Towards a Bijective Co-simulation Model Between Physical and Virtual Environments, Adapted to a Platform for Autonomous Industrial Vehicles

Moïse Djoko-Kouam, Alain-Jérôme Fougères

Automation, Control and Intelligent Systems
2023; 11(2): 27-44
http://www.sciencepublishinggroup.com/j/acis
doi: 10.11648/j.acis.20231102.12
ISSN: 2328-5583 (Print); ISSN: 2328-5591 (Online)

Towards a Bijective Co-simulation Model Between Physical and Virtual Environments, Adapted to a Platform for Autonomous Industrial Vehicles

Moïse Djoko-Kouam[1, 2], Alain-Jérôme Fougères[1, *]

[1]IT and Telecommunications Department, ECAM Rennes Louis de Broglie, Bruz, France

[2]Institute of Electronics and Digital Technologies of Rennes, CentraleSupélec, Rennes, France

Email address:

moise.djoko-kouam@ecam-rennes.fr (Moïse Djoko-Kouam), alain-jerome.fougeres@ecam-rennes.fr (Alain-Jérôme Fougères)

[*]Corresponding author

To cite this article:

Moïse Djoko-Kouam, Alain-Jérôme Fougères. Towards a Bijective Co-simulation Model Between Physical and Virtual Environments, Adapted to a Platform for Autonomous Industrial Vehicles. *Automation, Control and Intelligent Systems*. Vol. 11, No. 2, 2023, pp. 27-44. doi: 10.11648/j.acis.20231102.12

Received: May 31, 2023; **Accepted**: June 29, 2023; **Published**: July 11, 2023

Abstract: One of the major challenges faced by Industry 4.0 is the use of Automated Guided Vehicles (AGVs) and, more broadly, autonomous mobile robots. While autonomy in road transportation vehicles can already be well characterized, it is a different story for autonomous vehicles used in industries, such as Autonomous Industrial Vehicles (AIVs). The implementation and deployment of AIV fleets in industrial sectors encounter various issues, including vehicle localization, employee acceptance, traffic flow, and the ability of vehicles to adapt to fluctuating and dynamic environments. The challenge that autonomous vehicles represent for the future of the digital industry is so significant that it makes sense from our vision to go through a step of joint physical and digital simulation of these vehicles and their environments. This step would assist industrials in their respective development and fine-tuning activities. The objective of this article is to demonstrate that all the elements available to us at present (research conducted in our laboratory, technological building blocks, operating scenarios already carried out, state of the art) logically allow us to project ourselves towards a Co-Simulation platform based on a bijective model between physical and virtual environments. Furthermore, this projection is enriched by the idea of a digital component of the platform, capable of taking into account in an agnostic way, the different possible forms of the physical part of this platform. Thus, simulation enables the consideration of constraints and requirements formulated by manufacturers and future users of autonomous vehicles. Our approach is progressive, as presented in this article, and it is based on experiments with Co-Simulation platforms that combine physical and virtual approaches. We provide a detailed description of our AIVs and their traffic environments. The static architecture of the platform is described using class diagrams. The dynamic behavior of the platform is described, thanks to sequence diagrams, state diagrams, and algorithmic flowcharts. We also propose an approach to estimate the position of AIVs based on the combination of matrix-based tagging with a section change management technique. As a result of the presented approach, all of these diagrams make it possible to document different operating scenarios of the Co-Simulation platform. Beyond these results, and to complement them before concluding, we describe several application cases related to both algorithmic position estimation metrology and electric battery characterization. This serves to illustrate the potential value of our Co-Simulation platform model.

Keywords: Autonomous Industrial Vehicle (AIV), Matrix Beaconing, Co-simulation Platform, AIVs Position Estimation, Agent-Based Simulation, Fuzzy Agent

1. Introduction

Among the most significant challenges of Industry 4.0, the need to develop and optimize data, product, and material flows within manufacturing companies is paramount. Specific technological elements have been identified [1, 2], such as the use of Automated Guided Vehicles (AGVs) and autonomous mobile robots, to achieve this objective. While the level of

vehicle autonomy in the field of road transportation has been clearly characterized with the six levels of autonomous driving established by the Society of Automotive Engineers [3], a similar characterization is lacking for autonomous vehicles used in the industrial context, also known as Autonomous Industrial Vehicles (AIVs) [4]. Implementing and deploying fleets of AIVs in industry presents several challenges, including employee acceptance, vehicle localization, smooth traffic management, and the ability of vehicles to perceive and react to changing and dynamic environments. As a result, the autonomy of these vehicles has been limited to pre-established trajectories. Therefore, the possibility of exchanging information among different AIVs within a fleet should enhance their autonomy in terms of:

1) Adaptation to traffic constraints: This includes the ability to adjust to dynamic changes in the AIV environment in areas such as storage zones and production lines. This adaptability is greatly facilitated by advancements in Artificial Intelligence (AI) and Internet of Things (IoT) technologies [5], enabling improved environmental perception;
2) Decision-making: Even in the presence of incomplete, uncertain, or fragmented information [6], AIVs must be able to make informed decisions;
3) Communication with other AIVs in the fleet as well as with infrastructure or associated individuals, commonly referred to as "V2X communications" [7] and;
4) Reduction or control of energy impact, independently of traffic constraints [8].

Localization techniques are undoubtedly an important link in the chain of industry 4.0. The recent literature offers many works on this subject. Article [9] for example, presents a location solver based on a camera observation model, and exploiting ground markings. This solver is itself based on dynamic Kalman filtering, and the results in real traffic conditions have shown the promising character of this approach. In their article [10], Jung & al propose an indoor localization technique using an active infrared sensor. The resulting uncertainty increases with distance from the landmark. The proper functioning of the system requires an optimal arrangement of these same landmarks. An approach based on a geolocation model adapted to vehicular networks is the subject of the article [11]. Inexpensive smartphone sensors are used to estimate or improve the position of vehicles. These estimates are based on different types of data collected, such as inertial unit data, GPS, RSSI, as well as road maps. We can also mention the study presented in [12], which concerns a system allowing determining the position of a mobile robot using artificial markers placed in its environment. The robot is able to autonomously capture and process images of this environment, in order to calculate its location. The experimental results show that the robot is able to achieve an average positioning error of less than 5 cm.

For the proper functioning of a simulation or Co-simulation platform, it is necessary to have a position estimation strategy, based on a set of estimation techniques and algorithms. These algorithms are indeed unavoidable, since the vehicle which seeks its position, no longer has access to a sure mark, likely to provide non-ambiguous coordinates. The available literature shows a wide range of studies and works on this subject. As an example, we can see in reference [13], the presentation of a position estimation system for a robot vehicle, designed to be used in office or factory environments. The proposed position estimation scheme boils down to an odometry, completed by a matching algorithm whose mechanism is based on the congruence between the distance data and a 2D map of the environment. Being able to have simulation or even Co-simulation platforms seems to us to be of major use for the development of the technological bricks of industry 4.0. Various studies and works of the literature deal with this subject. A study on the design of a real-time simulator of collaborative and autonomous vehicles is presented [14]. The authors propose different parallel computing architectures, with the search for speed and precision as an important criterion. The dynamics of vehicles equipped with on-board sensors are simulated in a road environment. We should also mention the work presented [15] concerning a model dedicated to automated storage and using autonomous vehicles (AV) and elevators. The aim of this study is to find the most optimal values possible, for a number of AVs and elevators in the system, and allowing to obtain high performances in predefined scenarios.

In view of the publications on the simulation platform models proposed by the literature, no study to our knowledge concerns a Co-simulation platform, including on the one hand a potentially changing material part, on the other hand a digital part that can represent and control the physical part in a generic way.

In this article, we present our research on these questions from different angles, and give particular importance to both sharing information on what has been done and projecting what is planned. We start with a state of the art concerning firstly communication between and with autonomous vehicles, secondly the development of simulation platforms for AIVs, and thirdly, approaches and systems for estimating position. Then we present the core of our research: a Co-Simulation plateform and a method of AIV position estimation. Finally, we present some application cases, declined in two main fields, on the one hand, that of the metrology of position estimation algorithms, on the other hand, that of the characterization of electric batteries.

2. State of the Art

2.1. Communication Among Vehicles

The experimental self-driving vehicles currently roaming the world's roads to collect data and rack up miles are not engaged in cooperative interaction with their environment. Instead, they rely on integrated sensors [16, 17]. These sensors can include cameras, radars, GPS, lasers, or lidars, along with internally gathered data (such as odometer readings and wheel condition evaluation) to obtain raw information used for constructing an accurate representation of their surroundings

[18]. The control and command system of a vehicle then combines its perception with pre-existing information, such as a detailed map [19] or a learned representation of its traffic environment [20, 21], to select a driving trajectory and position itself on the road. A similar situation is observed with the increasingly deployed AIVs in industrial facilities; however, their adaptability remains highly limited. In recent years, the automotive sector has formed partnerships with telecommunication actors to develop communication standards that promote direct collaboration between vehicles through the exchange of structured information [22]. For instance, instead of solely relying on perceiving proximity to the vehicle in front, a vehicle can now initiate deceleration or braking based on indications emitted by the preceding vehicle. This form of coordination significantly reduces reaction time to critical events, enhancing safety while contributing to profitability. Vehicles can be grouped into convoys (platooning) on highways [23], or their passage at intersections can be synchronized to optimize traffic flow [24].

2.2. Simulation Platforms for AIVs

Before starting to test large-scale traffic scenarios involving autonomous vehicles in industrial contexts or other more complex contexts, it seems essential to us to consider the simulation involved. It should be noted that many methods are used for these tests [25]. Perhaps the greatest benefit of running a simulation is that actionable results can be obtained without applying a scaling factor. More generally, the field of AIV simulation techniques, whether or not based on a physical platform, is currently being investigated from different angles. The lack of consideration of the scale factor in many use cases is pointed out in reference [26], which proposes a simulation and test model on a platform, in order to help in the design of safe autonomous vehicles. The authors note the lack of tools for rigorous and scalable tests in the implementation of platforms for AIVs. One of the bases of their proposition is that real-world testing, while rightly considered a gold standard, is still likely to put the public at risk. The work on the platform presented in that reference announces significant time savings compared to real-world tests, which makes it possible to effectively evaluate an autonomous vehicle, seen as a simple black box. Still in the field of simulation models, we can see [14] a study on the design of a real-time simulator of collaborative and autonomous vehicles. The authors propose different parallel computing architectures, with the search for speed and precision as an important criterion. The dynamics of vehicles equipped with on-board sensors are simulated in a road environment. We can also mention the study of Kade & al [27] on a low cost mixed reality simulator for Industrial Vehicles. Human panel test results indicate that participants have a more realistic and natural view with the immersive experience of the Mixed Reality Simulator, unlike the Classic Simulator on PC. The authors see their simulator as a potential tool for designing and testing vehicle concepts.

The literature offers a highly uneven treatment between, on the one hand, the field of autonomous industrial vehicles, which has relatively few studies, and on the other hand, autonomous vehicles, whose progress is extensively documented and reported, including in non-specialized circles [28]. The remarkable adaptability of autonomous vehicles, particularly AGVs, to their environment is now a well-established fact. Through a combination of physical and IT solutions, it is possible to prioritize a distributed communication scheme, resulting in even more autonomous vehicles. In these cases, the literature often proposes agent-based approaches [29, 30]. In our view, the concept of fuzzy agents is a suitable means to account for the non-deterministic dimension when modeling the behavior of simulated vehicles [31, 32].

