Sciences appliquées Industrie Administration Débouchés Intervenant Stock
Technologies Santé Mention informatique Opérationnel Optimisation Innovation
Organisation Connaissance Logistique Groupe Projet Stages Motivation Système ERP
Ingénieur Cadre Réseau Supply chain management Droit Finance Marketing Design
Changement Amélioration Intégration Evolution Responsable Logiciel Développement
Direction Communication Management Achat Qualité Interface Formation Droit
Sciences Appliquées Industrie Administration Débouchés Intervenant Stock
Technologies Santé Mention informatique Opérationnel Optimisation Innovation
Organisation Connaissances Logistique Groupe Projet Stages Motivation Système ERP
Ingénieur Cadre Réseau Supply chain management Droit Finance Marketing Design
Changement Amélioration Intégration Evolution Responsable Logiciel Développement
Direction Communication Management Achat Qualité Interface Formation Droit

Master Sciences et Technologies,

Mention Informatique, Parcours SIGLIS

Stage 2ème année

Anomalies Detection by self-observation of Situations within a Connected Autonomous Vehicle

Figure : Détection et communication entre véhicules autonomes connectés
https://commons.wikimedia.org/w/index.php?curid=44189986

Mars – Août 2015

Kristell Aguilar

Université de Pau et des Pays de l'Adour Collège STEE Composante Interne de Formation de la Côte Basque Master Sciences et Technologies Mention Informatique Parcours SIGLIS 1 Allée du Parc Montaury 64 600 ANGLET *Tél : 05 59 57 44 52*	**IRIT** Institut de Recherche en Informatique de Toulouse 118 route de narbonne 31062 TOULOUSE *Tél :* 05 61 55 60 00
Tuteur de stage : M. E. Exposito	Maître de Stage : Mme. Elsy Kaddoum

Stage

du 26 mars 2018 au 31 août 2018

Anomalies Detection by self-observation of Situations within a Connected Autonomous Vehicle

Promotion 2017-2018	Kristell Aguilar

M2 INFORMATIQOUE

PARCOURS SIGLIS

ANOMALIES DETECTION BY SELF-OBSERVATION OF SITUATIONS WITHIN A CONNECTED AUTONOMOUS VEHICLE

By

Kristell AGUILAR

Research Tutor: Elsy KADDOUM

University tutor: Ernesto EXPOSITO

August 2018

Index

Index of Figures

Index of Tables

Chapter 1. Introduction

In this chapter, an introduction to the project RECOVAC is made. Then, the problem and the objectives of the work of my internship are presented.

1.1 Context

The vehicle is one of the most used means of transport in the world and is subject to a variety of innovation works. Beyond the material innovation, the vehicle of the future must address societal and human concerns such as: the wasted time spent traveling, the limited driving capacity of some people, or the availability of travel modes and / or vehicles. Automation technologies substantially affect these concerns, bringing as benefits: increased passenger safety and satisfaction, increased traffic flow, mobility available to people without driving ability, and reduced congestion, accident and pollution problems.

Nowadays, there exist many researches related to the autonomous vehicles field, known as a vehicle able to perceive the environment where it evolves and navigates without human intervention. The National Highway Traffic Safety Administration (NHTSA) in the United States recently published a statement defining five levels (from level 0 to level 4) of automation, going from no automation, where the driver has the complete control of the vehicle, to full autonomy, where the vehicle operates independently [1]. According to this study, the RECOVAC (conditions for REtaking COntrol by self-observation of situations within a Connected Autonomous Vehicle) project focuses on autonomous vehicles of level 3, which describes an automation mode that only requires limited control of the driver:

"Level 3 - Limited Self-Driving Automation: Vehicles at this level of automation enable the driver to cede full control of all safety-critical functions under certain traffic or environmental conditions and in those conditions to rely heavily on the vehicle to monitor for changes in those conditions requiring transition back to driver control. ***The driver is expected to be available for occasional control, but with sufficiently comfortable transition time****. The vehicle is designed to ensure safe operation during the automated driving mode."*

Essentially, this automation mode stipulates that the human driver can act as a fail-safe mechanism and requires the driver to take control in case something goes wrong.

Within the design of the driving system, are defined situations where the vehicle is not able to manage or is not able to control. In the same way, this system must provide a context to evaluate the situation as quickly as possible, so that the vehicle becomes aware of the criticality of this situation and can react in the best possible way. For situations of interest, three cases of control recovery are established: i) human regains control when he wishes, ii) human takes back control when the driving context requires it, iii) vehicle regains control over the human when the driving context allows it.

1.1 Problem Description

are playing an important role in transport systems. Although there are other developed means of transport such as maritime and air transport, the degree of automation in cars is continuously increasing. There are many technological, regulatory, legal, ethical and societal obstacles that limit this new concept [2]. However, there are already vehicles with driver assistance technologies that contribute in increasing safety, reducing the severity of accidents, alerting drivers of potential problems, and even intervening automatically in situations that represent a risk, with the aim of improving efficiency, protection and comfort when driving, maintaining continuous interaction with the driver, the vehicle and the infrastructure [3].

In this work, we study the connected autonomous vehicles of level 3, vehicles for which the driver delegates the driving and acts as a supervisor. However, it may be necessary for the human to regain control. The main objective of the project is to develop a system allowing the safe and effective transition of two-way control between the human and the autonomous vehicle. For this, the work to realize may be divided into two parts. First, the driving system must by self-observation and in real time identify situations in which the vehicle will no longer be able to ensure driving. Secondly, it must provide a context for assessing the criticality of the situation as quickly as possible in order to anticipate and react to it as best as possible. This context consists of the environmental data captured by the vehicle as well as data calculated by the vehicle itself. It is necessary to insure a safe transition to the driver if needed.

My internship is a part of the RECOVAC project and seeks to develop a learning system able to identify normal situations where the vehicle drives well from abnormal ones. The developed system is based on self-adaptive learning systems.

1.2 Objectives

In this internship, I focused on the first part of the work (identify situations by self-observation and in real time in which the vehicle will no longer be able to ensure driving) and designed an evaluation system, called Situations PredictiON System in Vehicles Autonomous (reSPONSiVA), based on the adaptive multi-agent system approach, able to dynamically detect if the vehicle: i) is not able to react in the expected way, or ii) reacts in the expected way.

More specifically my work consists in the design of a synthetic generator of environments that enables the definition of a set of normal and abnormal situations, and in the development of a self-adaptive multi-agent system that allows dynamic learning of abnormal situations over normal situations, for the automatic detection of driving situations.

Chapter 2. State of Art

In this chapter, we present a brief summary of related or antecedent works in the field of autonomous vehicles and introduce the approach used to solve our problem.

2.1 Related Works

One of the largest area in the development of automobiles has to do with the conception of driver assistance systems, which help the driver in the processes of driving the vehicle, in a non-invasive way, as shown by the works [4, 5]. In this way, we are moving to one of the vanguard topics in the automotive area, the *autonomous vehicles (AV)* where vehicles are able to imitate the human capacities of management and control, to perceive the environment that surrounds them and move accordingly [6]. Among the different types of levels of automation, the level 3 is known as conditional automation, automatic and dynamic driving system, where the human driver responds only to a request for intervention, as explained in the work [7,8].
In particular, these systems must consider emotional, environmental and vehicle factors, since they greatly affect the performance of the driver and the vehicle. Thus, there is a need to detect and diagnose the emotional state of the driver, the environmental state where the driving process takes place and the internal variables of the vehicle that influence their actions. This allows a safer and more pleasant experience. Therefore, it is important for the vehicle to be able to identify the situations in which it is immersed at a specific time, and thus to prevent its reaction to this situation. Some works that touch on topics of this type are [9, 10, and 11].

To achieve this level of automation, the designed system must:

i) Be an interactive control type human-in-the-loop, with 4 essential criteria:
   1. **Supervision**, must be able to determine if human intervention is necessary in function of past and current information about the system and its environment;
   2. **Minimally intervening**, must invoke the human operator when necessary;
   3. **Previsional**, can determine if a specification will be violated prematurely, and issues a warning to the human operator so that he has enough time to respond; and
   4. **Conditionally correct**, it must function correctly until a human intervention is considered necessary.

ii) Learn and reason according to the data it receives for automatic decision making, according to the context, i.e. with a level of autonomy, as it is analyzed in the work [12].

In order to define the context and exploit as much information as possible about the driver, the vehicle itself and its environment, the vehicle must be equipped with instrumentation. Thus, it can perceives those aspects that make up the driving process, with appropriate descriptors that allow characterizing this process, as can be seen in [13].

