Applying Software Agents to Make City Traffic Management Smarter

Robert Pniewski, Dominik Sellin, Katsiaryna Stankevich, Maria Ganzha, and Marcin Paprzycki

Abstract With the emergence of IoT technologies, autonomous vehicles, and introduction of 5G networking, the idea of a smart city is considerably more achievable than before. One of the ways of reaching this goal would be improvement of traffic management. In this contribution, we suggest how an agent-based system can optimize vehicular travel, thus reducing CO_2 emission, decreasing time spent in traffic jams, and improving overall city travel experience. We propose the design of an agent system describes how agents communicate and, experimentally, validates the design in two use case scenarios.

Keywords Smart city · Software agents · Traffic management · Traffic simulation

1 Introduction

As the concept of autonomous vehicles continues to develop, they bring enormous potential for improvement of daily lives. As Nico Larco suggested in his TEDx talk [1], this technology will transform our cities in multiple ways. With cars becoming smart, it is logical that, with use of IoT devices, "streets" (and city infrastructures,

R. Pniewski (✉) · D. Sellin · K. Stankevich · M. Ganzha
Department of Mathematics and Information Sciences, Warsaw University of Technology, Warsaw, Poland
e-mail: r.pniewski@student.mini.pw.edu.pl

D. Sellin
e-mail: d.sellin@student.mini.pw.edu.pl

K. Stankevich
e-mail: k.stankevich@student.mini.pw.edu.pl

M. Ganzha
e-mail: m.ganzha@mini.pw.edu.pl

M. Paprzycki
Systems Research Institute Polish Academy of Sciences, Warsaw, Poland
e-mail: marcin.paprzycki@ibspan.waw.pl

D. Goyal et al. (eds.), *Proceedings of the Second International Conference on Information Management and Machine Intelligence*, Lecture Notes in Networks and Systems 166,
https://doi.org/10.1007/978-981-15-9689-6_72

in general) can also become "smart." In this context, we focus on one aspect of potential use of these technologies—traffic management.

Let us consider two scenarios that guide our work. (1) Assume that vehicles and traffic lights can communicate. Here, vehicles may inform traffic lights that they will "arrive soon" (even, communicating the ETA). This, in turn, may allow the traffic system to decide if the lights should be changed earlier/later. It can be stipulated that having such infrastructure in place could decrease the overall travel time. Obviously, it could also lead to the reduction of CO_2 emission (by making travel "smoother"). (2) Let us now assume that citizens can use their phones to communicate with both the city transport infrastructure and the traffic lights. Hence, they could, for instance, inform public transport that they are near a bus stop[1] (possibly, providing it with the ETA). This may allow the bus to wait (for a limited time), and assure that the majority of potential passengers will board the vehicle. Obviously, traffic lights adjustment, similar to that in the first scenario, can be envisioned also for pedestrians, e.g., those approaching the specific bus stop.

Let us observe that both scenarios depend heavily on two factors: (a) message-based communication, and (b) some form of "intelligence," which would facilitate "strategic decisions." These two features are very often used in the context of software agents and agent systems (see, for instance, [2]). Moreover, agent-based approaches have materialized in the context of traffic management, i.e., in taxi services simulation [3], taxi service optimization [4], or autonomous car system simulation [5]. With this in mind, our goal is to develop a realistic agent-based traffic management system simulator and experiment with its capabilities.

2 Related Work

Let us start by summarizing related work. Here, we do not discuss "general work" related to issues such as 5G networks, Internet of things, and smart cities. While pertinent, they are not necessary to provide context to our approach.

Let us start from *agent-based simulation of autonomous cars* [5], which focuses on possible applications of autonomous vehicles. It also provides examples of interactions with public transport, which is a subset of functionality considered in our project.

Second, *Public Transportation Simulation by Using Agent-Based Simulation: Case of Tirana* [6] describes traffic situation in the said city and shows that usage of agent-based modeling can help pinpoint the causes of existing problems. In contrast, we focus on modeling and exploring possibilities for traffic management.