2.3. Estimating the Locations and Positions of AIVs

Estimating the position of a vehicle in a traffic area involves providing an approximate value of the vehicle's position given its environment. There is a very rich scientific literature on estimation theory, which generally covers a wide range of techniques and ideas, with the most common ones receiving frequent attention. These techniques can find their field of application in a wide range of problems. For example, in the field of parametric estimation approaches, one can mention techniques such as weighted least squares estimators, maximum likelihood estimation, or minimum mean square error estimators [13]. In reference [33], a Bayesian approach for multi-sensor localization with context-based data fusion is proposed. The approach is based in particular on the selection of the most efficient set of sensors, with extensive consideration of the precision criterion. The authors claim the robustness of this approach against localization errors. According to current and upcoming techniques and algorithms presented in the state of the art [34], a dominant trend emerges. It can be observed that localization, using both incremental and relative approaches [33], allows for a given vehicle to calculate its position and orientation, given the sequence of its movements starting from a known reference position. It is possible to calculate the position of a robot or a vehicle in its internal or external environment by implementing exteroceptive sensors. This type of localization is referred to as absolute [33]. Generally, there are two localization strategies to choose from, based on artificial or natural landmarks (such as beacons or GPS, for example). By definition, absolute localization avoids drift over time, unlike relative localization, which has the main disadvantage of losing visibility of the landmarks used by a vehicle to calculate its position. Reference [35] is also interesting on this same theme, because the authors present applications of the Local Positioning System (LPS) on the tracking of humans, objects, animals and automatic guided vehicles (AGV). Different approaches derived from this technology have limitations. However, Ultra-WideBand (UWB) technology stands out as being able to achieve centimeter-level accuracy based on Time of Arrival (TOA) or Time Different of Arrivals (TDOA). In our study, the issue of noise was of significant importance. Indeed, the measurements leading to the estimation were likely to be corrupted, as they were affected by noise. The

consequence can be the use of an input that introduces uncertainty in the interference. At the core of the estimation problem, uncertainty is indeed a reality, without which many problems would find simple algebraic solutions [13].

3. A Co-simulation Platform

3.1. Presentation of the Simulation Platform

Our Co-Simulation platform named *CoSiVA,* is structured around two main components. On the one hand, an object-oriented digital simulation software subset, which aims to simulate the dynamic behavior of various vehicles in a virtual environment. On the other hand, a physical sub-assembly used to execute different traffic scenarios involving a group of miniature vehicles. These scenarios intend to synchronize and demonstrate identical movements in both virtual and physical simulations. Figure 1 illustrates the overall architecture of the platform. If a vehicle encounters an obstacle on the physical platform, that obstacle must also be displayed in the graphical interface of the digital platform. This Co-Simulation platform also makes it possible to deploy augmented simulation scenarios. This includes, for instance, adding new vehicles or individuals to the traffic area, as well as considering virtual and direct communications between Autonomous Intelligent Vehicles (AIVs). We have developed a Human-Machine Interface (HMI) allowing a user to supervise and interact with the platform. Thus, it is possible to visualize a simulation (for example the movements of the AIVs on the virtual circuit), to manage the components present in the virtual environment of the simulator (for example to define the priorities between AIVs, to define the RFID TAGs present on the circuit of AIVs), to send commands to the virtual AIVs (for example to slow down, stop or restart an AIV). A movement requested in the virtual environment is supposed to materialize on the physical platform. It is also possible, thanks to this HMI, to display the status (Normal, stopped, slow speed, medium speed, fast speed) of each AIV, and finally, to display the traces of communications between the different agents present in the system (Figure 2). This HMI allows users to augment the simulation by introducing a new virtual vehicle into the set or by having a human operator appear on the vehicle traffic map. Thus, as can be seen in figure 2, the added AIV appears in orange color in the "*Supervision of AIV*" frame (in the "*AIV number*" list), and as an orange disc in the "*Circuit*" frame.

On our physical platform, the AIVs all have similar characteristics. They are small in size and capable of following the road using various line tracking algorithms. They can stop in the presence of an obstacle and geolocate themselves within the traffic area. The AIVs communicate through a radio link, exchanging information such as position and velocity with roadside equipment. These AIVs are also capable, given a set of actions to be taken, of opting for the best one, taking into account the information coming from the environment.

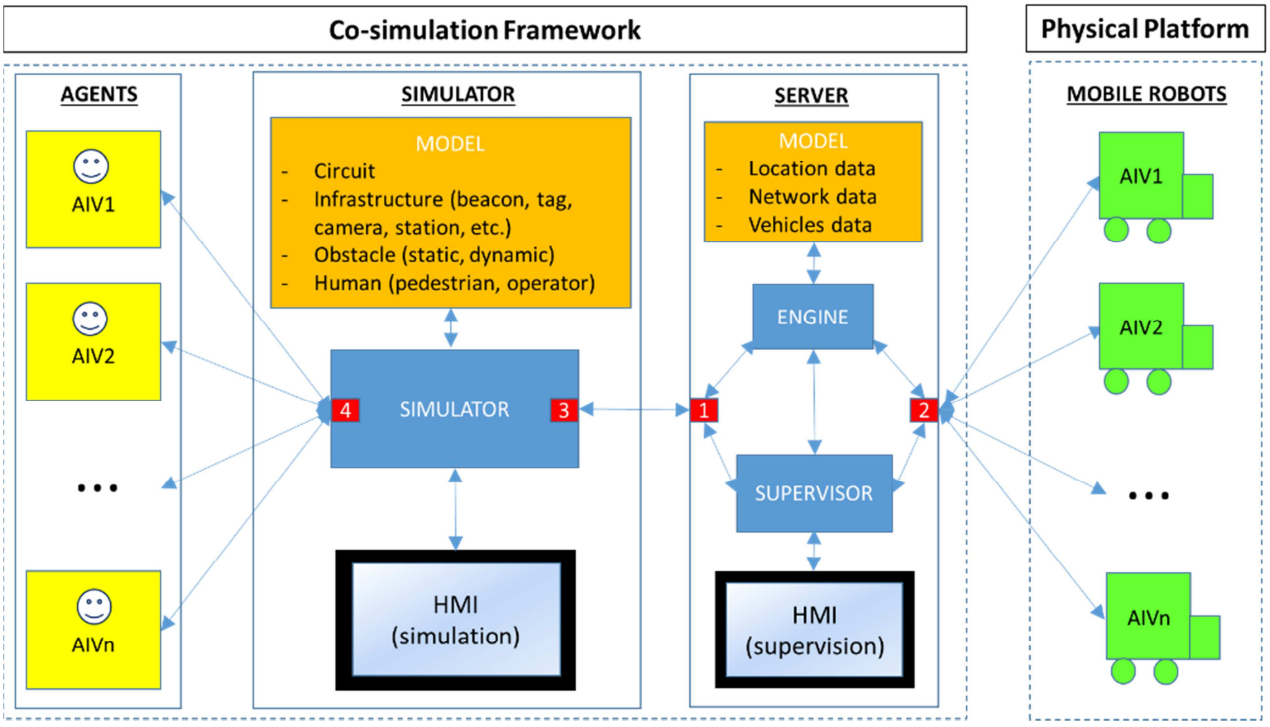

Figure 1. *Architecture of the Co-simulation platform CoSiVA.*

It should be noted that according to the expected behavior of the simulated AIVs, they are designed as fuzzy software agents. This means they can both manage their own movements and respond to instructions from the simulator through the server. For this purpose, the fuzzy agents dialogue with the server, or communicate with each other.

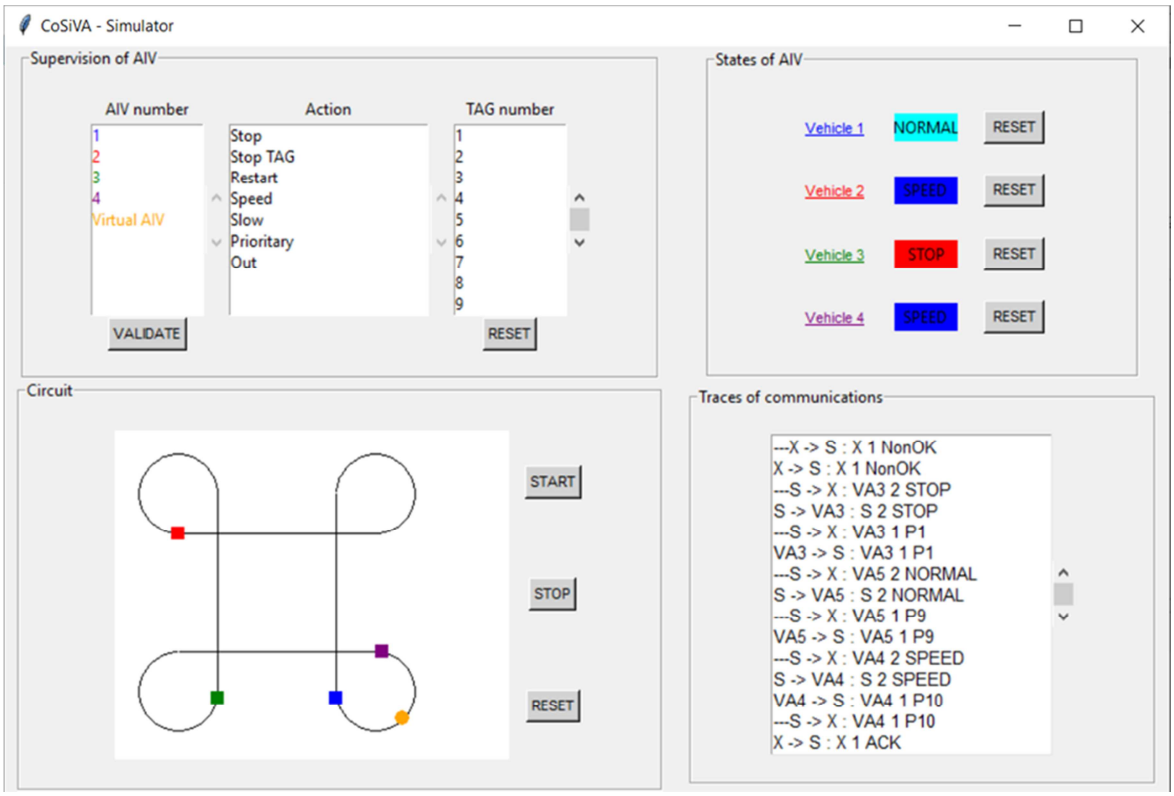

Figure 2. *Simulator HMI composed of 4 frames: 1) Supervision of AIVs, 2) States of AIVs, 3) Circuit, 4) Traces of communications between AIVs (note that the integration of a virtual vehicle is visible using the color orange).*

3.2. Presentation of the Physical Platform

Among the different scenarios that could potentially involve autonomous vehicles, our focus was on those that could most easily lead to traffic congestion. Taking this particularity into account, we have logically been led to define a traffic scheme consisting of four loops, as can be seen in Figure 2 for the GUI, and Figure 3 for the physical platform.

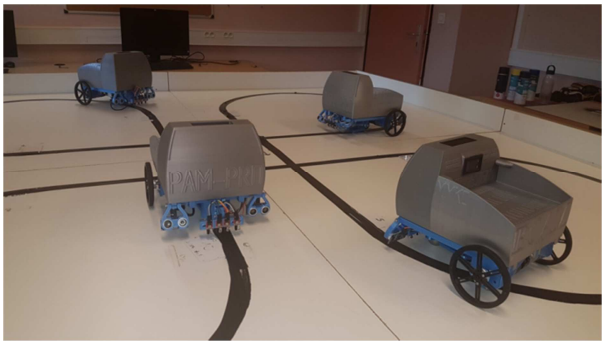

Figure 3. *AIVs on the physical platform.*

We thus defined a traffic platform as an entity consisting, on the one hand, of a circuit made up of four curving sections and four straight sections placed end- to-end to form what we designated “CircuitLacet4”, and, on the other hand, of a set of 12 RFID tags distributed along the circuit. Each quadrant had three markers, with each marker being represented by one RFID tag, so that any nearby vehicle could position itself on the circuit in the traffic zone.