Work [14] presents a system for driving style recognition through a hierarchical pattern model. This pattern is composed of three levels: driving styles, driver states and driver emotions. Each event is represented by a descriptor, which represents a characteristic of the driver (e.g. facial expressions, tone of voice, etc.), or the environment in which the driving takes place (e.g. the type of road, weather conditions, etc.). This allow equipping vehicles with emotional intelligence, to make them able to recognize how the driver feels, and generate actions or behaviors adapted to that emotion.

According to [15], an autonomous vehicle must be capable of much more than recognition of emotions. As the information requested changes according to the level of automation in driving, the internal characteristics of the vehicle should be taken into account. A set of descriptors is proposed defining the characteristics to be taken into account. Each descriptor has variables that define it; each variable is composed by certain fields that allow specifying it. The source, from which it is possible to extract these properties, is provided using the standard equipment of a vehicle. The proposed characteristics are classified into two perceptions types:

i) External state perception: describes the variables that specify the external environment of the vehicle system. In our case, it represents **environmental** properties around the driving process, such as route descriptors (route type, altitude), current environment conditions (weather, temperature), conditions of the route (light conditions, surface quality) and aspects related to traffic (traffic density) (for more details, see Appendix 1, table 4).

ii) Internal state perception: describes the variables that composed the internal aspects of the vehicle system. In [15], it represents **driver** and **vehicle** properties in the driving process (for more details, see Appendix 1, table 5 and 6).
   a. Driver, through descriptors such as: basic characteristics of the driver (age, gender, reported violations, its physical limitation, driving experience), behavior of the driver (seat belt use, use horn, hands on the wheel, direction of look), driver physiological conditions (blood alcohol limit, blood pressure, heart rate, body temperature), and current emotion (facial and voice expressions).
   b. Vehicle, through descriptors such as: class of the vehicle (type of vehicle), types of mechanisms in the vehicle (speed, break, hand brake, fuel status, cruise control speed, seat belt use, state of doors), and current conditions of the vehicle (state of tires, airbag, car battery, steering system, service or maintenance).

The set of descriptors can be perceived in different ways (sound, vision, internet, etc.), making use of current technologies such as cloud, drivers, sensors, intelligent infrastructures, navigation systems, on-board diagnostic systems (OBD), and/or recognition applications. As we can see, an AV is an extremely complex system, made up of sensors, actuators and communication media, with built-in intelligence capable of making decisions and controlling the various parts of the vehicle. A complex system is mainly characterized by irreversible, sudden, unpredictable and aperiodic dynamics, and often has a large number of entities and interactions [16].

2.2 Learning Techniques

Autonomous vehicles must be able to learn by themselves, and through the collection of experiences, to generate deductions for decision making. The learning system of an autonomous vehicle works by learning to detect objects and patterns automatically, the more data it has the more accurate the system will be. The system is trained through repetition, continuously receiving input data and optimizing its ability to make predictions about what the data means and what to do with it [31]. Our work requires the instantiation of a learning mechanism to achieve the detection of abnormal situations of the normal ones, possible thanks to the repetitive behavior of the system studied, and artificial intelligence techniques.

Artificial learning, or machine learning (ML), is any method capable of generalizing behaviors and constructing models of the reality from unstructured information [17]. There are many ML techniques like: association rules, generic algorithms, support vector machines, Bayesian networks, among others. However, two major perspectives of ML that respond to learning problems in the context of complex systems like VA should be highlighted: i) neural network approaches, and ii) constructivist approach.

2.2.1 Neural Network Approaches

Neural Network Approaches are based on the functioning of biological neurons. A neuron has several inputs and one output, representing the axons of the biological neuron. More formally, the artificial neuron is a function that has a certain number of inputs and parameters (or weights) attached to each of the inputs, and that weight them [17]. This learning technique has been applied in different works associated with autonomous vehicles:

- Evolutionary method for the creation of an autonomous land vehicle controller based on an artificial neural network, techniques to integrate evolutionary search and error retro propagation are presented. They are designed to control Carnegie Mellon University's NAVLAB vehicles for roadside monitoring tasks [33].
- Implementation of two systems, framed in the "Intelligent Stop & Go" driver assistance system. The first system deals with the detection without segmentation of overtaking and ego estimation vehicles on highways. And the second allows the recognition of pedestrians in the traffic scenario of the city center [34].

Neural networks have a very positive dynamic, and their good results in many areas make them interesting candidates. However, the learning time is still very significant, which hinders the use of such approaches to achieve real-time learning without an efficient computer infrastructure [18]. In addition, they are difficult systems to parameterize, due to the multitude of variants, different parameters and topologies to choose [32]. Therefore, it is not a pertinent technique to meet our objectives.

2.2.2 Constructivist approaches

The constructivist approaches seek to imitate the mechanism of knowledge construction in the human being through interactions with the real world. The previously acquired concepts serve as a basis for assimilating and interpreting new experiences, and these old concepts themselves are reshaped in the light of these observations. The basic unit of knowledge in this theory is the scheme [35]. This learning technique has been applied in different works:

- Set of mechanisms to improve the capabilities of adaptation of schemes in the mechanism of constructive anticipatory learning (CALM) [18, 36].
- Implementation of a multi-agent learning system based on schemes, designed for environmental intelligence and adapted to continuous environments [18, 37].
- Constructivist learning system, also based on a multi-agent system, and applied to carry out a control activity in the context of traffic control [18, 38].

Constructivist approaches offer very interesting points, they allow to easily add new data sources during the execution as a new element. They are designed to learn as the learning system explores the environment. In addition, the learning carried out by a constructivist approach is independent of the task that will be carried out [18].
However, the main limitation is linked to the combinatorial explosion when it must scale, the constructivist approaches are from this point of view quite vulnerable to enlargement. On the other hand, the schemas in their original structure are not adapted to continuous data [18].

As we see, such learning technique can satisfy our needs but the scalability issue must be addressed. In this study, we investigate the Adaptive Multi-Agent System (AMAS) approach that have shown its adequacy to address real-time adaptive learning problems in similar conditions.

2.3 The AMAS theory

In this study, the developed system is based on the adaptive multi-agent system (AMAS) theory. In this section, this theory is presented after a brief description of multi-agent systems in general.

2.3.1 Multi-Agent System

There are many definitions of the term agent as well as many paradigms using it. A (mostly) consensual one proposed by Wooldridge in [22]:

"An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives."

This definition is enriched by Ferber [23], notably adding a locality criterion:

"*An autonomous physical or virtual entity able to act (or communicate) in a given environment given local perceptions and partial knowledge. An agent acts in order to reach a local objective given its local competence*"

From those two definitions of an agent, different characteristics can be noted [22] [23]:

- An agent is autonomous, which means it decides on its own
- It is situated in an environment, that it is able to perceive, and on which it can act
- It has a local perception of this environment
- It is able to communicate with other agents
- It owns resources and skills

The agent's behavior is ruled by three steps looping lifecycle:

1. *Perception*: in this step, the agent acquires information on its environment
2. *Decision*: in this step, the agent decides of the next actions it will perform
3. *Action*: in this step, the agent performs the decided action

A multi-agent system (MAS) is a system composed of several agents, interacting among each other's and with their environment.

2.3.2 Adaptive Multi-Agent System

An Adaptive Multi-Agent System (AMAS) is a MAS in which agents interact cooperatively. By their interactions, the system self-organizes itself and reaches its objective function.

The AMAS theory was developed by the SMAC team and formalized in [24]. It focuses in *cooperation* as the fundamental mechanism of self-organisation and self-adaptation in MAS design. To help this cooperation, each agent computes a local criticality measure that represents the distance between the agent and its local final objective. By interacting cooperatively, agents always try to help the most critical agent in order to reduce the criticality of the system.

To ease the design of such agents, the AMAS theory identifies seven generic NCSs [21], that are spread on the three steps of the agent lifecycle:

- Incomprehension: the agent is not able to extract information from a received message.
- Ambiguity: the exact meaning of a message cannot be determined, or missing information are required
- Incompetence: the agent does not have the capabilities to treat a received information.
- Unproductiveness: A received information does not lead to any useful conclusion.
- Conflict: The action of the agent is incompatible with an action from its environment.
- Competition: The action of the agent leads to the same result than an action from its Environment (another agent).
- Uselessness: The action of the agent has no effect on itself or its environment.

A cooperative agent actively tries to avoid these NCSs and solve them to the best of its capabilities. To this end, it uses one of three following mechanisms [21]:

- Tuning: The agent can change one or several of its internal parameters (e.g., adjusting the priorities of its behavior rules).
- Reorganization: The agent can change its relationship with its environment (e.g., removing or creating new links with other agents).
- Evolution: The agent can change the nature of its environment (e.g., removing or creating new agents).

The AMAS theory was successfully applied to a large variety of applications including:

- resolution of complex optimization problem such as the manufacturing control problem [26] or planning of heterogeneous constellation of satellites [29].
- real-time adaptive learning such as learning by demonstrations [27] or learning to control complex system [28].