Finally, other contributions dealt with the use of software agents as related to specific aspects of traffic management. Here, (1) *agent-based design of intermodal freight transportation systems* [7] focused mainly on cargo transport; (2) *software*

[1] We will use the generic terms: *bus* and *bus stop* to denote any form of public transport, e.g., bus, tram, trolley, etc., and its respective stopping locations.

agents in support of a taxi corporation [4] and (3) *agent-based modeling for simulating taxi services* [3] concentrated on supplying an "on-demand taxi" service. While those projects dealt with a specific aspect of smart transport management, our work attempts at developing a complete simulation ecosystem.

3 Agent System Design

Let us start form general assumptions that underline our work. (1) We consider traffic within a single city. Specifically, we use an actual map of Warsaw, Poland. However, our system works with any city map. (2) 5G networking has been installed within the city. (3) Vehicles can use 5G infrastructure to communicate with each other (which is not used, for the time being), and with city infrastructure. (4) Citizens can use their (5G) phones to communicate with city infrastructures (as needed). (5) Software agents will be used as the conceptual and technological foundation of the developed smart traffic simulator. (6) The goal of the project is to study how to use software agents to optimize city traffic.

Based on these assumptions, and keeping in mind the two (above outlined) scenarios, let us now summarize main functionalities that are to be realized at this stage of the project. (a) *Cars interacting with traffic lights*. Here, intelligent intersection systems will be proposed. They should make decisions based on the current situation, e.g., number of cars waiting for the green light, number of cars known to approach, etc. Furthermore, our system will place traffic lights in their actual locations within the city. (b) *Citizens interacting with public transport system*. Citizens that plan to use public transport will specify where they are, and where they are going. Here, the infrastructure should make decisions based on, for instance, number of travelers waiting at a given bus stop, and the estimated time of traveler's arrival at the stop. Moreover, while the schedule of public transport may be chosen arbitrarily, we believe that it should be based on the real-life schedule of the simulated city. Therefore, we have used an actual Warsaw public transport timetable. This decision allows comparison of simulated scenarios with real-life ones, as the actual performance of the public transport system can be used as a reference.

3.1 Agent Specification

When designing the agent system, we have used GAIA methodology [8]. As a result, we have identified six different types of agents.

- *CentralAgent*: This is the core agent, which retrieves all data necessary for the simulation to be instantiated. Next, it manages the creation of agents needed in the system. Then, it activates agents and starts simulation. Finally, it is responsible for GUI, and time measurements (for the test cases).

- *LightManager*: Agent(s) that manage(s) all lights on a particular crossing (one agent per one crossing). Furthermore, it decides when, and which group of lights, will change color. Its responsibilities include also: communicating with agents representing vehicles and travelers, keeping track of their number (as they approach and leave), and informing them when to pass. *LightManager* should have a strategy (possibly, strategies), aiming at optimizing traffic.
- *CarAgent*: Agent(s) representing cars traveling in the city. *CarAgent*s inform *LightManager*s about their intention of passing the light in two cases: (i) when they have already arrived, and (ii) when they are approaching the light. There is no imposed limit on the number of *CarAgent*s.
- *TravelerAgent*: Agent(s) which represents traveler(s). It communicates with *StationAgent*s to notify them about travel details needed to facilitate use of public transport. Moreover, while traveling by public transport, *TravelerAgent* receives information, from *BusAgent*, when it reaches the destination. Finally, *TravelerAgent*s can communicate with *LightManager*s, the same way as the *CarAgent*s. There is no imposed limit on the number of *TravelerAgent*s.
- *BusAgent*: Agent(s) that represent public transportation vehicles, traveling between assigned stations. *BusAgent*s communicate with *LightManager*s the same way that *CarAgent*s do. *BusAgent*s communicate also with *StationAgent*s and *TravelerAgent*s (to keep track of their number and to inform them, later, that they arrived at the desired station). There is no imposed limit on the number of *BusAgent*.
- *StationAgent*: Agent(s) representing public transport stops. Their number equals that of the number of stops in the simulated area. It is assumed that stations are placed based on their real-world location (to use transport schedules). *StationAgent*s communicate with: (i) public transport (*BusAgent*s) to receive information about their location; (ii) *TravelerAgent*s to receive information about potential passengers. They also decide (apply strategy) about possibly delaying the time of bus departure to serve late passengers.