Line following and close obstacle detection are part of the natural operation of Autonomous Intelligent Vehicles (AIVs). We found it useful to go beyond these basic functionalities, and this is how we created three speed profiles, in order to fix the kinematic behavior of our AIVs. In the operating scheme of the Co-Simulation platform, each vehicle must be able to transmit its position to the server, relative to a set of RFID tags placed on the road circuit.

3.3. Internal Architecture of an AIV

Each AIV has been outfitted with a set of components that allow us to test our most interesting scenarios. These components include control modules or sensors for RFID tag detection. These modules also include obstacle detectors and power supply regulators. Furthermore, these modules include communication devices to transmit data to the server, and ensure the reception of instructions from the server concerning the movements of the AIV. Finally, these modules include motors to propel the vehicle, as well as a ground marking tracking system to guide the AIV. It is essential that each system function has a clearly identifiable electronic module, and each function should be easily verifiable during the dynamic search for malfunction sources. To ensure that each autonomous vehicle can perform its assigned tasks, it was necessary to establish an organization as structured as possible regarding software, hardware, and electronics. This work resulted in a comprehensive internal mapping of autonomous vehicles. Figure 4 illustrates the different modules and their respective connections.

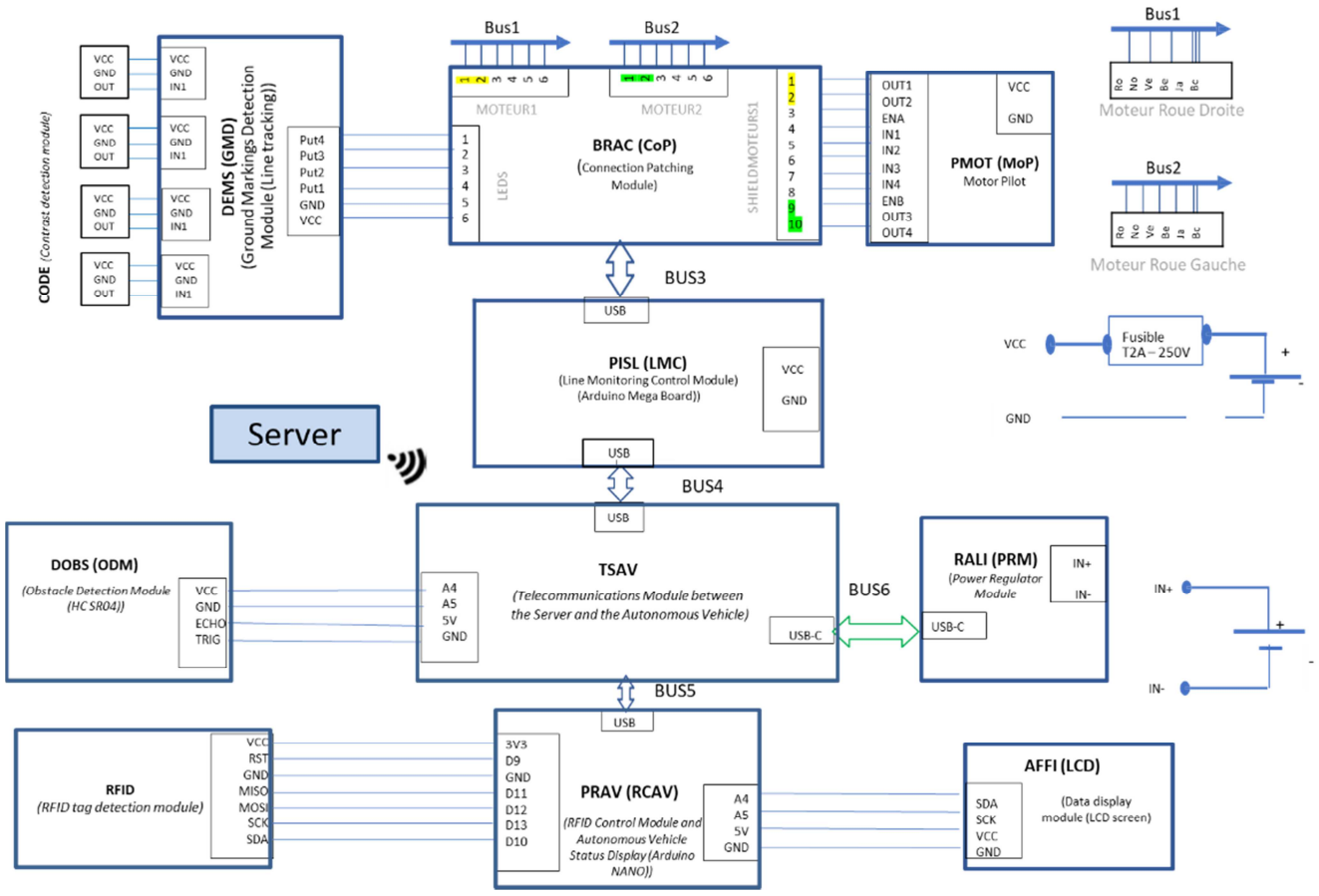

Figure 4. *Internal architecture of the AIV.*

The RFID tag detection and display control module (PRAV) is physically connected to both the RFID module and the display module (AFFI). The modules that detect ground markings (DEMS) and those that control the wheel motors (PMOT) are connected to the line tracking control module (PILS) via the connection module (BRAC). As a central connection point, the BRAC module acts as an intermediary by consolidating and optimizing the multiple paths of the electrical connections, which are already somewhat disorganized, making it difficult to locate malfunction sources. To facilitate this connectivity, we designed the BRAC module to connect to the PILS module by overlaying, with a form factor similar to an Arduino board, in an interface designated as Bus3 in Figure 4. The telecommunications and obstacle control module (TSVA) also drives an obstacle detection module (DOBS) through a direct four-wire connection and communicates with a remote server. It is important to note that the TSVA is implemented using a Raspberry Pi board, which requires sufficient and stable power supply. That's why the power regulator module (RALI) is present, connected to the input of TSVA via Bus6 through a USB-C interface. In a network of multiple connections, the TSVA module is the central module of the system, surrounded by the server via the radio link, the DOBS module via a direct connection, the RALI module via Bus6, the PRAV module via Bus5, and finally the PILS module via Bus3. Figure 4 clearly illustrates the central positioning of the TSVA module in terms of both functions and systemic connections.

3.4. Traffic Zone Agnostic Representation

We explain here, the mechanism of creation, and blind display of the traffic zone in the graphic space of the simulator. The "Simulator" entity is easily identifiable on the diagram of figure 1, as on that of figure 5. We mean by blind display, the fact of being able to display an object without actually knowing it. The blind nature of the creation and display of this circulation zone therefore comes from the fact that the simulator has the capacity to create and manipulate this circulation zone, without knowing either the nature or the format of the concrete circuit used. From this comes the agnostic character of the representation in the space of the simulator, with respect to the nature of the circulation zone. The Circuit component can be seen on the class diagram in figure 5.

The concrete circuit could be indifferently of a road nature, or of an industrial nature, its shape could be mainly with very low curvature, or on the contrary with very high curvature, without this having any impact on the actions of the simulator, on any object of Circuit type. This operation is made possible by the fact that the notion of circuit is thought out at two very different conceptual levels. The first conceptual level

considers the concrete circuit with all its characteristics and particularities, which come from the fact that it is for example a road circuit of the "CircuitLacet4" type as shown in figure 6, or a circuit of industrial type like "CircuitIndus8" as can be seen in figure 7.

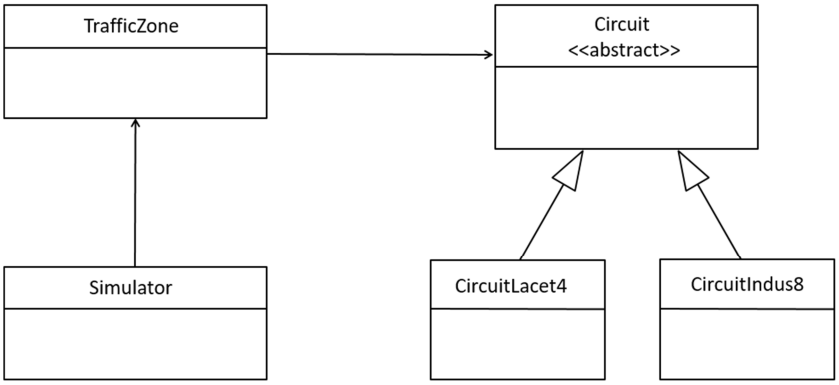

Figure 5. *Abstract and concrete circuit patterns in traffic area (illustration with the 2 circuits CircuitLacet4 and CircuitIndus8).*

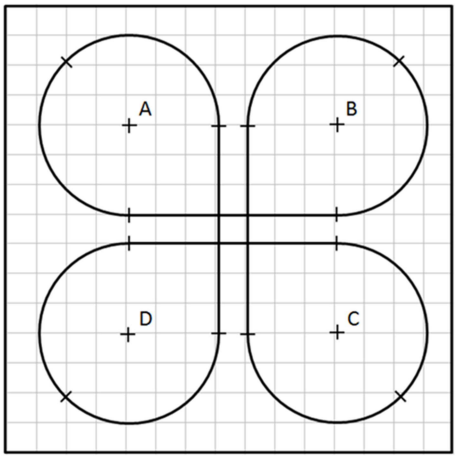

Figure 6. *"CircuitLacet4" circuit profile.*

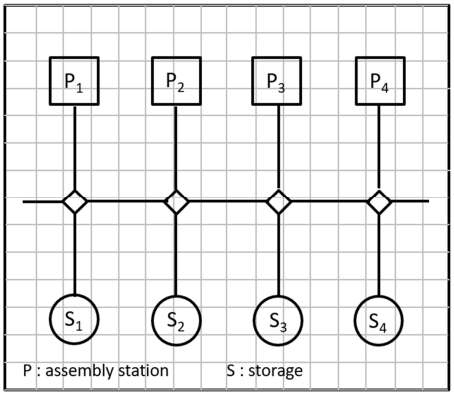

Figure 7. *"CircuitIndus8" circuit profile.*

It is quite obvious that any entity that would come to manipulate such a concrete circuit would be obliged to know all these geometric and topological particularities (length, radius of curvature, etc.). The second conceptual level considers the abstract circuit. The important thing here is the idea of circuit, the latter being reduced to its factorizable version on the set of all possible specific circuits. From the simulator's point of view, by confining the vision of the circuit to that of the abstract circuit, it becomes obvious that the simulator does not need to make any a priori assumptions about the real circuit on which it is working, nor about its nature, nor on its form. The knowledge of the specificities of each concrete circuit is therefore left to the side of the physical part of the Co-simulation platform.

The traffic area that contains the concrete circuit, simultaneously hosts a set of beacons distributed in a matrix according to a grid that we can see both in figure 6 and in figure 7. This markup of the traffic area allows us to define a set of fixed and reliable references of positions on the circuit. We preferred this beacon distribution topology compared to a linear topology applied directly on the circuit, in order to better take into account, the constraint of adaptation to any type of circuit. Furthermore, we have planned to configure the graduation of the associated grid, which makes it possible to vary the density of the markup, and thus, to be able to easily test the influence of this density on the operating performance of the Co-simulation. Thanks to this matrix markup, the current position of an AIV can be rectified in real time, when the latter is on a beacon.