Thus, this theory seems to be adapted to our problem. To study its relevance for developing the reSPONSiVA system, we use the ADELFE methodology (*Atelier pour le DEveloppement de Logiciels à Fonctionnalité Emergent*; or *Workshop for Software Development with Emerging Functionality*, in English), dedicated to the design of systems based on the AMAS theory. ADELFE considers all the sequences of activities of a classic software design from needs to implementation. It guarantees that the software is developed according to the AMAS theory [19]. In addition to this, it provides a tool based on a questionnaire that allows us to establish the relevance of the AMAS approach to our problem [20]. The questions try to identify if:

i) there exists an already known algorithm for solving the global task,
ii) the system involves a lot of heterogeneous entities to solve the problem, and if their behaviors can evolve,
iii) solutions is obtained by repeating tests,
iv) the system environment evolves, is dynamic, non-linear, and open.

To answer these questions, it was necessary to specify the AV system (see Appendix 2).

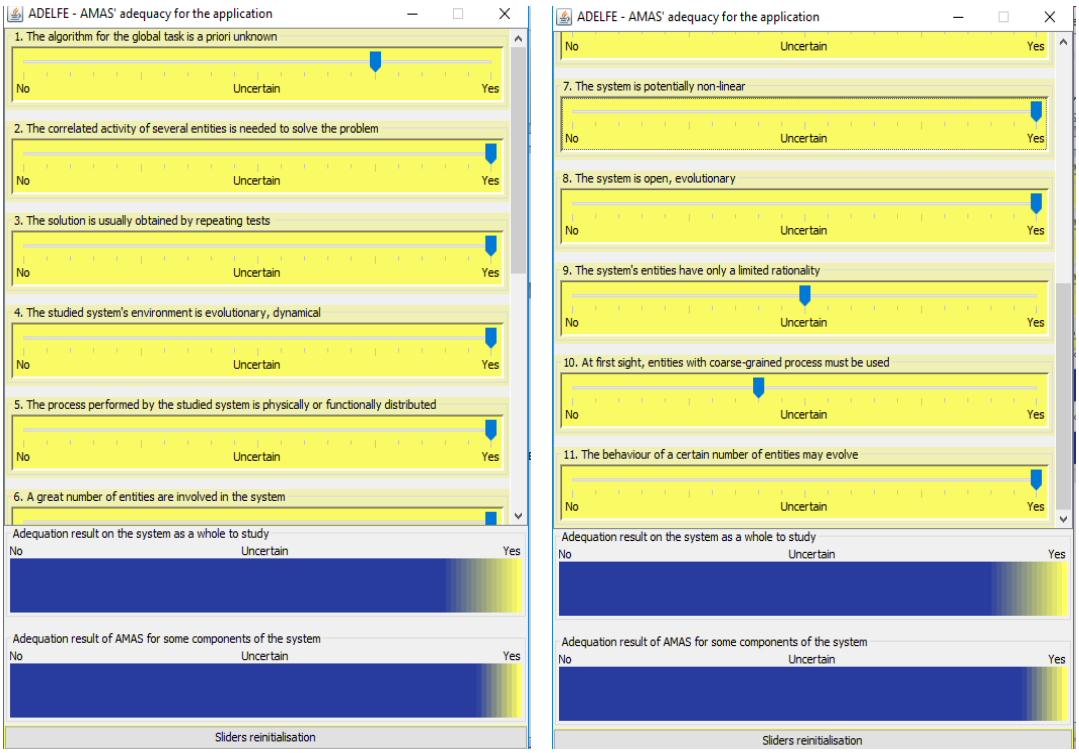

Figure 1. ADELFE tool

As shown in figure 1, the tool confirms the relevance of the use of the AMAS approach for the design of our system, as it answers correctly and pertinently to the characteristics of our problem:

- Absence of a known algorithm that answers the problem;
- It involves a large number of correlated, cooperative entities interacting with each other;
- With a dynamic, evolutionary, non-linear, open, and partially perceptible environment.

The AMAS theory has interesting properties to address complex systems such as AV. They may be represented by agents that are essentially characterized by their autonomous decision-making. Moreover, the MAS may be decentralized and particularly resistant and is able to react to noise, sudden changes and failures (self-adaptive). That is why; we used this paradigm to design our system.

Chapter 3. reSPONSiVA

This chapter concerns the two contributions of my internship. First, a presentation of the synthetic generator that allows defining a set of scenarios to test our system is done. Second, I present the self-adaptive multi-agent learning system, that computes a difference between the observed situation and the situation predicted by the vehicle and learns by feedback to distinguish abnormal situations from normal situations.

3.1 Synthetic Scenario Generator

In order to be able to test the system and due to the absence of the means to obtain data from real physical devices, I developed a generator of artificial data.
This generator aims at simulating an autonomous driving process that, in this study, is defined as a loop of three main steps:

i) Perceive the driving context
ii) Decide a set of actions to perform
iii) Transform the driving context by performing the decided actions.

In real physical environment, the driving context, composed of a set of variables describing the environment and the vehicle, is perceived (step 1) through physical sensors. The set of actions to perform (step 2) is decided by the driving algorithms of the vehicle. Once the set of actions is performed (step 3), the physical sensors provide the real driving context from which the driving process continue (step 1).

We consider that, while performing the actions, the driving algorithms can compute a predicted driving context (see figure 2). If, once the actions are performed, a large gap exists between the real perceived driving context and the computed predicted one, then the vehicle is in an abnormal situation. Such situations must be anticipated, to immediately inform the driver (supervisor) that take back the control. Once a normal situation is retrieved, the control can be given back to the driving algorithms inside the vehicle.

Based on that, the generator creates a set of N driving contexts. Thus, it starts by generating randomly a **driving context** and a **set of actions.** Then it applies a **transformation function** to obtain the new predicted driving context. As in normal situations the real driving context is equivalent to the predicted one, the generator uses this predicted driving context as the real one for starting the next generation step. In order to simulate the abnormal situations, for a subset M (with $M<N$) of driving contexts, the generator disrupts the predicted driving context thus creating a different real driving context from which the generation process restarts.

The generator allows to define different scenarios representing the evolution of the driving process. Each scenario is described by a set of driving contexts. As a driving context is a combination of the environment variables and the vehicle variables, it is represented by a set of variables V_n to which a set of actions A_m to be performed by the vehicle is generated at each step.

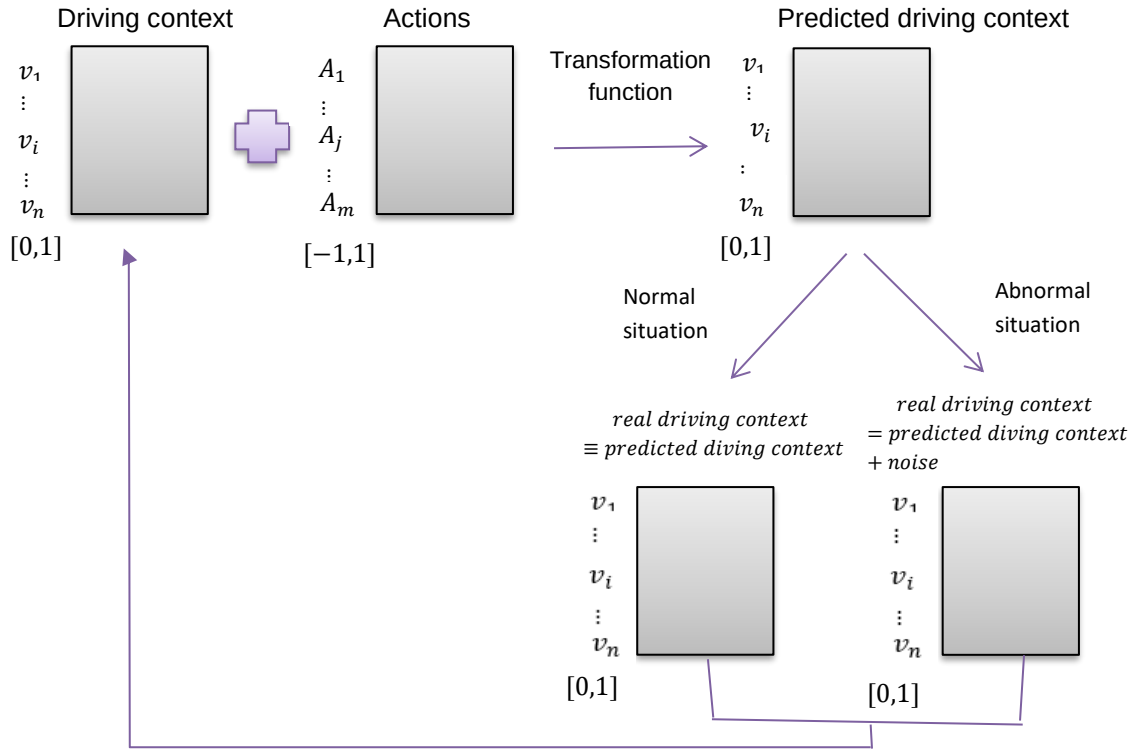

Figure 2. Structure of generator

The generator uses two main functions (described in the next sections):

- **Function 1:** generation of a normal situation: given a driving context and a set of actions, this function computes a predicted driving context and its corresponding real driving context. In this case, both driving contexts are equivalent.
- **Function 2:** generation of an abnormal situation: given a predicted driving context by function 1, function 2 disrupts it in order to compute a different real driving context.