3.2 Communication Flow

The agent knowledge map is shown in Fig. 1. This map overlaps with the information exchange map. Connected agents send messages both ways.

CarAgent, *BusAgent*, *TravelerAgent* $\longleftrightarrow$ *LightManager*. Upon starting the journey, and after passing through the previous light, agents send an *INFORM* message to the *LightManager*, representing the next light, containing the ETA. After reaching the light, they send a *REQUEST-WHEN* message to the *LightManager*, asking for the information when to pass (note that, to make system more fun, for the time being, we assume that passing is "message-managed"), then the *LightManager* responds with the *AGREE*, and waits for the moment when vehicle/traveler is allowed to pass. When this moment occurs, *LightManager* sends a *REQUEST* message to the vehicle/traveler. The *REQUEST* message is sent only for simulation purposes. Obvi-

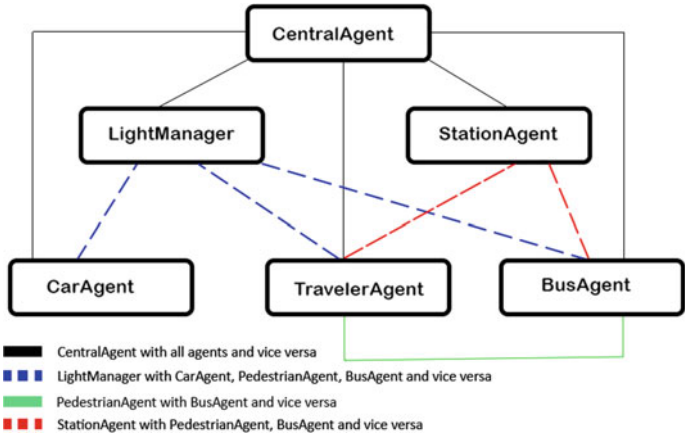

Fig. 1 Agent knowledge and information exchange map

ously, pedestrians can see the light without the need to check the message. The vehicle/traveler responds *AGREE* and passes through the light.

TravelerAgent ⟷ *StationAgent*. When moving to the bus stop, *TravelerAgent* sends an *INFORM* message to the *StationAgent* containing the ETA, and the number of bus line of interest. When the traveler arrives at the bus stop, *TravelerAgent* sends a *REQUEST-WHEN* message to the *StationAgent*, asking to enter the bus as soon as possible. Here, the *StationAgent* acknowledges with the *AGREE* message. When the bus arrives, the *StationAgent* sends a *REQUEST* message, asking to enter a particular bus, using which the traveler will move toward her/his destination. The request contains the *BusAgent* ID, so that the *TravelerAgent* can communicate with it.

BusAgent ⟷ *StationAgent*. While approaching the next station *BusAgent* sends the *INFORM* message to the *StationAgent*, containing the ETA. After arrival, *BusAgent* sends a *REQUEST-WHEN* message to the *StationAgent*, asking to gather people and pass. Next, the *StationAgent* responds with the proposed departure time (capturing approaching passengers). Upon departure, the *BusAgent* informs the *StationAgent* about this fact.

TravelerAgent ⟷ *BusAgent*. When the traveler enters the bus, he/she waits for the *REQUEST* message from the *BusAgent*, which will indicate arrival at the destination. Upon reception of such message, *TravelerAgent* answers (*AGREE*) to confirm that the traveler left the bus.

4 Simulator Implementation

The proposed traffic simulation system has been implemented in Java 8, using the JADE agent platform (version 4.5). Information regarding location of traffic lights and bus stops has been obtained via Overpass API (OpenStreetMap). To get data for

a given route, system sends requests to Overpass API and parses the results using the GraphHopper library [9] to find traffic lights and stations on the route. Furthermore, information about (all) bus timetables was obtained from the Warsaw public transport API [10].

To visualize simulated traffic, we prepared an user interface with a customizable map (provided by JXMapViewer2 [11]). Our system allows to select, which entities spawn on the map (cars or/and travelers or/and buses), choose the simulation area, adjust simulation time, and number of created agents. To measure time, we have *BusAgent*, *PedestriamAgent*, or *CarAgent* communicate with the *CentralAgent* that "runs the system time" and inform it about pertinent events.