3.5. Scenario of Blind Creation and Display of the Circuit by the Simulator

The scenario of blind creation and display of the circuit by the simulator in its graphic space involves various objects in the virtual environment of the simulator. The *AIV_Server* plays a central role in the platform as well as the Co-simulation strategy, through its position as a two-way gateway, between the simulator on the one hand, and the physical circulation area of the AIVs on the other. It is indeed this *AIV_Server* which will receive the specific data of the physical part, to inform the requests of the simulator. The real-time module (*RTM*) exists as a unified entity in the simulator. The *RTM* only exists in the physical part of the Co-simulator, in the form of separate data, hosted in a configuration file. Once created and populated with data from the configuration file, the *RTM* is ready to respond to requests from the simulator, concerning entities not only having an existence in the physical part of the platform, but also specificities related to their concrete nature. It will be for

example a traffic zone, a circuit, or even a section of a circuit. The simulator, which does not have the necessary know-how to create a concrete circuit, delegates this task to the circuit factory, whose existence and purpose are linked to this skill. The circuit factory (denoted by "CircuitFactory" in figure 8) is an entity that has great importance in this scenario. The dynamics of this scenario is started when the Co-simulation platform is started, and more precisely when the Simulator component is started. The *AIV_Server* which has received the *RTM* configuration file from the physical part of the Co-Simulator, transfers it to the simulator. The simulator has structurally and from its creation, a pending reference on a future *RTM*. The simulator instantiates the *RTM*, and instructs it to self-initialize, using the data from the configuration file, received from the physical part. The *RTM* is now a queryable object in the simulator environment. The latter then asks the *RTM* to provide the reference of the circuit factory to be used, by sending a request to this effect to the *RTM*. As soon as the response from the *RTM* is received, the simulator requests and obtains from the circuit factory the reference of the concrete circuit to be used. It is important to note that the simulator environment is now well configured in terms of the circuit and circuit factory, without any specific simulator knowledge of these new elements. The simulator can now display the layout of the circuit in its graphics space. This is possible without specific knowledge of the simulator, because the know-how necessary for this layout is hosted by the circuit itself, and not by the simulator. The sequence diagram in figure 8 gives a good overview of all the objects involved, as well as the dynamics of this scenario.

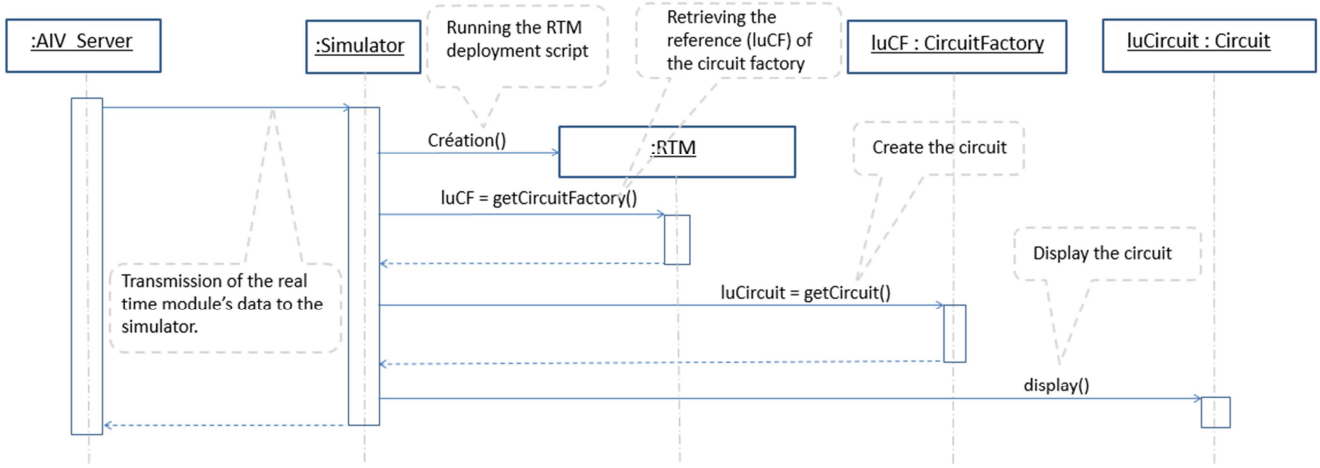

Figure 8. *Creation and display of the circuit by the simulator.*

3.6. Scenario for Starting a Physical AIV

We are interested here in the scenario of the entry of a real AIV into the physical circulation zone of the Co-simulator. This is to show how an action performed in the physical part of the Co-simulation platform, affects the digital part. The first significant event in this scenario is the arrival of the AIV on the traffic area. This arrival must take shape in the digital part of the platform. The *AIV_Server* which is the interface of the simulator, must be notified of this event. This is done via the radio interface that both the *AIV_Server* and the AIV itself have. It should be noted that each machine which hosts an agent capable of communicating remotely, also hosts locally, the proxy of the remote receiving agent. Thus, any outgoing communication to a remote agent is delivered to the proxy, and any incoming message from a remote agent is delivered locally by the latter's proxy. The appearance of the AIV in the physical circulation zone is thus notified to the *AIV_Server* by radio. In turn the *AIV_Server* transmits the event to the simulator. In the case where it is the first appearance of the AIV considered, the simulator understands that it is necessary to assign a visible mark to the AIV, and to position it in the right place in the graphic space of the simulator. This corresponds to the update of the position of the AIV on the circuit in the circulation zone. In the scenario presented here, a certain number of components participate which have been presented previously, such as the AIV, the *AIV_Server*, as well as the circuit. We see in the dynamics of the scenario, that the first outgoing message, from the AIV to the *AIV_Server*, is the publication of the current position of the AIV. As soon as this position message is received by the *AIV_Server*, the latter commands by a request to the simulator, the updating of the position of the AIV, by moving the associated marker in the graphic space. It should be noted that the update carried out by the simulator concerning the AIV concerns both the graphic display and the data of the AIV, as an object. At regular time intervals according to a timer armed on the AIV, the latter sends its position to the *AIV_Server*. Upon receipt of this information by the simulator, the latter anticipates the movement of the AIV on the circuit, by estimating the next position of the AIV. It should be noted that the memorization of the route data on the circuit, such as for example the distance traveled on the current section, are stored in the AIV, while the know-how necessary for the calculation linked to the geometry and the topology of the circuit, is hosted in the circuit itself, as well as for the circuit sections. The sequence diagram in figure 9 visually summarizes the start-up scenario of an AIV. The details of the algorithm for estimating the future position of the AIV are given in a later section of this article.

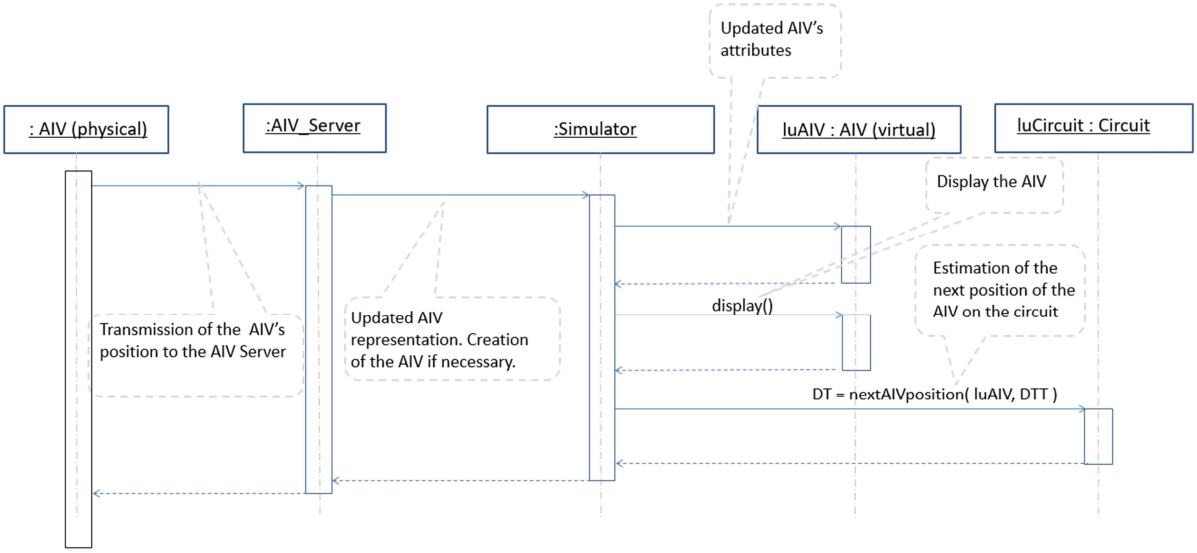

Figure 9. *Scenario for starting a physical AIV.*

4. The AIV Position Estimation Method

The question of the localization of an autonomous industrial vehicle (AIV), represents one of the key aspects of this article. Given that we are in a dynamic of cooperation between a physical part and a digital part of a Co-simulation platform, the need to anticipate in the digital part, the kinetics of an AIV, is of a great utility, especially for position estimation algorithms. Indeed, different use cases of our Co-Simulation platform require a position estimation. Some of these uses are related to the basic operation of the platform, some others are related to applications in other contexts. It seemed useful to us to show our approach in this part, with a light on the conceptual models, the mathematical approach, as well as certain interesting aspects of implementation.

4.1. The Conceptual Model of the Platform

The Co-Simulation platform includes, among other things, a simulation unit capable of duplicating in its virtual environment, the movements of different AIVs which evolve in their traffic zone. By adopting a conceptual approach to this system, the natural approach consists in identifying a set of entities, each representing a concrete object considered sufficiently relevant to be integrated into the simulation model. The key elements that make up the overall static model of the simulation system include the traffic area, the autonomous vehicle components involved in estimating their positions, the beacons, the circuit referred to as *CircuitLacet4* as shown in the figure 6, as well as section. The class diagram depicted in figure 10 presents the complete static model of the simulation system.

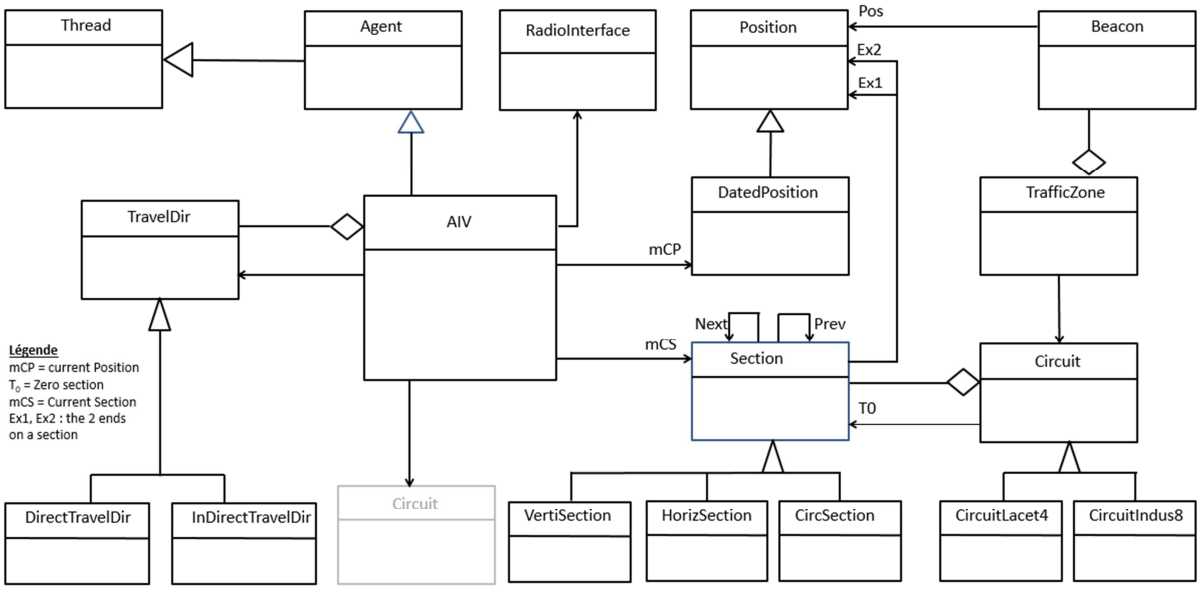

Figure 10. *Static model of the global simulation system.*

4.2. The Mathematical Position Estimation Model

Our algorithmic approach also makes it possible to determine the next position of the AIV on the circuit based on its current position. In a simplified way, we have defined the current position of the AIV as being P_n and the next position as being P_{n+1}. Our approach was to establish an abstract Section model where updating the current position would only be stated in principle, without providing specific details about the concrete implementation. Then, we built a specific model of the circuit Section based on the previously mentioned abstract model. This concrete section model was used to perform the actual calculation of P_{n+1}. Therefore, we have formally established three types of concrete sections with their respective calculation logics: a section in the form of an arc of a circle, a horizontal section and a vertical section. We will now give the elements of mathematical modeling in specific cases. We will come back to structural modeling in detail later.