Figure 3 shows the algorithm for generating a scenario. It works as follows: a set of chained normal situations is produced using function 1. At a random instant, a certain number of abnormal situations are introduced using function 2 and starting from the last driving context generated by function 1. This process is repeated until the number of desired situations to be generated in the scenario is reached. The Output is a situations file specifying at each time t, the predicted driving context, the real corresponding driving context and the type (normal/abnormal) of the situation. This file is later used by the learning system as described in section 3.2.1.2.

Once all the driving contexts are generated, they are written in a situations file specifying at each time t, the predicted driving context, the real corresponding driving context and the type (normal/abnormal) of the situation. This file is later used by the learning system as described in section 3.2.1.2.

Inputs:

- nbDCtoGenerate: number of diving context to generate
- minSN, maxSN: minimum and maximum number of normal situations to generate
- minSAN, maxSAN: minimum and maximum number of abnormal situations to generate
- noise_value: the noise to be added for the generation of abnormal situations

Terminology:

- DC: Driving Context
- NV: represents the noise vector
- IM: represents the influence matrix
- RDC: real driving context
- PDC: predicted driving context

```
Begin
[dc1] ← randomly_generated          //Generation of the first driving context
[Act1] ← randomly_generated         //Generation of the first vector of actions
PDC ← function1(dc1, Act1, IM)      //Computing the first predicted driving context using function1

nbDCgenerated = 1                   //Counter of generated driving contexts

    //Generation of nbDCtoGenerate driving contexts
While (nbDCgenerated < nbDCtog)
    nbSN = random (minSN, maxSN)        // Number of normal situations to generate
    nb = 1                              //Counter of situations
    //Generation of nbSN normal situations
    While (nb < nbSN & nbDCgenerated < nbDCtoGenerate)
        RDC ← PDC
        scenarios ← (T + (nbDCgenerated - 1) + PDC + RDC + "normal")
        Act ← randomly_generates            //Generation of next vector of actions
        PDC ← function1(RDC, Act, IM)       //Computing next predicted driving context
        nb++
        nbDCgenerated++
    End While

    nbSAN = random (minSAN, maxSAN) // Number of abnormal situations to generate
```

```
    nb = 1
    //Generation of abnormal situations
    While (nb < nbSAN & nbDCgenerated < nbDCtog)
        RDC = function2 (PDC, NV, noise_value)        //Computing next real driving context using function 2
        scenarios (T + (nbDCgenerated - 1) + PDC + RDC + "abnormal")
        Act ← randomly_generates                      //Generation of next vector of actions
        PDC ← RDC                                     //Initialization of next predicted driving context
        nb++
        nbDCgenerated++
    End While
End while
End
```

Figure 3. Macro-algorithm for scenarios generation

3.1.1 Function 1: Generation of a Normal Situation

This first function concerns the normal situations where the vehicle drives in the expected way. We remind that in this case, the predicted driving context is equivalent to the real one. Indeed, as the values of sensors are noisy and inaccurate, the predicted and the real driving contexts can be slightly different. The **transformation function** that applies the set of actions to the driving context to obtain the new predicted driving context, uses an **influence matrix** to control the weight of an action performed by the vehicle on the driving context, avoiding by that meaningless values. The definitions introduced below present some conceptual notions used by the algorithm implemented for this step.

Definition 1 presents the **variables** v_i that characterize a driving context of the driving process, through properties such as weather time, light conditions, etc. In our system, we leave the semantic meaning of the variables, being represented as generic input signals with values randomly generated between 0 and 1.

D**efinition 2** shows the possible **actions** *Act* to be performed in the environment such as change of speed, brake, etc. As the previous definition, we do not use semantic meaning, but generic variables with values randomly generated between -1 and 1, since in this way we can symbolize the positive or null (≥ 0), or negative (< 0) effects of an action.

Definition 3 concerns the **influence matrix** IM that specifies the influence of actions on the variables of the driving context. The influence value a_{ij}, is 1 when A_j influences v_i, 0 otherwise. In a real system this matrix can be given by the constructor, or learned by learning techniques, in our case, it's generated in a random way.

Definition 4 introduces the **transformation function** that computes a predicted driving context given the set of actions and the influence matrix. For each variable v_i, the sum of values of the actions

influencing v_i is computed $(ActionInfluence = \sum A_j)$. If $ActionInfluence$ is negative (< 0) then the function *Eq.3* is used; if it is positive (≥ 0) the equation *Eq.4* is used.

Definition 1: Driving context variables	$V = \{v_1, \dots, v_i, \dots, v_m\}_{[0,1]}$ (**Eq.** 1) Where, • $v_i =$ represents a variable from the environment or the vehicle. • $[0,1] =$ range of values of v_i.
Definition 2: Actions variables	$Act = \{A_1, \dots, A_j, \dots, A_n\}_{[-1,1]}$ (**Eq.** 2) Where, • $Act =$ represents the set of possible actions. • $A_j =$ represents a possible action. • $[-1,1] =$ range of values of A_j.
Definition 3: Influence matrix	$IM = \begin{matrix} A_1 & A_j & A_n \end{matrix}$ $\begin{bmatrix} a_{11} \dots a_{1j} \dots a_{1n} \\ \vdots \quad \ddots \quad \vdots \\ a_{i1} \dots a_{ij} \dots a_{in} \\ \vdots \quad \ddots \quad \vdots \\ a_{m1} \dots a_{mj} \dots a_{mn} \end{bmatrix} \begin{matrix} v_1 \\ \vdots \\ v_i \\ \vdots \\ v_m \end{matrix}$ Where, • $IM =$ represents the influence matrix, of rows ***m*** being the number of variables in the driving context, and of columns ***n*** the number of possible actions • $a_{ij} = \begin{cases} 1 \ if \ Aj \ influences \ v_i \\ 0 \ if \ not \end{cases}$
Definition 4: Transformation function	$For\ i = 1 \rightarrow m$ $ActionInfluence = \sum_{j=1}^{n} A_j$ If $ActionInfluence < 0$ $v_i' = v_i \cdot \left(\frac{1}{1 - ActionInfluence}\right)$ (**Eq.** 3) else $v_i' = v_i + (1 - v_i') \cdot ActionInfluence$ (**Eq.** 4) Where, • $v_i' =$ represents the predicted value of v_i once the actions are performed.

Figure 4. Conceptual notions used by the generator

3.1.2 Function 2: Generation of an Abnormal Situation

This second function concerns the abnormal situations where the vehicle does not drive in the expected way. In this case, the predicted environment is different from the real environment. For the generation of such situations, the last computed predicted driving context from function 1 is disrupted using a **noise function** as given by definitions 5 and 6.

Definition 5 specifies the **noise vector** *Noise* defining the variables of the driving context to which a noise is applied in case of an abnormal situation. The values of n_i are 0 or 1: 1 if a noise must be added to v_i, 0 otherwise. The noise value to be added is fixed as an input of the algorithm. In this study, this vector is generated randomly.

Definition 6 presents the **noise function** that computes the real driving context. As inputs, it requires the last predicted driving context obtained by function 1, the vector of noise and the noise value. As outputs, it produces an abnormal situation. It works as follow: for each variable v_i of the last predicted driving context, if the corresponding n_i is equal to 1 then the noise value is added to v_i, otherwise nothing changes.

Definition 5: Noise vector	$Noise = [n_1, \dots, n_i, \dots, n_m]$ Where, • $n_i = \begin{cases} 1 \; if \; a \; noise \; must \; be \; added \; to \; v_i \\ 0 \;\; Otherwise \end{cases}$
Definition 6: Noise function	$For \; i = 1 \rightarrow m$ If $n_i = 1$ $v_i' = v_i + noiseValue$ Where, • $n_i =$ indicates given the noise vector if a noise must be added • $noiseValue =$ the noise to be added to a v_i given in inputs

Figure 5. Real driving context generation in abnormal situation

3.2 Prediction Model of Situations

The RECOVAC project is responsible for the vehicle system management of:

- Monitoring the state of the system tasks and supervising its behaviour.
- Detecting abnormal situations in order to provide the driver with adequate information to recover the control of the vehicle. Vice versa, the system must be able to take back the control of the vehicle based on self-observations.