Since, for the time being, our focus was on testing the usability of the system, only extremely simple strategies (managing when buss will leave and when lights will change) were implemented. Exploring possible strategies is part of research planned for the future.

In Fig. 2, we present an image of a simple simulation.

Here, we can see generated cars, with their corresponding routes, the simulation area with the chosen center, and crossing lights (from the Overpass API).

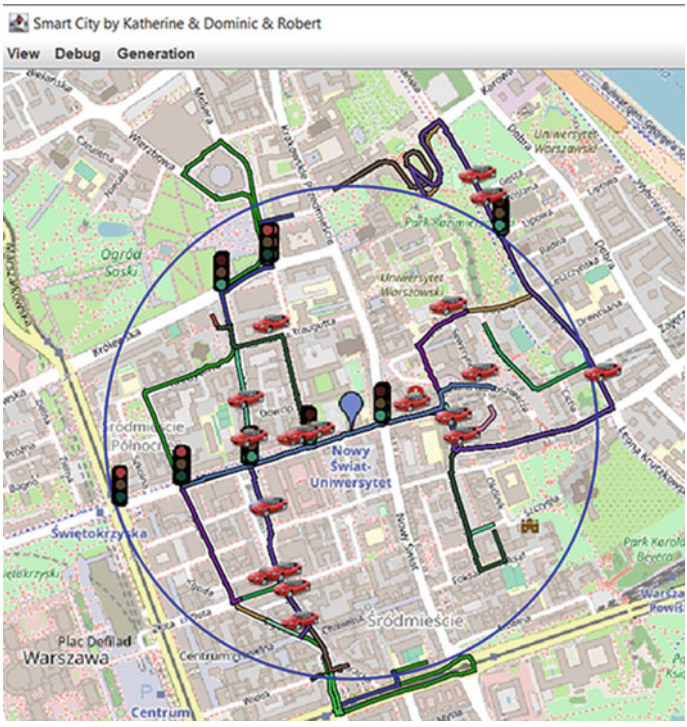

Fig. 2 Example of simulation

5 Experimental Validation

The developed system realizes use case scenarios, described in Sect. 1. This allows us to, jointly, validate the design of the system, and address two questions: (1) Does vehicle lights communication reduce journey time? and (2) Does traveler station communication improve travelers' life?

To answer the first question, we have measured the average time it takes for a car to travel from point A to point B with, and without, smart lights. Normally, lights change, without delay, after 60 time units. Here, light smartness will be represented by possible delay, of up to 30 time units, in changing the lights (depending on the number of cars that are to pass the crossing). Worth mentioning is that this 2:1 ratio needs further adjustment to ensure better strategy effectiveness. In experiments, we have increased the number of cars $5, 10, 15, \ldots, 35, 40$. The test is designed to gradually increase the traffic, while maintaining the same size of the simulation area. When the traffic is dense enough, strategy is more needed, because the queues of cars, waiting to cross (all) lights gets filled quite fast, in comparison with the "normal conditions." Results are depicted in Fig. 3.

It is noticeable that applying the strategy for light management slightly—by ∼6%—improves the average journey time. Here, note that this difference concerns area with just four intersections.

For the second question, we have also measured average journey time, with possible delay of up to 60 time units at a bus stop. Results have been mixed. In the simplest case, when each bus stop allowed delay, such delays could accumulate (even if they were relatively small) and, while some travelers were "winners," others were "losers." This resulted in strategy change, and waiting only if the *total delay* time was less than 60 time units. This, in turn, practically eliminated the overall average travel time gains. Therefore, returning to the issue of making public transportation more flexible is one of the open issues that require further research.