For a circular arc section with center C and radius R, and for a time step Δt, the update of the position of the AIV is given by the expression (1).

$$\begin{cases} x_{n+1} = x_c + R\cos\left(arctg\left(\dfrac{y_n - y_c}{x_n - x_c} \right) + \dfrac{v}{R}\Delta t \right) \\ y_{n+1} = y_c + R\sin\left(arctg\left(\dfrac{y_n - y_c}{x_n - x_c} \right) + \dfrac{v}{R}\Delta t \right) \end{cases} \quad (1)$$

If we consider a horizontal section, the coordinates of the next position of the AIV knowing the current position are given by equation (2).

$$\begin{cases} x_{n+1} = x_n + v\Delta t \\ y_{n+1} = y_n \end{cases} \quad (2)$$

In the case of a vertical section, the coordinates of P_{n+1} knowing the coordinates of P_n are given by equation (3).

$$\begin{cases} x_{n+1} = x_n \\ y_{n+1} = y_n + v\Delta t \end{cases} \quad (3)$$

It is obvious that adding new types of edges with specific topologies does not represent any problem, and the generic calculation of the position will keep the same principle.

4.3. Temporal Sequencing of the Estimation of P_{n+1} Given P_n

We are interested here in the sequencing of actions in the calculation of the new position P_{n+1} of an AIV on the circuit, knowing its previous position P_n, and the distance to be traveled *DTT*. We assume that the travel direction is known and the corresponding attribute suitably filled in within the AIV. This calculation is performed when the AIV receives the "move" request, accompanied by the AIV reference, and the *DTT* parameter. The AIV can answer to the sender of the *move* request using the *move()* function. This function can be invoked explicitly (for example to perform tests), but it can also be called automatically, if the *move()* function has been defined as a "callback" to respond to the occurrence of an event. The event can in particular be a timer expired. We can consider without loss of generality that this event occurs at a random instant in the evolution of the AIV on the circuit. At this moment, the representation of the AIV in the digital environment of the simulator is characterized by a set of parameters linked, on the one hand, to its location (in particular its current position on the circuit, or the reference of the circuit section on which the AIV is currently located), on the other hand to variables linked to the management of changes in direction of travel, in particular the distance traveled by the AIV on the current section of the circuit (*mDTCS*). The estimation of the next position of the AIV on the circuit knowing the previous position requires access to certain objects in the environment of the simulator. Among these is the AIV itself, but there is also the current section of the circuit, as well as the next section and the previous one. From the dynamic point of view, the reception by the AIV of the "move" request immediately leads to the calculation of its new position given the distance to travel (*DTT*). One can easily observe in the digital environment of the simulator and the AIV, that the most competent entity to calculate a position at a known linear distance from a reference position, on a circuit formed by sections, is the section itself. In fact, we know very well that under these conditions, the position sought depends on the local topology, which is itself data specific to the section. The AIV which has the reference of its current section, sends the *nextAIVposition* request to this section. This request is accompanied by the reference of the AIV, which will allow the current section to access the AIV in order to update its position attribute. In return for the *nextAIVposition()* function coded to answer to the eponymous message in the Section class, the AIV as sender of the message receives the distance actually traveled (*DT*) on the current section (*CS*). This *DT* should be, in most cases, equal to the distance to travel (*DTT*). However, it may happen that this distance traveled is strictly less than *DTT*. This then means that the distance remaining to be traveled on the current section by the AIV is less than the distance to be traveled requested. This observation means that the AIV has almost reached the end of the current section corresponding to the position *Ex2*, and that consequently, the remaining distance must be traveled on the following section, provided of course that the direction of travel of the AIV on the circuit has not changed. In the event that the AIV observes a distance traveled strictly less than the distance to be traveled (*DT*<*DTT*), it would first be necessary to designate the following section as the current section of the AIV, reset the distance traveled by the AIV on the current section (*mDTCS*) to zero, update the distance to travel (*DTT*), so that it coincides with the rest of the distance to go, but carried over to the next section, which would amount to applying the equation *DTT* = *DTT* - *DT*. Secondly, the AIV would have to relaunch its *nextAIVposition* request, accompanied by the new *DTT* value. In return for this second call to the

nextAIVposition() function, the distance traveled and the distance to be traveled should coincide. At this stage, the AIV can close its response to the *move* request, by updating the distance traveled on the current section, by incrementing it by the distance actually traveled. This amounts to applying the equation *mDTCS = mDTCS + DT*. It is important to note that the new position P_{n+1} of the AIV will have been obtained in the *nextAIVposition()* function, by updating the *mCP* (Current Position) attribute of the AIV. The sequence diagram in figure 11 graphically shows the main sequential steps in the calculation of P_{n+1} knowing P_n.

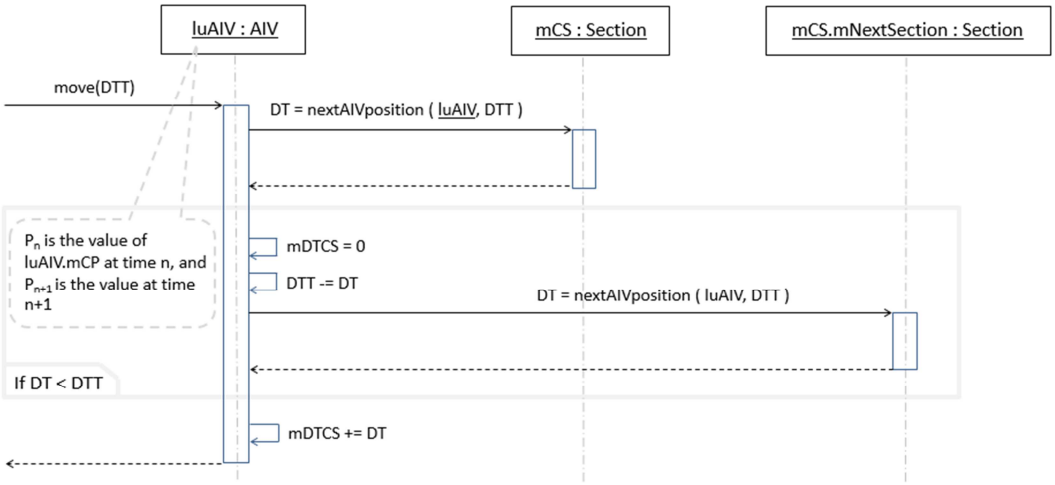

Figure 11. *AIV position update sequencing.*

4.4. Reification of the Travel Direction

We show in this part, the interest of reifying the direction of travel of the AIV, in a queryable object. The idea of the direct or indirect direction of travel of an AIV on a circuit located in a traffic zone seems simple, and it is. However, its impact is significant on various aspects relating to the management of the movement of the AIV on the circuit. Among the aspects impacted, we can mention the changes of sections, the identification of the ends of these same sections, or the correct calculation of the distance traveled on the current section, or finally the correct identification of the following and previous sections on the circuit, given the current section. Given all these impacts, and also the potential complexity that could be due to naive management driven by a case-by-case strategy, it seems logical to reify the direction of travel, so as to make it a queryable object, capable of responding adequately to the various requests, while avoiding generating ambiguous situations. We define an abstract model of the travel direction called *TravelDir*. The AIV has a reference to a particular *TravelDir*, seen by the AIV as the current *TravelDir*. The abstract *TravelDir* is also an interface that exposes different functions that allow calling entities to take advantage of the polymorphic character of the *TravelDir* model. One of these functions makes it possible to return to the calling entity (an AIV for example), the reference of the next section, the determination of which depends on the direction of travel.

Another of these functions provides the caller with the position of the start of the section on which the AIV is currently located. This interface also exposes functions whose role is to react to events such as *OnDirectMoveReq* or *OnInDirectMoveReq*. With two concrete models called *DirectTravelDir* and *InDirectTravelDir*, heirs of the abstract model *TravelDir*, we redefine the functions of the interface, so that these inheriting models can respond to the corresponding requests, in accordance with the context and the "travel direction". Note that the change management dynamic of the concrete *TravelDir* associated with the AIV is based on the state design pattern. The reader interested in details on this design pattern can usefully consult reference [36]. The class diagram in figure 12 gives a good overview of the model for managing the needs of the AIV, linked to the direction of travel on the circuit, according to the state design pattern.

When creating an instance object of the AIV model in the simulator environment, this object is by default in the *St_DirectTravelDir* state. In this state, two events of interest may occur, namely *Ev_OnDirectMoveReq* and *Ev_OnDInirectMoveReq*. If the *Ev_OnDirectMoveReq* event occurs, the state remains unchanged, and the AIV position update action is performed once. If it is the *Ev_OnInDirectMoveReq* event that occurs, then the action of updating the *mDTCS* variable of the AIV is performed immediately before any change of state. The new state associated with the AIV becomes *St_DirectTravelDir*. In this new state, the position update of the AIV will be performed once. The AIV keeps this state if the *Ev_OnDInirectMoveReq* event occurs. The update of the AIV's position is then performed once. If, on the contrary, the *Ev_OnDirectMoveReq* event occurs, then there is a change of state with a return to the *St_DirectTravelDir* state, and the *mDTCS* variable is updated. Once in the new state, the AIV position update is performed. All of the state changes and actions related to these changes are presented in the state diagram of figure 13.