The reSPONSiVA system, part of RECOVAC, oversees:

- The detection of situations in the context,
- The identification of the type of situations: normal or abnormal, and
- The prediction of difficult situations to handle by the driving system.

reSPONSiVA collects the data related to the situation, analyses them, and indicates the type of the detected situation. In addition to this, it registers in a file the parameters that show the behaviour of the system.

To predict the type of the situation, reSPONSiVA uses a learning mechanism by feedback to distinguish abnormal situations from normal situations. The learning mechanism used in this work is based on an adaptive multi-agent system, presented in section 3.2.1, that detects if the vehicle: i) is not able to react in the expected way, or ii) reacts in the expected way. The output of the multi-agent system is a file of learned AbNormal Situation (ANS).

Once the learning is finalized, reSPONSiVA uses the ANS file to predict the types of new arriving driving contexts provided by the generator. It works as follow, for each new driving context, a comparison to the driving contexts in the ANS file is done. The considered driving context is considered as an abnormal situation if at least a given number x of its variables values is equal to those of a driving context contained in the ANS file (x depends on the number of varaiables describing a driving context). If this new driving context is considered abnormal, then reSPONSiva adds it to the ANS file. In this study, if no similar driving context is found in the ANS file, the driving context is considered as normal. This last part of my internship was implemented just to test the ability of the MAS to learn. A lot of improvements must be done, a list of perspectives is detailed in the last chapter.

3.2.1 The AnoMASly agent model

The multi-agent learning system I defined and implemented in this internship is an instantiation of the AnoMASly model developed by the SMAC team [25].

AnoMASly is a generic agent model developed for anomaly detection in any context. It is based on the following terminology:

- Activation profile of a sensor: the context is observed through sensors ***S***, each captor s ∈ S has an activation profile at all times ***t***. This profile $\boldsymbol{A_S^t}$, is described by a set of ***N*** events associated with the captor occurred in the lasts ***N*** cycles.

$$A_S^t = [\, e_s^0, e_s^1, \dots, e_s^N]$$

- Nominal profile of a sensor: $\boldsymbol{P_S^t}$ is the set of reference values r ∈ R that define the nominal profile of a captor s ∈ S at time ***t***. Each reference value is associated with a set of examples of nominal behavior.

$$P_S^t = [\, r_s^1, r_s^2, \dots, r_s^N]$$

- Disparity Values: dissimilarity between the activation and nominal profile of a captors ∈ S at time ***t***.

$$\Delta_S^t = \sum_{i=0}^{N} \left| r_s^i - \ e_s^i \right|$$

- Detection of anomalies: generation of alert when abnormal behavior is detected. The detection of anomalies is done through a linear combination of disparity values $\boldsymbol{\Delta_S^t}$ of each sensor s ∈ S. To

model the influence of each dissimilarity value on the anomaly, a weight value is used $\boldsymbol{w_S}$ associated to each sensor. The criticality of a situation is calculated in each step of time $\boldsymbol{t}$.

$$C_t = \sum_{s \in S} \Delta_s^t \cdot w_s^t$$

The anomaly alert is raised when the criticality exceeds a threshold $C_t > T$, with $\boldsymbol{T}$ an input of the system and $\boldsymbol{w_s^t}$ represent weights associated to the captors. If the system raises an alert, the user makes a feedback. There are 4 types of feedback: (1) False positive, the system generates alert (detects abnormal situation) but is not classified as abnormal by the user, (2) False negative, the user detects abnormal behavior while the system does not active alarm, (3) True positive, an alert was generated and the user confirms the existence of an abnormal situation, and (4) true negative, the system did not generate any alarm and the user confirms that there was no abnormal situation. Each feedback allows readjusting the weights associated to the sensors, which allows adjusting the system using a linear optimization process with a dynamic set of constraints.

The multi-agent system composing AnoMASly counts three types of agents:

- *Profile* Agents: each agent is associated with an environmental sensor and a single *weight* agent. Its roles is to compare the nominal profile and the activation profile of the sensor s ∈ S, and to send the disparity value of the sensor Δ_S^t at time *t* to its associated weight agent.
- *Weight* Agents: each agent is associated with a single *profile* agent and interacts with all the *constraint* agents. Its role is to calculate and send dynamically the value $(\Delta_s^t \cdot w_s^t)$ of the sensor s ∈ S to the alert activation rule.
- *Constraints* Agents: they model an inequality. They perform the calculation of a criticality C_t, who expresses its degree of satisfaction, if $C_t > T$ an alert is raised, nothing otherwise. The local objective of each *constraint* agent is to minimize its criticality being as close as possible to the defined threshold *T*. Thus, if $C_t > T$, it may ask *weight* agents to decrease their values, otherwise it asks them to increase their values.

The steps of AnoMASly are:

1. Create a *profile* agent for each sensor and associate it to a *weight* agent.
2. Each *profile* agent computes its disparity: the variation between the nominal profile and the activation profile of its sensor.
3. For each new encountered situation, a constraint agent is created. It calculates, thanks to the disparity and the weights given by the *weight* agents, the criticality of the current situation. When the criticality reaches a threshold, an alarm is activated.
4. Receive from the user the feedbacks about these alerts: false positives / negatives, and true positives / negatives.
5. Inform constraints agents about the feedback so they can ask the *weight* agents to adapt their values according to the received feedback.

During the learning phase, the failure of the system results in raising false negatives or false positives alerts detected by the feedback. These alerts are the consequences of failures given by the bad estimation of the weights associated to the sensors that characterize the situation, thus leading to a wrong collective estimation of the type of a situation. The cooperative rule of the model is to allow *weight* agents to adjust their values to only raise true positive alerts and correctly identify the type of the situations.
For that, at each step, two *constraint* agents are identified: *CHigh* and *CLow*. *CHigh* is the *constraint* agent requesting an increase of weights with the higher criticality level. *CLow* is the *constraint* agent requesting a decrease with the higher criticality level. As they request antagonist actions (increase and decrease), reducing the criticality of one involves the increase of the other criticality. Depending on which of the two *constraint* agents has the higher criticality, each *weight* agent decides to do nothing or increase or decrease its current value. Every agent must locally choose the most cooperative action, which reduces the difference between the criticalities of *CHigh* and *CLow*.
The resolution process is then a succession of increase and decrease requests sent by *constraint* agents to *weight* agents and adjustment decisions made by the *weight* agents. The successive cooperative resolution of these requests enables *weight* agents to tune their weights, and thus to converge towards optimal parameters.

3.2.1.1 AnoMASly instantiation for reSPONSiVA

The three types of agents that composes AnoMASly are specified in reSPONSiVA as follows:

- *Profile* Agents: represents the values of disparity associated to the difference between the current driving context (last real driving context) and the predicted one.
- *Weight* Agents: represents weight of the variables that characterize the driving context.
- *Constraints* Agents: represents each driving context in a scenario, these are found in the situations file obtained by the generator. For the initialization phase of AnoMASly, we define two virtual constraint agents, with a fixed location of 20% with respect to the threshold value and a fixed value of criticality of -0.2. Those constraints agent will be used to compare with the first driving context, allowing weight adjustment in a more controlled manner.

Figure 6 illustrates the functioning of AnoMASly, the algorithm is presented in figure 7. The input of the algorithm is the data of the situations file produced by the generator. The output of AnoMASly is the ANS file that contains the learned abnormal situations. It works as follow: AnoMASly starts by creating the required *profile* and *weight* agents. For each line in the situations file, AnoMASly receives the predicted driving context at time t and the last real driving context (at t-1) (step 1 of figure 6). The *profile* agents associated to each variable of the driving context computes the disparity between both driving contexts. The *constraint* agent associated to the predicted arriving driving context computes its criticality and returns the type of the situation (normal or abnormal) (step 2 of figure 6). The type of the situation is then sent to the situations file (step 3 of figure 6). This type is compared to the one in the situations file and a feedback (step 4 of figure 6) is given back to the multi-agent system allowing the adjustment of the *weight* agents. Finally, all abnormal situations detected are stored in a file, called abnormal situations file (ANS) (step 5 of figure 6).