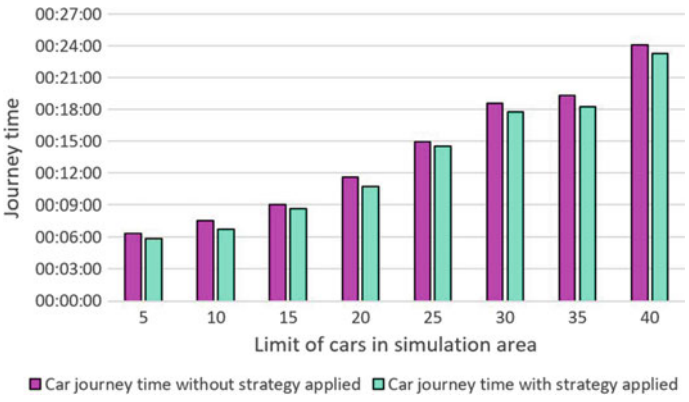

Fig. 3 Experimental validation for average car travel time

Interestingly, the current design of the simulator is heavily dependent on the quality of the Internet connection. Each time a vehicle/traveler is created, simulator Internet-connects to external APIs, for route calculation. Weak Internet connection resulted in not realizing the preset generation interval, directly affecting the traffic conditions (interfering with the structure of planned experiments).

6 Concluding Remarks

In our work, we have designed and implemented an initial version of an agent-based smart city traffic simulator. Developed system works on a map provided by the OpenStreetMaps and uses actual street light locations and public transportation schedules. We have initially validated the design against two use case scenarios. We were able to establish that improving car flow can be achieved when cars communicate with traffic lights. Here, even an extremely simplistic strategy of "slightly delaying light change in response to incoming traffic" can reduce the average travel time. The situation is more complex with travelers using public transport. Here, the obvious delay strategy did not work, because of the constant accumulation of passenger travel time. While, after reconsideration, a "better" solution was found, it was not as effective as desired. Hence, there is a lot of open questions, how to improve public transport management. Nevertheless, let us recall that, at this stage, our goal was to develop (and validate) a working agent-based smart city traffic simulator. With the working simulator in place, we can start extending it and explore approaches to traffic management.

Here, we plan to consider multiple pathways. (1) Making public transportation multi-modal, by introducing rail, subway, tram, trolley, etc. Each of them would be represented by an agent, which would be based on the design of the *BusAgent*. (2) Including also other forms of transportation, such as: taxi, bicycle rentals, electric scooters, etc., as means of getting from point A to point B, either directly, or in multi-modal combinations. (3) Exploring ways of proposing complex multi-modal transport routes, while making them flexible and optimal. The latter would require research in answering the question: what makes the transport route optimal (e.g., price, time, CO_2 consumption, etc.). (4) Introduction of electric vehicles and charge stations. (5) Adding autonomous vehicles to the mix or replacing human-driven vehicles with autonomous ones completely. (5) Adding (possibly autonomous) freight/supply transportation and exploring models of optimization of their use. (7) Exploring strategies for each stakeholder entity (agent representing it). We will proceed with this work and report on our progress in the next publications.

References

1. Larco N (2018) How will autonomous vehicles transform our cities? TEDxCollegePark
2. Ganzha M, Jain LC (2013) Multiagent systems and applications. https://doi.org/10.1007/978-3-642-33323-1

3. Grau J, Romeu M (2015) Agent based modelling for simulating taxi services. In: The 6th international conference on ambient systems, networks and technologies (ANT-2015). https://doi.org/10.1016/j.procs.2015.05.162
4. Leszczynski M, Niedabylski M, Kutkowski R, Ganzha M, Paprzycki M (2016) Software agents in support of a taxi corporation. In: 2016 20th ICSTCC. https://doi.org/10.1109/ICSTCC.2016.7790703
5. Bösch PM, Ciari F (2015) Agent-based simulation of autonomous cars. In: 2015 Proceedings of the American control conference 2015, pp 2588–2592. https://doi.org/10.1109/ACC.2015.7171123
6. Duzha E, Hakrama I (2015) Public transportation simulation by using agent based simulation: case of Tirana. In: ISTI 2015, Tirana, Albania
7. Zhu K, Bos A (1999) Agent-based design of intermodal freight transportation systems
8. Wooldridge M, Jennings N, Kinny D (2000) The Gaia methodology for agent-oriented analysis and design. https://doi.org/10.1023/A:1010071910869
9. https://github.com/graphhopper/graphhopper
10. https://api.um.warszawa.pl/
11. https://github.com/msteiger/jxmapviewer2