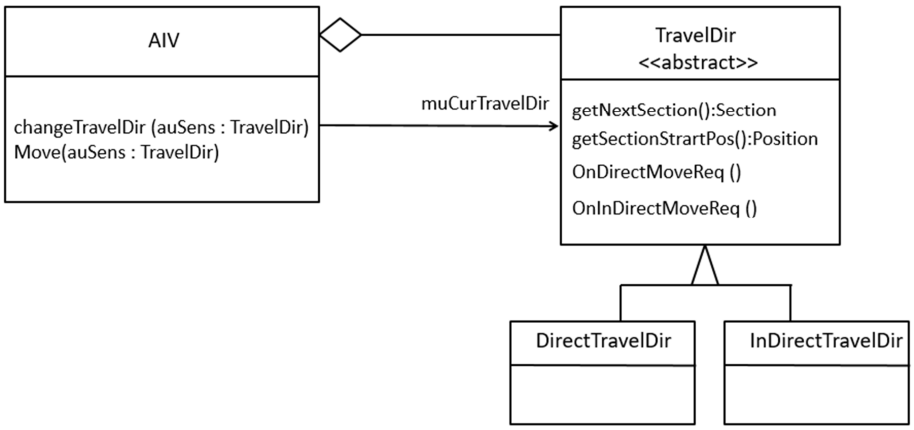

Figure 12. *Static model of travel direction reification.*

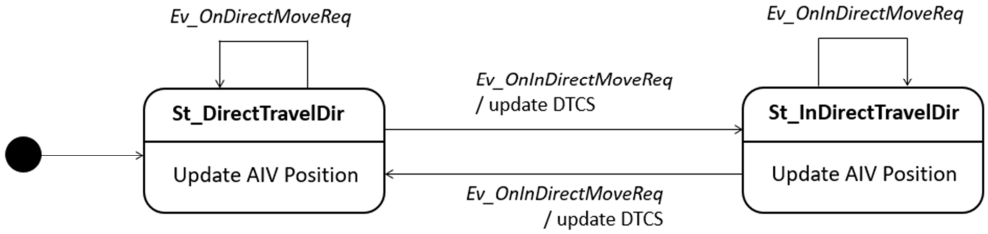

Figure 13. *AIV kinematics state diagram.*

4.5. The AIV Position Update Algorithm and Section Switching Management

The calculation of the AIV's next position in the circuit is based on a two-step strategy. The applicant for this calculation will in most cases be the AIV itself. As we mentioned in part 4.5, the AIV calls the *nextAIVposition()* method, and receives in return the distance actually traveled(*DT*). We will now detail the operating mechanism of this method. It is clear that for topological reasons, this method can logically only be hosted in the object model that describes a Section. Therefore, the *nextAIVposition()* method will be invoked to act on an instance object of the Section model. However, the actual calculation, for a real Section, can only be done in a concrete Section model, such as for example the circular arc Section model, vertical or horizontal. As a result, the Section object model, like the *TravelDir* model, satisfies the "abstract" stereotype. It is important to note that before deploying the calculation of the next position of the AIV, we check the precondition that we will call "Section conservation condition". This precondition consists in ensuring that on the current section, there remains enought room for the requested piece of path, of length *DTT*, to be deployed there. If the "section conservation condition" is not verified, then we immediately decide to go to the next Section. This change of section is accompanied by the update of variables defining the environment of the AIV, as well as its location. If, on the contrary, the "conservation condition section" is verified, then the *nextAIVposition()* method is called again, this time in its redefined version for the concrete Section inheriting from the abstract Section, because we now know that the remaining space on the current Section is large enough to accommodate the next position of the AIV. With the circular arc section model for example, the coordinates of the next position of the AIV will be calculated analytically according to the equation system (1). With the horizontal section model, the update of this position is carried out thanks to equations (2). Equation system (3) is used to update the position of the AIV for a vertical section model. It is therefore clear that each specific model of Section, by having its own redefinition of the *nextAIVposition()* method, can respond for its specific case, to the calculation request which remains generic. On figure 14 which shows the "flowchart" of this calculation, we can clearly distinguish the two calls to the *nextAIVposition()* method. As we can see, this algorithmic scheme is structured in three phases identified by (a), (b), and (c). A good way to summarize both this scheme and this algorithmic processing would be to emphasize the following two things: First, in the case where the AIV remains on the same section, we perform phases (a)+(c). Secondly, if on the contrary the AIV changes section, then it is the phases (a)+(b)+(c) which are carried out.

On figure 15, we can clearly see the previous, current and following sections. The configuration of interest here is that of a change of section, and this can be seen in this figure, by the fact that the distance to be covered *DTT*, is strictly greater than the distance remaining to be covered on the current section. Figure 15 also illustrates the fact that the next position calculation intelligence of the AIV also makes it possible to manage the handover of the autonomous industrial vehicle from one section to the next.

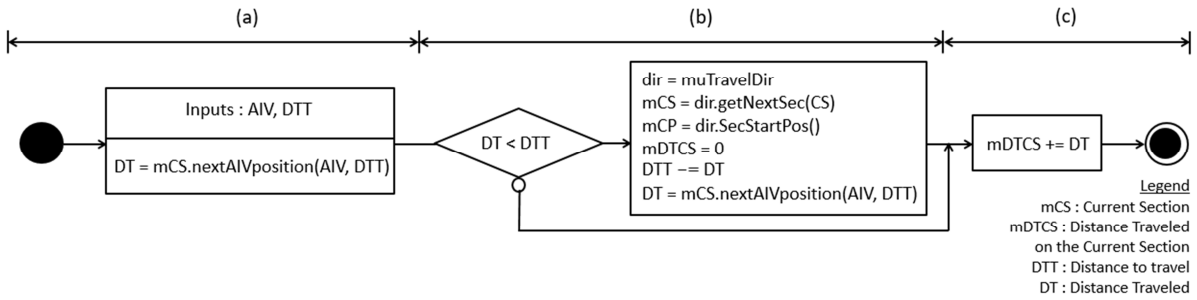

Figure 14. *Algorithm for updating the position of an AIV.*

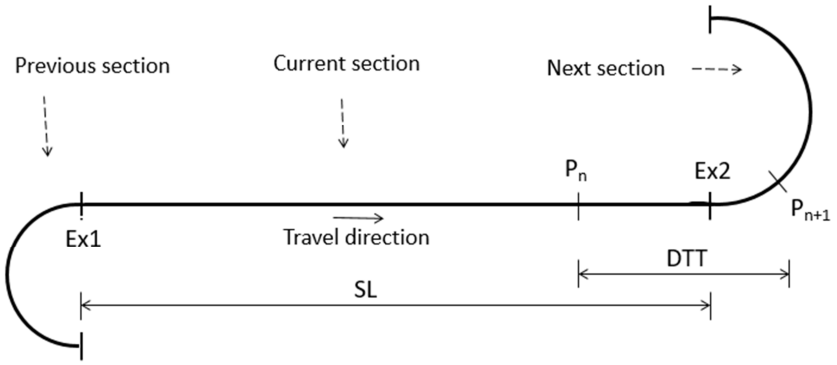

Figure 15. *Current and next positions of the AIV in the case of a section switch.*

5. Some Application Cases

We present in this part, some examples of applications that we considered interesting, as well for needs of research, as for industrial needs. We have chosen to present two families of applications. The first family concerns the metrology of position estimation algorithms. Based on judiciously chosen metrics, we show how our Co-simulation platform can become a tool that allows to qualify or compare two position estimation algorithms. As a metric, we have for example the localization precision given by the algorithm. Accuracy is then, as a metric, an element of the metrology of the algorithms taken into account, allowing them to be characterized and compared.

5.1. Metrology of Position Estimation Algorithms

Realizing the metrology of a position estimation algorithm consists in measuring, thanks to a set of metrics, the operation as well as the performance of this algorithm. This metrology can also be broken down into a comparative dynamic, which allows us to classify a set of algorithms in relation to each other. In this part dealing with algorithm metrology, we will present two different situations. The first situation concerns the qualification of an algorithm, while the second situation concerns the comparison of two algorithms. The scenarios that we present relate to an example of application of the Co-simulation platform, in a research context, for which our laboratory is particularly concerned.

5.1.1. Qualification of a Position Estimation Algorithm

The qualification of a position estimation algorithm globally follows a scenario in which at each time step (this step is defined within the simulator), the algorithm gives an estimate of the next position of the AIV. This estimated position is then compared with the actual position actually occupied by the AIV on the circuit in the traffic area. The difference between the two positions then corresponds to the instantaneous estimation error of the algorithm. The metrics which seem useful to us to qualify the estimation algorithm are three in number: 1) the average of the estimation error, 2) the variance of the estimation error, 3) the precision of the algorithm. We define the precision of the estimation algorithm, by the number of iterations for which the estimation error is less than a minimum error, compared to the total number of iterations. The qualification scenario of a position estimation algorithm involves various objects present in the environment of the simulator, which is itself the pivot object of the scenario. In other words, the goal here is to qualify an estimation strategy defined by an algorithm, itself reified into an object. This "Algorithm" object is interrogated in order to predict the next position of an AIV, which is also an important object in this scenario. A fourth object of importance in this scenario is the Global Vision System (*GVS*). This object will be interrogated each time it is necessary, to give the position actually occupied by the AIV at the current instant on the circuit. In concrete terms, the *GVS* is a vision system equipped with a camera capable of capturing the image of the entire traffic area, and of extracting the position of an AIV present on the circuit. From an operational point of view, the simulator object sends to the AIV a request called *nextEstimatedPosition*, accompanied by an argument designating the algorithm which will define the estimation strategy. In return for this request, the simulator waits for the

value of the estimated position in the variable *PE*. The AIV in turn calls on the "Algorithm" instance, which is logical, given that the know-how allowing the position estimation is hosted by the Algorithm object model. Once the future position of the AIV has been estimated, the simulator must command the AIV to move towards this next position. This is done by sending the "move" request to the AIV. As soon as the AIV has reached the target position, the simulator asks the *GVS* for the actual position of the AIV, using the *actualAIVposition* request. The estimated position (P_E) and the actual position (P_A) are then recorded for later use. After this recording, the overall processing loops back to the estimation of the next P_E position of the AIV. The second phase of this scenario consists in carrying out an analysis of the recorded data. First, the precision of the algorithm is calculated, followed in a second step by the calculation of the statistics of the deviations in terms of mean and variance. The global qualification scenario of a position estimation algorithm is summarized in the sequence diagram of figure 16.

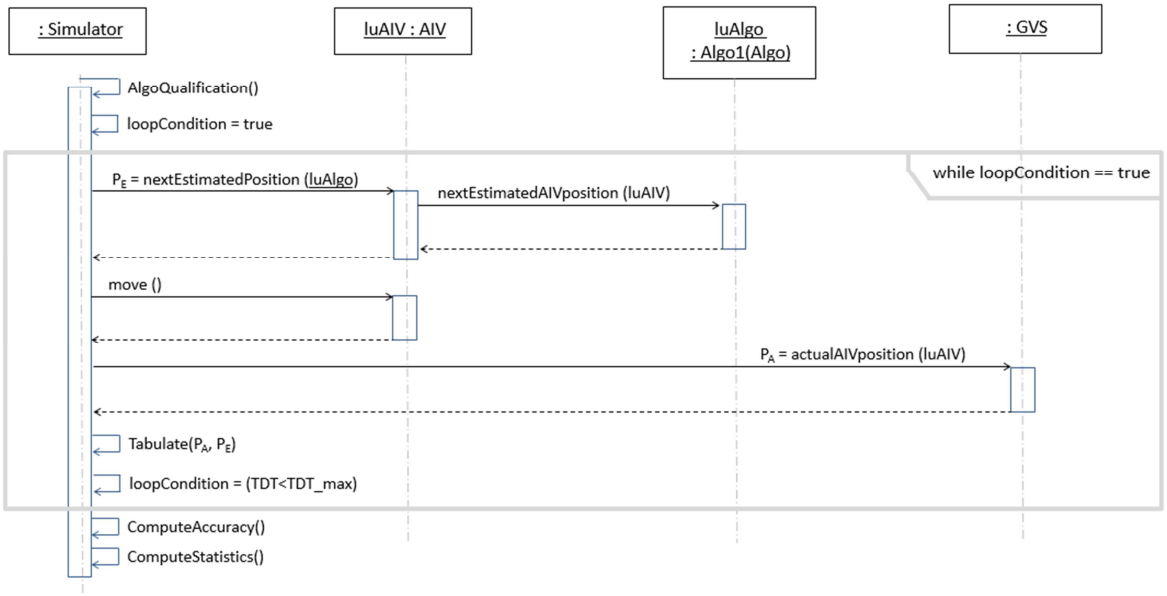

Figure 16. *Example scenario for qualifying a position estimation algorithm.*

5.1.2. Comparison of Position Estimation Algorithms

The comparison of two position estimation algorithms is, from our point of view, one of the services that our Co-Simulation platform can provide. The qualification scenario of a position estimation algorithm is in fact an important building block, which allows us to easily implement this comparison in an easy way. Figure 17 shows the comparison scenario of two different position estimation algorithms, for the same AIV.