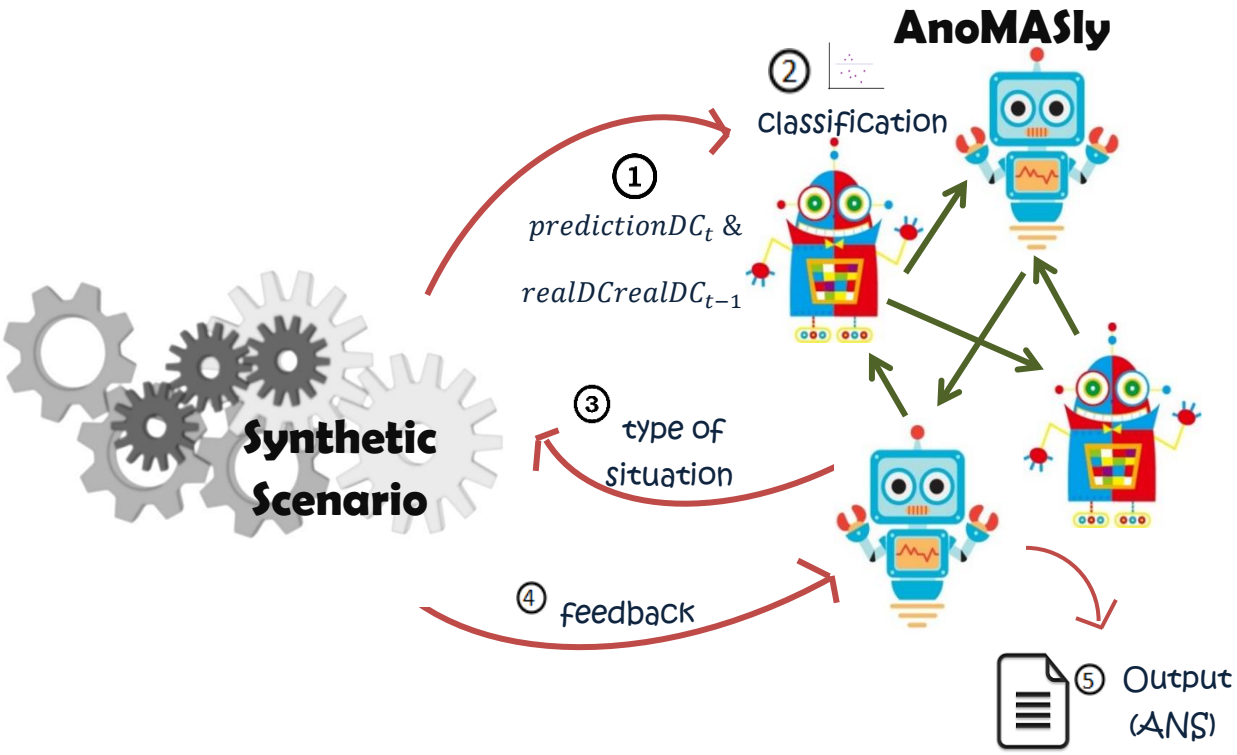

Figure 6. Structure of AnoMASly for reSPONSiVA

```
Begin
ArrayList drivingContexts = read(situationsFile)        //read the situations file provided by the generator
ProfilAgent(data) = new AnoMASly()        //instantiation of AnoMASly
For each drivingContext in  drivingContexts                //predict the type of a new driving context
        AnoMASly.EventClass event = classifier.newEvents(drivingContext)
        If event  generate alert
            type = "abnormal"
        Else
            Type = "normal"
        End if
            //Feedback to the system
        If type = data.type and data.type = abnormal
            Classifier.feedback = true positive
        If type = data.type and data.type = normal
            Classifier.feedback = true negative
        If type != data.type and data.type = abnormal
            Classifier.feedback = false negative
        Else
            Classifier.feedback = false positive
classifier.resolveCriticality()                //ask the MAS to minimize criticalities
If (type = "abnormal" and classifier.feedback = true positive)
        ANS ← drivingContext
End
```

Figure 7. Macro-algorithm for learning situations

Chapter 4. Experimentation and Results

The previous chapter introduces the two contributions of my internship identifying two major tasks: i) development of a synthetic generator of situations and ii) implementation of the reSPONSiVA system. This chapter describes the experiments performed to validate the developed system.

4.1 Generation of Scenarios with the Synthetic Generator of Situations

For the implementation of the scenario generator, the java language was used.

To validate reSPONSiVA, we generate several scenarios. This section presents their global characteristics. In this study, a driving context is described by a set of 12 variables, each randomly generated between 0 and 1. The set of actions is composed of 10 actions; each randomly generated between -1 and 1.
As seen in the algorithm for scenarios generation (figure 3), several inputs are required. For the reSPONSiVA validation, they were initiated as follow:

- nbDCtoGenerate: minDC = 8, maxDC = 250
- minSN = 1, maxSN = 5
- minSAN = 1, maxSAN = 3
- noise_value = 0.05

Table 1 shows an example of a generated scenario counting 11 driving contexts. The scenario is a repeated cycle of 1 to 5 normal situation (blue), continued by 1 to 3 abnormal situations (red), till the number of situations to generate its reached. We remark that the predicted driving context and the real driving context, for normal situations are equivalent (see table 1, row 1, line 1), and for abnormal situations they are different with a noise value defined as input (see table 1, row 1, line 13). As we said before, the situations file specifies for each generated situation: the time, the predicted driving context, the real driving context and the type (normal/abnormal) of the situation (see table 1, row 2).

Generation scenarios	Scenarios e_normal 1 : **[0.511, 0.409, 0.855, 0.606, 0.646, 0.339, 0.073, 0.26, 0.538, 0.072, 0.923, 0.039]** : **[0.511, 0.409, 0.855, 0.606, 0.646, 0.339, 0.073, 0.26, 0.538, 0.072, 0.923, 0.039]** e_normal 2 : [0.638, 0.234, 0.246, 0.186, 0.298, 0.341, 0.505, 0.027, 0.374, 0.627, 0.515, 0.042] : [0.638, 0.234, 0.246, 0.186, 0.298, 0.341, 0.505, 0.027, 0.374, 0.627, 0.515, 0.042] e_normal 3 : [0.795, 0.903, 0.68, 0.015, 0.677, 0.396, 0.37, 0.704, 0.991, 0.446, 0.577, 0.882] : [0.795, 0.903, 0.68, 0.015, 0.677, 0.396, 0.37, 0.704, 0.991, 0.446, 0.577, 0.882] e_normal 4 : [0.838, 0.179, 0.293, 0.239, 0.529, 0.268, 0.531, 0.503, 0.167, 0.645, 0.725, 0.555] : [0.838, 0.179, 0.293, 0.239, 0.529, 0.268, 0.531, 0.503, 0.167, 0.645, 0.725, 0.555] e_normal 5 : [0.867, 0.953, 0.921, 0.677, 0.464, 0.42, 0.257, 0.128, 0.759, 0.169, 0.843, 0.612] : [0.867, 0.953, 0.921, 0.677, 0.464, 0.42, 0.257, 0.128, 0.759, 0.169, 0.843, 0.612] e_abnormal 1 : **[0.66, 0.054, 0.632, 0.69, 0.19, 0.499, 0.408, 0.059, 0.239, 0.377, 0.059, 0.514]** : **[0.61, 0.004, 0.632, 0.69, 0.14, 0.499, 0.358, 0.059, 0.239, 0.377, 0.009, 0.464]** e_abnormal 2 : [0.986, 0.161, 0.459, 0.289, 0.146, 0.668, 0.721, 1.0, 0.729, 0.575, 0.797, 0.647] : [0.936, 0.161, 0.459, 0.239, 0.096, 0.668, 0.671, 0.981, 0.729, 0.575, 0.797, 0.647] e_abnormal 3 : [0.293, 0.901, 0.735, 0.506, 0.536, 0.399, 0.632, 0.73, 0.274, 0.784, 0.934, 0.477] : [0.243, 0.901, 0.735, 0.506, 0.486, 0.399, 0.632, 0.73, 0.224, 0.734, 0.884, 0.427] e_normal 6 : [0.323, 0.431, 0.836, 0.415, 0.584, 0.623, 0.181, 0.154, 0.018, 0.746, 0.894, 0.655] : [0.323, 0.431, 0.836, 0.415, 0.584, 0.623, 0.181, 0.154, 0.018, 0.746, 0.894, 0.655]

