calculated by looking at the events and the obstacles and trying to match each event to an obstacle. We then take the average of the distance between the two for every event.

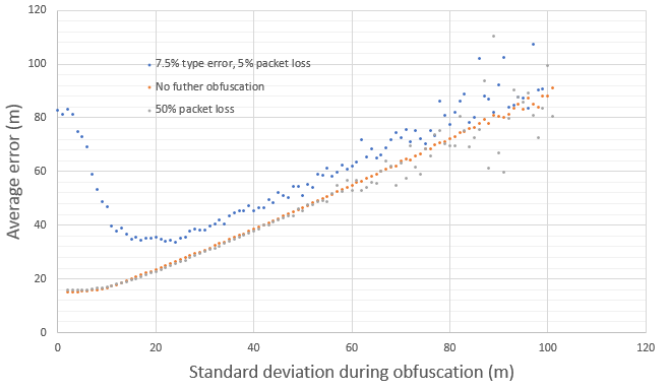

Fig. 5. Impact of the standard deviation during obfuscation on the average distance between an event and its corresponding obstacle

Fig. 5 shows us that removing events tend not to affect the precision of the analysis step, but adding type error does. This is likely due to the fact that the change of type generates can potentially create an unforeseen event which would increase the overall average. As the standard deviation during obfuscation increases, however, it becomes harder for those to form, due to the increase in distance between points in a cluster, bringing the average distance down to a level comparable to the other two types of obfuscation shown in the graphic.

Considering the small number of events generated by vehicles due to technical constraints imposed by VSIMs. A larger amount of data would facilitate clustering, which in turn would allow us to input more restrictive parameters for OPTICS. Those would then make it harder for aberrations to tamper with the perception.

b) Algorithmic Performance of A with DW*

The pathfinding system we proposed in AMNAM performs just as expected. With DW, its efficiency is indeed greater than Dijkstra's Pathfinding Algorithm. In table IV, we compare the two algorithms in terms of explored nodes. A

lower node count makes the algorithm more efficacious. To give an order of magnitude, the whole network is made of 354 nodes where 252 of them composes the inner city.

TABLE IV. A* + DW PERFORMANCE COMPARISON TABLE

	Nodes explored		
Path length (m)	**Dijkstra**	**A* + DW**	**Comments**
320.36	26	8	Route through pothole obstacle
404.79	34	18	Route clear of obstacles (CE)
611.38	45	17	Route through variator, pothole avoiding blockage
1385.71	77	69	Route through city
7053.20	164	140	Route from sector with obstacles to highway
16,293.58	153	143	Complete highway cycle— CE

VI. CONCLUSION

In conclusion, we developed the AMNAM Java platform for the detection of road events from vehicular data and for the computation of the optimal route influenced by the latter. We started by introducing DSRC and its SCHs and their potential applications for road event transmission. We compared our research to related work. After, we presented our modified VSIMs used to harvest vehicular data. Then, we showcased the AMNAM platform. Finally, we show our simulation analysis results. In this paper, we demonstrated that the detection of road events is possible with our proposed algorithms. For future work, we aim to add real time capabilities to our platform to demonstrate its capabilities in real world usages.

ACKNOWLEDGMENT

This research was financially supported by the "Fonds Québécois de la recherche sur la nature et les technologies (FRQNT)." We would like to thank Caroline Houle, teacher at Collège de Maisonneuve, for her valuable comments.

REFERENCES

[1] P. Dally-Bélanger and P. Rivest, "AMNAM", GitLab, 2017. [Online]. Available: https://gitlab.com/LRIMA/AMNAM. [Accessed: 12- Jun- 2017].

[2] "TomTom Traffic Index: Los Angeles", Tomtom.com, 2017. [Online]. Available: http://www.tomtom.com/en_gb/trafficindex/city/los-angeles. [Accessed: 08—May - 2017].

[3] "TomTom Traffic Index: Montréal", Tomtom.com, 2017. [Online]. Available: http://www.tomtom.com/en_gb/trafficindex/city/montreal. [Accessed: 08- May- 2017].

[4] J. Bourque, COMBATTRE LA CONGESTION ROUTIÈRE À MONTRÉAL PAR L'IMPLANTATION D'UN SYSTÈME DE PÉAGE AUTOMOBILE, 1st ed. Montréal: Université de Sherbrooke, 2013, p. 18.

[5] VS500M Car Driving Simulator for Training and Research—Virage Simulation", Virage Simulation Driving Simulator Systems (Car Simulator, Truck Simulator), 2017. [Online]. Available: http://viragesimulation.com/vs500m-car-simulator-training-and-research/. [Accessed: 11—Jan - 2017].

[6] M. Ankerst, M. Breunig, H. Kriegel and J. Sander, "OPTICS: Ordering Points To Identify the Clustering Structure", ACM SIGMOD Record, vol. 28, no. 2, pp. 49–60, 1999.

[7] P. Hart, N. Nilsson and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", IEEE Transactions on Systems Science and Cybernetics, vol. 4, no. 2, pp. 100–107, 1968.

[8] E. Dijkstra, "A note on two problems in connecion with graphs", Numerische Mathematik, vol. 1, no. 1, pp. 269–271, 1959.

[9] J. Rezgui & S. Cherkaoui, "Analytical Transmit Power Adjustment in Cooperative Vehicle Safety Systems", IEEE Globecom, 2012

[10] F. van Diggelen, "GNSS Accuracy: Lies, Damn Lies, and Statistics", GPS World, no. 18, 1, pp. 28–32, 2007.

[11] Departement of Defence of the United States of America, "GLOBAL POSITIONING SYSTEM STANDARD POSITIONING SERVICE PERFORMANCE STANDARD", Washington, 2008