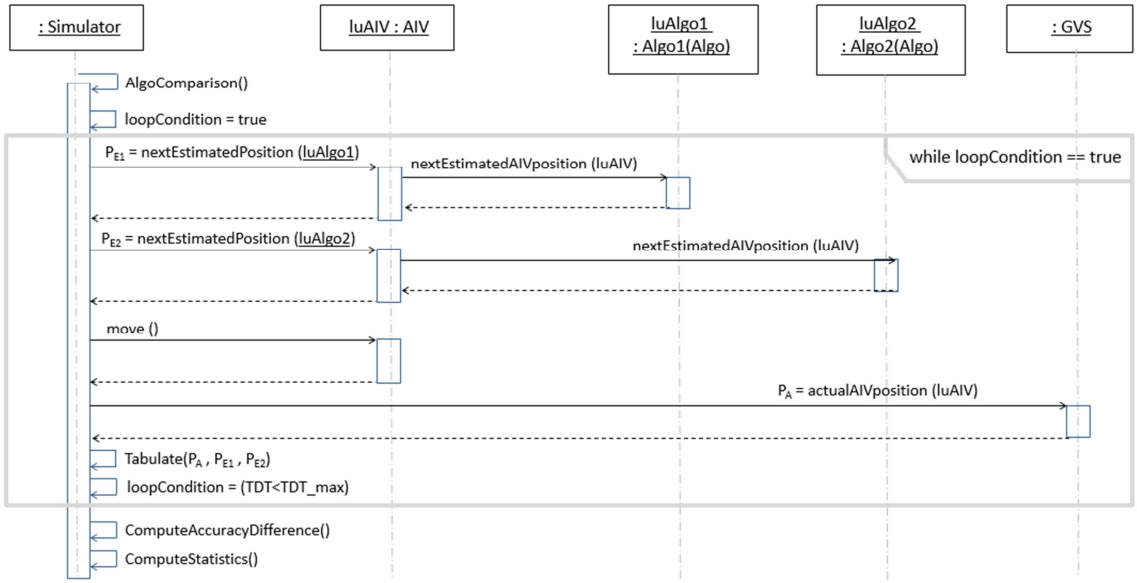

Figure 17. *Example scenario for comparing position estimation algorithms.*

The observation of figure 17 shows that *Simulator*, *AIV* and *GVS*, already present in the scenario of figure 16, are included in the present scenario. Note also the presence of two new objects called *luAlgo1* and *luAlgo2*, designating the two algorithms to be compared. It is also important to note that *luAlgo1* and *luAlgo2* are respectively instances of the two models *Algorithm1* and *Algorithm2*, themselves respective heirs of the base model *Algorithm*. According to the dynamics of this scenario, we can see in figure 17, that the Simulator instance requires and obtains from the AIV, the two estimates (P_{E1} and P_{E2}) of its position on the circuit, according to the strategies respectively controlled by *luAlgo1* and *luAlgo2*. After the AIV has reached its next position on command from the Simulator, the *GVS* provides the simulator with the *PA* Position of the AIV, which here represents the ground truth. Similar to the qualification scenario, the triplet (P_A, P_{E1}, P_{E2}) is recorded by the simulator. This triplet corresponds to the values of the actual and estimated positions according to the two algorithms under test. The processing loops back to the estimated positions, with the maximum limit of the distance traveled reached (*TDT_max*) as stopping criterion. This scenario ends with: on the one hand the calculation of the accuracy difference between the two algorithms, and on the other hand the calculation of the statistics of the respective differences between the positions estimated by the two algorithms, and the real position of the AIV.

5.2. Characterization of Electric Batteries

The Co-Simulation platform that we describe in this article can, from our point of view, play a very useful role in the characterization of electric batteries. Manufacturers of electric batteries could, in our opinion, find real interest in using this platform to characterize and therefore validate their products. The interest that we see in it essentially lies in the automation of tests, as well as their dematerialized management. It is generally a question of defining test scenarios to be implemented in the physical part of the platform, and controlled from the digital part. In addition, this implementation of the tests on the Co-Simulation platform makes it possible to overcome the constraint linked to the duration of the scenarios (these can last several tens of hours). The first characterization scheme that we present is the sampled measurement of the autonomy of electric batteries. This scheme leads to an autonomy value, estimated on the battery subject of the experiment. The second characterization scheme concerns the statistical measurement of the autonomy of a type of electric battery, in which the measurement of the autonomy time is carried out on several batteries at the same time, which makes it possible to arrive at a measurement of the autonomy averaged over all the batteries tested.

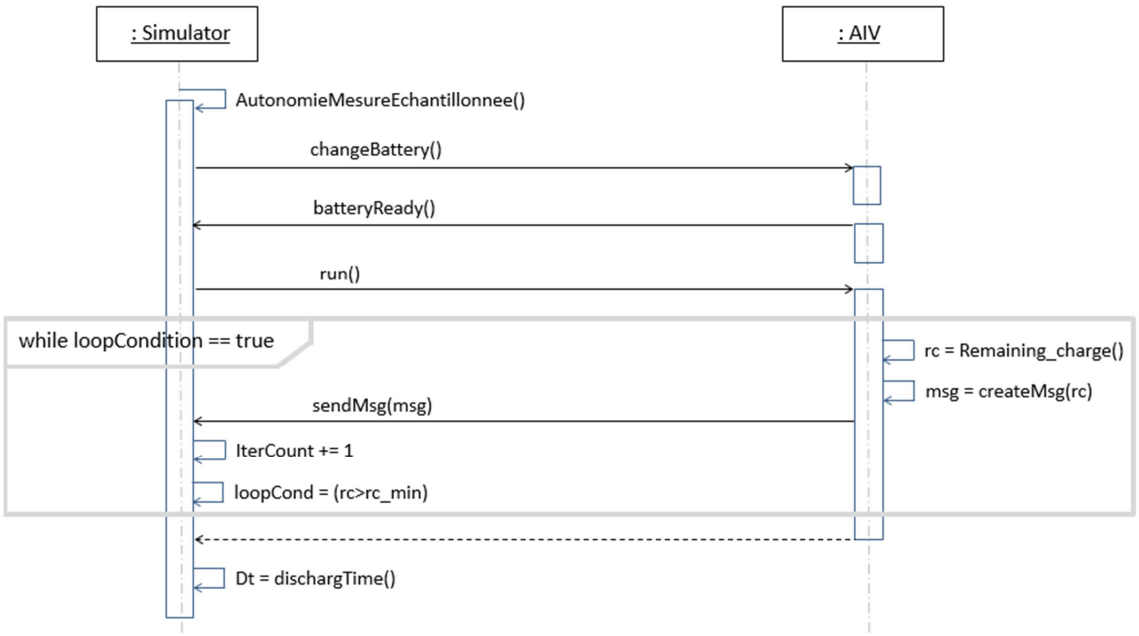

Figure 18. *Example scenario of sampled measurement of electric battery autonomy.*

5.2.1. Sampled Measurement of the Autonomy of Electric Batteries

The autonomy measurement experiment proposed here is made on a single battery. The result is therefore a measure of autonomy for a single individual. The concept of sampled measurement is in fact linked to this uniqueness of the individual tested. The *Simulator* and the *AIV* are the two main agents involved in this scenario. The dynamic scheme is organized in two phases. The first phase concerns the initialization of the process. In this initialization phase, the simulator ensures that the battery to be tested is indeed in its fully charged state. To do this, it sends the *changeBattery* request to the AIV. Note that the AIV which receives this request is located in the circulation zone of the physical part of the Co-Simulation platform. The battery change process may involve human intervention, but in all cases we assume that at

some point a fully charged battery is in place, and the AIV is ready. The AIV then sends the *batteryReady* message to the Simulator, which can then launch the autonomy measurement scenario, by sending the "run" request to the AIV. This request is used to set the AIV in motion in a background task. In the second phase of this scenario, the AIV begins by acquiring the remaining battery charge (*rc*). This remaining charge value is included in a formatted message, intended for the simulator. This message is sent by the AIV to the simulator via the *sendMsg* request. This second phase is in fact a looped processing, which brings the flow of control back to the instruction for determining the remaining charge. This looping is carried out after the update of the stopping criterion of the processing loop. This shutdown criterion, based on the remaining charge *rc* of the battery, is given by the Boolean equation: *loopCond* = (*rc*>*rc_min)*, where *rc_min* designates the value of the floor charge whose crossing triggers the *AIV* to stop. At the exit of the program loop, the counter incremented at each iteration, makes it possible to deduce the discharge time which has elapsed between the battery state at full charge, and the battery state at the floor charge *rc_min*. It is also possible to mark a pause of a calibrated duration within the program loop, according to the discharge profile of the battery. The scenario for this sampled measurement is shown in the sequence diagram in figure 18.

We can finally note that it is possible to perform this sampled measurement of autonomy in a time-averaged version, simply by renewing the scenario presented, *N* times with the same battery.

5.2.2. Statistical Measurement of the Autonomy of Electric Batteries

We present here a scenario of statistical measurement of the autonomy of a family of electric batteries. By this concept of "statistical measurement", we mean the fact that several individuals (battery samples) are taken into account at the same time in the experiment. The objective is to take advantage of the fleet of AIVs available, to carry out a measurement extended to a set of batteries, and thus to arrive at a measurement of autonomy, averaged over all the individuals. The scenario presented here uses the overall structure of the scenario in figure 18, except that instead of one AIV, we have *N*. The initialization phase with a possible change of battery involves the simulator, this time with each of the *N* AIVs. The initialization phase is considered complete when the last AIV has sent the *batteryReady* message to the simulator. The simulator commands the start-up of the fleet of AIVs, by sending the message "run" to each AIV. It is important to note that by assumption, the different AIVs involved in this scenario are supposed never to cross in the traffic area. They are indeed all placed on the same circuit, and all have the same direction of travel. At regular intervals, the simulator receives from each AIV a message including the remaining charge of said AIV, as well as the date. Each AIV in the fleet, supervised by the simulator, is stopped as soon as its remaining charge drops below the floor charge, defined at the same *cr_min* value for all AIVs. After stopping the last AIV, the simulator uses the data received. From the autonomy times obtained for each AIV, the simulator can deduce the autonomy time (*AT*), averaged over all the batteries in the fleet. This average time will be obtained by the simulator, thanks to the formula 4.

$$AT = \frac{1}{N}\sum_{i=1}^{N} T_i \tag{4}$$

where T_i designates the autonomy of the AIV number *i*.

The scenario for this statistical measure is given by the sequence diagram in figure 19.