	e_normal 7 : [0.598, 0.191, 0.835, 0.562, 0.142, 0.697, 0.096, 0.423, 0.391, 0.646, 0.018, 0.88] : [0.598, 0.191, 0.835, 0.562, 0.142, 0.697, 0.096, 0.423, 0.391, 0.646, 0.018, 0.88] e_abnormal 4 : [0.607, 0.494, 0.305, 0.044, 0.095, 0.199, 0.403, 0.231, 0.719, 0.013, 0.523, 0.44] : [0.607, 0.444, 0.255, 0.044, 0.045, 0.149, 0.353, 0.231, 0.719, 0.013, 0.523, 0.44]
Situations file	Situations 1 > time 0 > Env_normal > [0.511, 0.409, 0.855, 0.606, 0.646, 0.339, 0.073, 0.26, 0.538, 0.072, 0.923, 0.039] > [0.511, 0.409, 0.855, 0.606, 0.646, 0.339, 0.073, 0.26, 0.538, 0.072, 0.923, 0.039] Situations 2 > time 1 > Env_normal > [0.638, 0.234, 0.246, 0.186, 0.298, 0.341, 0.505, 0.027, 0.374, 0.627, 0.515, 0.042] > [0.638, 0.234, 0.246, 0.186, 0.298, 0.341, 0.505, 0.027, 0.374, 0.627, 0.515, 0.042] Situations 3 > time 2 > Env_normal > [0.795, 0.903, 0.68, 0.015, 0.677, 0.396, 0.37, 0.704, 0.991, 0.446, 0.577, 0.882] > [0.795, 0.903, 0.68, 0.015, 0.677, 0.396, 0.37, 0.704, 0.991, 0.446, 0.577, 0.882] Situations 4 > time 3 > Env_normal > [0.838, 0.179, 0.293, 0.239, 0.529, 0.268, 0.531, 0.503, 0.167, 0.645, 0.725, 0.555] > [0.838, 0.179, 0.293, 0.239, 0.529, 0.268, 0.531, 0.503, 0.167, 0.645, 0.725, 0.555] Situations 5 > time 4 > Env_normal > [0.867, 0.953, 0.921, 0.677, 0.464, 0.42, 0.257, 0.128, 0.759, 0.169, 0.843, 0.612] > [0.867, 0.953, 0.921, 0.677, 0.464, 0.42, 0.257, 0.128, 0.759, 0.169, 0.843, 0.612] Situations 6 > time 5 > Env_abnormal > [0.61, 0.004, 0.632, 0.69, 0.14, 0.499, 0.358, 0.059, 0.239, 0.377, 0.009, 0.464] > [0.66, 0.054, 0.632, 0.69, 0.19, 0.499, 0.408, 0.059, 0.239, 0.377, 0.059, 0.514] Situations 7 > time 6 > Env_abnormal > [0.936, 0.161, 0.459, 0.239, 0.096, 0.668, 0.671, 0.981, 0.729, 0.575, 0.797, 0.647] > [0.986, 0.161, 0.459, 0.289, 0.146, 0.668, 0.721, 1.0, 0.729, 0.575, 0.797, 0.647] Situations 8 > time 7 > Env_abnormal > [0.243, 0.901, 0.735, 0.506, 0.486, 0.399, 0.632, 0.73, 0.224, 0.734, 0.884, 0.427] > [0.293, 0.901, 0.735, 0.506, 0.536, 0.399, 0.632, 0.73, 0.274, 0.784, 0.934, 0.477] Situations 9 > time 8 > Env_normal > [0.323, 0.431, 0.836, 0.415, 0.584, 0.623, 0.181, 0.154, 0.018, 0.746, 0.894, 0.655] > [0.323, 0.431, 0.836, 0.415, 0.584, 0.623, 0.181, 0.154, 0.018, 0.746, 0.894, 0.655] Situations 10 > time 9 > Env_normal > [0.598, 0.191, 0.835, 0.562, 0.142, 0.697, 0.096, 0.423, 0.391, 0.646, 0.018, 0.88] >[0.598, 0.191, 0.835, 0.562, 0.142, 0.697, 0.096, 0.423, 0.391, 0.646, 0.018, 0.88] Situations 11 > time 10 > Env_abnormal > [0.607, 0.444, 0.255, 0.044, 0.045, 0.149, 0.353, 0.231, 0.719, 0.013, 0.523, 0.44] > [0.607, 0.494, 0.305, 0.044, 0.095, 0.199, 0.403, 0.231, 0.719, 0.013, 0.523, 0.44]

Table 1. Example of a scenario generation

4.2 Experimentation and Validation of reSPONSiVA

The aim of the experimentations is to:

- Evaluate the process of learning abnormal context situations, which makes use of the AnoMASly system.
- Study the correct performance of the reSPONSiVA system, which evaluates driving contexts looking to correctly identify the type of current situation.

4.2.1 Evaluation of the Learning Process

This first experience aims at studying the capacity of the algorithm of *Figure 7* to correctly classify situations by analyzing the capacity of the learning system to reach a certain level of precision in weight estimation, and the influence of the number of situations (driving contexts) in the evolution of the criticality.

Figures 8.a, 8.b and 8.c shown below present the evolution of the criticality of *CHigh* (red), and *CLow* (blue) during the simulation of n situations, n being 25 (Figure 8.a), 125 (Figure 8.b) and 250 (Figure 8.c). Remind that *CHigh* and *CLow* are the *constraint* agents requesting the increase respectively the decrease of the weight values with the higher criticality level at each learning step. The figures illustrate the process of minimization of criticalities through the cooperative behavior of weights agents.

At the beginning of the learning process, as few situations are known, the criticalities of new situations can be very different. This explains why, in some points of the curves, the criticalities increase or decrease rapidly. Still, the figures show that after few steps of the learning process, a convergence towards the value minimizing both criticalities are reached.

According to the learning model, the more situations are encountered, the more the *CHigh* and *CLow* criticality values will tends to zero and the more precise the weights associated to the sensors will be. In our case, figures show that, when more situations are added, the criticality values tend to be quite large. With n = 25, the criticalities tend to zero, however with n = 125 and 250 the criticalities tend to 700. Concluding that, as the number of situations increases, the restrictions of AnoMASly and the linear combination function used fail to satisfy the desired precision in the estimation of weights.

(a) n = 25

(b) n = 125

(c) n = 250

Figure 8. Evolution of the criticality of CHigh and CLow during n situations

4.2.2 Evaluation of the Prediction of Abnormal Situations

This second experience aims at evaluating, once the learning process is finalised, the ability of reSPONSiVA to correctly predict the abnormal situations within a set of new occurrences of driving contexts.

For that, a set of driving contexts were generated (see table 2, row 1). For each driving context, reSPONSiVA uses the ANS established file to classify the situation (see table 2, row 2).

Generated driving contexts	Situations 1 > time 0 : [0.463, 0.767, 0.095, 0.916, 0.46, 0.595, 0.898, 0.778, 0.914, 0.492, 0.383, 0.917] Situations 2 > time 1 : [0.801, 0.467, 0.599, 0.898, 0.472, 0.83, 0.745, 0.722, 0.191, 0.062, 0.397, 0.973] Situations 3 > time 2 : [0.384, 0.579, 0.2, 0.975, 0.502, 0.277, 0.669, 0.626, 0.183, 0.175, 0.994, 0.89] Situations 4 > time 3 : [0.943, 0.811, 0.831, 0.277, 0.626, 0.783, 0.418, 0.547, 0.698, 0.663, 0.379, 0.801] Situations 5 > time 4 : [0.624, 0.473, 0.535, 0.752, 0.533, 0.249, 0.343, 0.646, 0.681, 0.239, 0.054, 0.748] Situations 6 > time 5 : [0.783, 0.345, 0.909, 0.648, 0.721, 0.648, 0.34, 0.753, 0.93, 0.71, 0.749, 0.965] Situations 7 > time 6 : [0.326, 0.438, 0.141, 0.485, 0.319, 0.047, 0.317, 0.088, 0.133, 0.593, 0.04, 0.032] Situations 8 > time 7 : [0.666, 0.821, 0.747, 0.613, 0.236, 0.28, 0.611, 0.731, 0.939, 0.209, 0.577, 0.594] Situations 9 > time 8 : [0.556, 0.505, 0.716, 0.426, 0.844, 0.344, 0.237, 0.529, 0.216, 0.961, 0.998, 0.532] Situations 10 > time 9 : [0.665, 0.272, 0.392, 0.235, 0.898, 0.999, 0.458, 0.349, 0.136, 0.345, 0.241, 1.0] Situations 11 > time 10 : [0.416, 0.746, 0.959, 0.599, 0.721, 0.741, 0.884, 0.657, 0.905, 0.676, 0.721, 0.809] Situations 12 > time 11 : [0.524, 0.633, 0.78, 0.23, 0.807, 0.348, 0.97, 0.168, 0.361, 0.238, 0.185, 0.336] Situations 13 > time 12 : [0.545, 0.373, 0.539, 0.385, 0.322, 0.731, 0.371, 0.141, 0.61, 0.108, 0.418, 0.282] Situations 14 > time 13 : [0.426, 0.582, 0.408, 0.83, 0.945, 0.907, 0.927, 0.339, 0.408, 0.93, 0.668, 0.236]
reSPONSiVA predictions	reSPONSiVA The situation 1 it's predicted Normal The situation 2 it's predicted Normal The situation 3 it's predicted Normal The situation 4 it's predicted Abnormal The situation 5 it's predicted Abnormal The situation 6 it's predicted Abnormal The situation 7 it's predicted Abnormal The situation 8 it's predicted Normal The situation 9 it's predicted Abnormal The situation 10 it's predicted Abnormal The situation 11 it's predicted Abnormal The situation 12 it's predicted Normal The situation 13 it's predicted Abnormal The situation 14 it's predicted Abnormal Statistics: vp = 5 vn = 5 fp = 4 fn = 0

Table 2. Example of input file given to reSPONSiVA and the obtained results

The evaluation studies the percentage of errors produced by the system evaluating abnormal situations on a set of scenarios. For this, 20 scenarios were generated, with a total of more than 249 situations. For each scenario, table 8 presents the number of false positive (f_p) / negative (f_n) and true positive (t_p) / negative (t_n) produced.