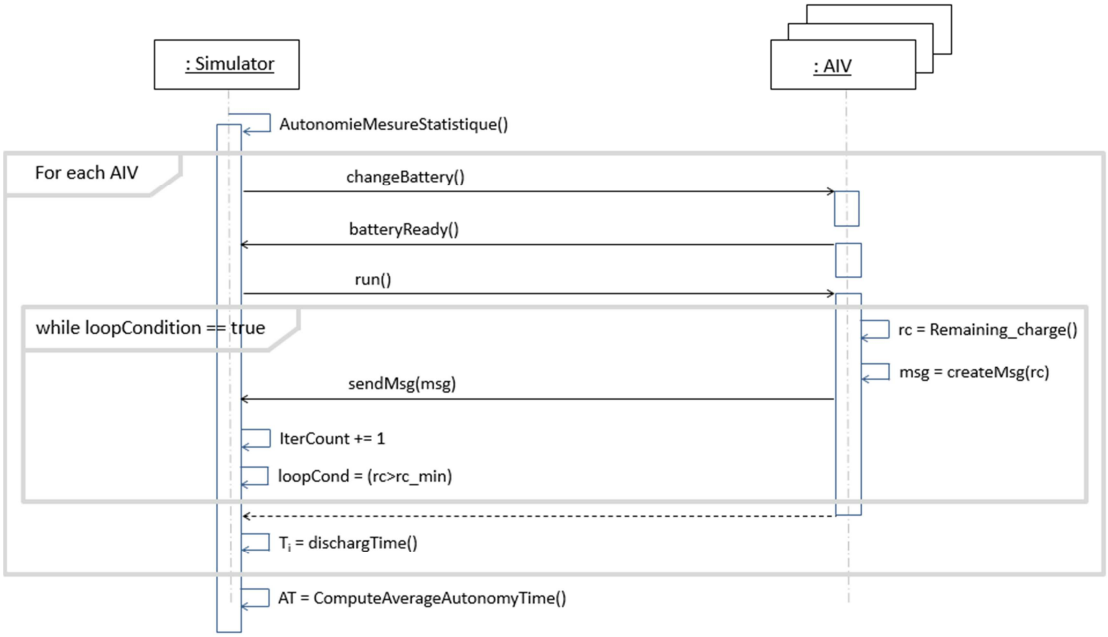

Figure 19. *Example scenario of statistical measurement of autonomy.*

6. Conclusion and Future Work

In this article, we have presented a review of the literature which highlights a very contrasting reality. Indeed, on the one hand, a lot of research is documented on autonomous vehicles, but on the other hand, there is little work regarding autonomous industrial vehicles (or mobile robots), given that the Attention has primarily focused on road-type autonomous vehicles. There are obvious similarities between these two uses of autonomous vehicles, including the need to simulate the vehicles associated with their traffic contexts before designing and implementing them in real environments. In the industrial context, the use of simulation makes it possible to take into account the constraints and requirements issued by manufacturers and future users of autonomous vehicles. Therefore, the development of simulation platforms plays a crucial role in improving the autonomy of AIVs. As we have seen, the platforms identified in the literature have often been exclusively either virtual or physical. Our approach, as presented in this article, is based on experiments with Co-simulation platforms combining physical and virtual approaches.

Both on a physical and virtual level, determining the correct location of vehicles is essential. We have thus proposed an approach for estimating the position of AIVs, based on the combination of a matrix markup, with a technique for managing the change of section on the circuit. The matrix marking of the traffic area, materialized by a grid such as we have seen in figures 6 and 7, constitutes a means of defining fixed and reliable references of positions on the circuit. This matrix beaconing, the graduation of which can be parameterized, will make it possible in particular to test the influence of its density on the efficiency of the position estimation.

We have emphasized the necessary scalability of our Co-simulation platform, scalability based on a generic basis of the digital part of the system. This principle of scalability allowed us to demonstrate the agnostic character of our Co-Simulation platform, with respect to the circulation zone. There are many possibilities for the evolution of the Co-simulation system, in particular that of being able to freely modify the physical platform, without relaxing the constraint of automatic adaptation imposed on the digital part. Ultimately, it is about being able to deploy the simulation in any industrial environment. An AIV manufacturer could thus provide the simulation to its customers and work with them on deployment scenarios specific to their worksites. We have also described some examples of interesting applications from our point of view, in order to show the possibilities of valuing our Co-simulation platform model, on the one hand in the field of metrology of algorithms for estimating position, on the other hand in that of the characterization of electric batteries.

It should be noted that only part of the static and dynamic model presented in this article has actually been implemented to date. Indeed, if our Co-simulation platform is now able to control the circulation zone including the *CircuitLace4*, the functionality of switching to another type of circulation zone remains to be carried out, this is part of our planned. The various scenarios described within the framework of the examples of proposed applications have been described and formalized in various diagrams, as we have seen in part 5 of this article. Our next work will consist in particular in physically realizing and testing these scenarios.

References

[1] H. Lasi, P. Fettke, H. G. Kemper, T. Feld, and M. Hoffmann, (2014). Industry 4.0. Business & information systems engineering, 6 (4): 239-242.

[2] A. C. Pereira and F. Romero, (2017). A review of the meanings and the implications of the Industry 4.0 concept. Procedia Manufacturing, 13: 1206-1214.

[3] Y. Wiseman, (2021). Autonomous vehicles. In Encyclopedia of Information Science and Technology, Fifth Edition, IGI Global, pp. 1-11.

[4] H. Andreasson, A. Bouguerra, M. Cirillo, D. N. Dimitrov, D. Driankov, L. Karlsson, A. J. Lilienthal, F. Pecora, J. P. Saarinen, A. Sherikov, and T. Stoyanov, (2015). Autonomous Transport Vehicles: Where We Are and What Is Missing. IEEE Robotics & Automation Magazine, 22 (1): 64-75.

[5] H. Khayyam, B. Javadi, M. Jalili, & R. N. Jazar, (2020). Artificial Intelligence and Internet of Things for Autonomous Vehicles. In Nonlinear Approaches in Engineering Applications, Springer, Cham, pp. 39-68.

[6] R. S. Peres, X. Jia, J. Lee, K. Sun, A. W. Colombo, and J. Barata, (2020). Industrial artificial intelligence in industry 4.0-systematic review, challenges and outlook. IEEE Access, 8, 220121-220139.

[7] C. Medrano-Berumen and M. I. Akbaş, (2020). Validation of decision-making in artificial intelligence-based autonomous vehicles. Journal of Information and Telecommunication, DOI: 10.1080/24751839.2020.1824154.

[8] R. Bostelman and E. Messina, (2016). Towards development of an automated guided vehicle intelligence level performance standard. In Autonomous Industrial Vehicles: From the Laboratory to the Factory Floor, ed. R. Bostelman and E. Messina (West Conshohocken, PA: ASTM International 2016), pp. 1-22. https://doi.org/10.1520/STP159420150054

[9] Z. Tao, P. Bonnifait, V. Fremont, J. Ibanez-Guzman, (2013). Lane marking aided vehicle localization. 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013).

[10] J. H. Oh, D. Kim, and B. H. Lee, (2014). An Indoor Localization System for Mobile Robots Using an Active Infrared Positioning Sensor. Journal of Industrial and Intelligent Information Vol. 2, No. 1, March 2014.

[11] S. B. Cruz, T. E. Abrudan, Z. Xiao, N. Trigoni, and J. Barros, (2017). Neighbor-Aided Localization in Vehicular Networks. IEEE Transactions on Intelligent Transportation Systems, Vol. 18, No. 10, October 2017.

[12] A. Debski, W. Grajewski, W. Zaborowski, W. Turek, (2015). Open-source Localization Device for Indoor Mobile Robots. Procedia Computer Science, Volume 76, 2015, Pages 139-146.

[13] I. J. Cox, (1990). Blanche: Position estimation for an autonomous root vehicle. In Autonomous robot vehicles, Springer, New York, NY, pp. 221-228.

[14] F. Bounini, D. Gingras, V. Lapointe, and D. Gruyer, (2014). Real-time simulator of collaborative autonomous vehicles. In 2014 Int. Conf. on Advances in Computing, Communications and Informatics (ICACCI), pp. 723-729.

[15] B. Y. Ekren and S. Heragu, (2011). Simulation based performance analysis of an autonomous vehicle storage and retrieval system. Simulation Modelling Practice and Theory. 19 (7): 1640-1650.

[16] S. D. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. H. Eng, and M. H. Ang, (2017). Perception, planning, control, and coordination for autonomous vehicles. Machines, 5 (1): 6, https://doi.org/10.3390/machines5010006.

[17] M. B. Alatise and G. P. Hancke, (2020). A review on challenges of autonomous mobile robot sensor fusion methods. IEEE Access, 8: 39830-39846.

[18] N. A. K. Zghair, and A. S. Al-Araji, (2021). A one-decade survey of autonomous mobile robot systems. International Journal of Electrical and Computer Engineering, 11 (6): 4891.

[19] F. Rubio, F. Valero, and C. Lopis-Albert, (2019). A review of Mobile Robots: Concepts, Methods, Theoretical Framework, and Applications. International Journal of Advanced Robotic Systems, 16 (2): 1-22.

[20] H. Zhu, K. V. Yuen, L. Mihaylova, and H. Leung, (2017). Overview of environment perception for intelligent vehicles. IEEE Transactions on Intelligent Transportation Systems, 18 (10): 2584-2601.

[21] F. Rosique, P. J. Navarro, C. Fernández, and A. Padilla, (2019). A systematic review of perception system and simulators for autonomous vehicles research. Sensors, 19 (3): 648, https://doi.org/10.3390/s19030648.

[22] F. Arena and G. Pau, (2019). An overview of vehicular communications. Future Internet, 11 (2): 27, https://doi.org/10.3390/fi11020027.

[23] M. Y. Abualhoul, O. Shagdar, and F. Nashashibi, (2016). Visible Light inter-vehicle Communication for platooning of autonomous vehicles. In 2016 IEEE Intelligent Vehicles Symposium (IV), pp. 508-513.

[24] O. Grembek, A. Kurzhanskiy, A. Medury, P. Varaiya, and M. Yu, (2019). Making intersections safer with I2V communication. Transportation Research Part C: Emerging Technologies, 102: 396-410.

[25] W. Huang, K. Wang, Y. Lv, and F. Zhu, (2016). Autonomous vehicles testing methods review, in 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), pp. 163-168.

[26] M. O'Kelly, A. Sinha, H. Namkoong, R. Tedrake, J. C. Duchi, (2018). Scalable End-to-End Autonomous Vehicle Testing via Rare-event Simulation. In Advances in Neural Information Processing Systems 31 (NeurIPS 2018), arXiv preprint arXiv: 1811.00145.

[27] D. Kade, M. Wallmyr, T. Holstein, R. Lindell, H. Ürey, and O. Özcan, (2016). Low-Cost Mixed Reality Simulator for Industrial Vehicle Environment. In International Conference on Virtual, Augmented and Mixed Reality, Springer, Cham, pp. 597-608.

[28] S. Liu, L. Li, J. Tang, S. Wu, and J. L. Gaudiot, (2020). Creating autonomous vehicle systems. Morgan & Claypool Publishers.

[29] P. Jing, H. Hu, F. Zhan, Y. Chen, and Y. Shi, (2020). Agent-based simulation of autonomous vehicles: A systematic literature review. IEEE Access, 8: 79089-79103.

[30] K. Dresner and P. Stone, (2008). A multiagent approach to autonomous intersection management. Journal of artificial intelligence research, 31: 591-656.

[31] A.-J. Fougeres, (2013). A Modelling Approach Based on Fuzzy Agent. International Journal of Computer Science Issues, 9 (6): 19-28.

[32] A.-J. Fougeres and E. Ostrosi, (2013). Fuzzy agent-based approach for consensual design synthesis in product configuration. Integrated Computer-Aided Engineering, 20 (3): 259-274.

[33] C. Aynau, C. Bernay-Angeletti, R. Aufrere, L. Lequievre, C. Debain, and R. Chapuis, (2017). Real-time multisensor vehicle localization: A geographical information system based approach. IEEE Robotics & Automation Magazine, 24 (3): 65-74.

[34] M. de Ryck, M. Versteyhe, F. Debrouwere, (2020). Automated guided vehicle systems, state-of-theart control algorithms and techniques. Journal of Manufacturing Systems, 54: 152-173.

[35] H. S. Hasan, M. Hussein, S. M. Saad, and M. A. M. Dzahir, (2018). An overview of local positioning system: Technologies, techniques and applications. International Journal of Engineering & Technology, 7 (3.25): 1-5.

[36] E. Gamma, R. Helm, R. Johnson, and J. Vlissides (1994) Design Patterns – Catalogue des modèles réutilisables, Book, p 125, 357.