Scenario	Nb Situations	f_p	f_n	t_p	t_n
1	10	1	2	2	5
2	15	2	4	2	7
3	12	3	2	3	4
4	12	0	4	2	6
5	10	5	4	0	1
6	13	5	0	4	4
7	12	5	2	3	2
8	14	4	2	3	5
9	14	3	3	3	5
10	13	4	2	1	6
11	12	5	2	1	4
12	13	4	3	1	5
13	13	3	2	2	6
14	13	3	3	3	4
15	12	5	1	2	4
16	12	1	3	2	6

17	12	6	3	1	2
18	11	4	3	3	1
19	11	3	4	2	2
20	15	3	3	1	8
Total	249	52	69	41	87

Table 3. Prediction of abnormal situations

As we can see in table 3, a success rate of 51.41% is obtained, i.e. over 249 situations, the system managed to identify 128 situations correctly, while 121 situations were incorrectly identified. The amount of erroneously detected data is possibly due to the form of the proposed resolution. By using an offline file of learned abnormal situations, the sensitivity and accuracy of the system are lost:

1. The sensitivity: since our evaluation method is subject to a fixed value of similarity (at least 6 variables are similar among 12), which, as it increases or decreases, influences the probability of matching between the variables of the studied driving context and those of the ANS file, thus producing false negative.
2. The accuracy: as our method uses a comparison file that specifies only abnormal situations, reSPONSiva is unable to recognize an abnormal situation if it is not found or does not have similarity with any driving context described in ANS, thus generating false positive and affecting the accuracy of the system.

Both experiences underline several perspectives to improve our system. The following chapter concludes this study and presents them.

Chapter 5. Conclusion and Perspectives

This last chapter concludes this study and presents some perspectives for resSPONSiVA and the RECOVAC project.

5.1 Conclusion

The great advances in the field of robotics, artificial intelligence and machine learning have a direct influence on automation processes. Automation is increasingly involved in everyday activities, sometimes exceeding human performance in a wide range of tasks, including those requiring cognitive skills.
Automation makes possible to improve the performance of tasks, reducing errors and improving the quality, speed and productivity of solutions, in some cases achieving results that go beyond human capacity. However, the scope of automation is affected by social, technical, and economic factors, so it is important to study the elements involved in an automated system.

Automotive technology is a fast-growing area with a wide generation of knowledge. Technological evolution in this area has led to the emergence of a new concept call autonomous vehicles. AV, vehicles capable of moving and transporting without human intervention.
During the internships, we work with AV of level 3, where the automated driving system performs all the dynamic driving tasks with the possibility of driver intervention if needed. Specifically, the internship aimed to identify by real time self-observation situations where the vehicle can no longer guarantee driving. To meet the objective, two main points was carried out:

- Implementation of a synthetic scenario generator, allowing the definition of a set of scenarios that simulated an autonomous driving process. Each driving context were specified as a combination of environmental and vehicle variables. A transformation function combines the generated driving context with a set of actions to compute the predicted driving context which is equal to the real driving context simulating normal situation. Particular driving context have been defined as abnormal, for these situations, noise is introduced in order to simulate a difference between the real environment and the predicted environment.
- Development of a situation prediction mechanism that allows to predict the type (normal/abnormal) of the new driving contexts provided by the generator.
  For this purpose, a self-adaptive multi-agent learning system was used, which calculates a difference between the observed situation and the situation predicted by the vehicle and learns by feedback to distinguish abnormal situations from normal ones.

5.2 Perspectives

The reSPONSiVA prediction model needs to be improved.

First, as shown in the results of AnoMASly, a convergence towards the value minimizing criticalities is reached. Nevertheless, as the number of situations increases, the restrictions of AnoMASly and the linear combination function used fail to satisfy the desired precision (criticity = 0) in the estimation of weights. This can be due to the way the data is generated or to the fact that the linear combination function is not suitable for our case. More investigations and experiments have to be done in order to improve the cooperative behaviour of the *weight* agents

Second, our system can anticipate the type of situations using an offline file containing the specification of certain abnormal situations detected with AnoMASly. However, this file is insufficient when an entry situation is not similar to any of those already encountered. For this reason, reSPONSiVA and AnoMASly must be coupled and the learning process must continue in real-time mode. In case a situation cannot be identified by reSPONSiVA, AnoMASly must immediately analyze it in order to detect its type. Allowing thus, the continuous learning of situations.

Finally, the similarity function used in this study must be improved. In this study, two driving contexts were similar if at least x variables are similar (6 over 12 for the experience). Another way to compute this similarity is the distance computation that will be investigated in the near future.
.

Appendix

Appendix 1. Characteristics that describe a driving context through the internal and external perception of the system

i) External state perception: environment

Descriptor	Description	Variable	Field	Source
Characteristics of Route	Describe the category of the route	Altitude	-	GPS
		Type of road	Number of intersections	GPS
			Number of carriageway road	
			Number of traffic lanes	
Environmental Conditions	Characterize the current environmental conditions	Weather	Weather time	Windshield washer
			Weather visibility	Fog Light (conditions of poor visibility due to rain, fog, dust or snow)
				Ice warning indicator on the board
			Car defrosters	Defrosters button in dashboard
		Temperature	-	Outside temperature display (center control)
State of the Road	Characteristics of the current road conditions	Light conditions	-	Headlights
		Surface quality	Shock absorber	Wear of shock absorbers
			Tires	Tire wear
			Vehicle Speed	Speedometer
			Brake	Brake lights
			Vehicle steering system	Steering wheel turns
Traffic Characteristics	Aspects related to traffic	Traffic density	-	GPS

Table 4. Environment Properties

ii) Internal state perception: vehicle

Descriptor	Description	Variable	Field	Source
Class of the Vehicle	Describes the category of the vehicle	Type of vehicle	-	Vehicle tab
Control Actions on the Vehicle	Types of mechanisms in the vehicle	Vehicle speed	-	Speedometer
		Hand brake	-	Brake indicator on the board
		Fuel status	-	Fuel light on the board
		Cruise control speed	-	Cruise light on the board
		Seat belt use	-	Seat belt warning light (Dashboard)
		State of doors	-	Door light on the board
		Brake	-	Brake lights
Vehicle Conditions	Defines the current conditions of the vehicle	Tires state	Tire pressure	Tire-pressure gauge
		Brake pads	-	Brake pads indicator on the board
		Airbag status	-	Airbag light on the board
		Car battery charging system	-	Car engine indicator on the board
		Car battery	-	Car battery indicator on the board
		Coolant temperature (engine)	-	Coolant temperature light on the board
		Engine oil pressure	-	Engine oil pressure light on the board
		Operation of the steering system	-	Steering indicator on the board
		ABS (Antiblockiersystem) system	-	ABS light on the board
		Service or maintenance (oil change, filter change…)	-	Service indicator on the board
		Fails at some exterior light	-	Headlight indicator on the board

Table 5. Vehicle Properties

iii) Internal state perception: driver

Descriptor	Description	Variable	Field	Source
Driver Identifier	Describe the basic Characteristics of the driver	Age	-	Personal file
		Gender		
		Reported accidents		
		Reported violations		
		Physical limitation		
		Driving experience		
Driver Attention	Define the behavior of the vehicle driver	Seat belt use	-	Seat belt warning light (Dashboard)
		Use horn	-	Touch sensor
		Hands on the wheel	-	Preprocess
		Direction of look	-	Preprocess
Physiological behavior of the driver	Define the driver's current physiological conditions	Blood alcohol limit	-	breath sensor/ infra-red light
		Blood pressure	-	Blood pressure sensor
		Heart rate	-	ECG (electrocardio-graphy) sensor
		Body temperature	-	infrared temperature sensors
Emotions of the driver	Characterize the current emotion of the driver	Facial expressions of the driver	-	Preprocess
		Voice expressions of the driver		

Table 6. Driver Properties

Appendix 2. Specification of the vehicle system

1. Preliminary Requirements

- Functional system requirements
  A. Generation of correct and opportune actions adapted to the driving context observed by the system.
  B. Dynamic adaptation of the system to atypical or unforeseen situations due to variable behavior of the driver and the driving environment.
  C. Reasoning of the system in real time, must be able to quickly indicate any condition in the operation of the system, and in addition said time must be considerable for the driver.
  D. Evaluation of the competitiveness and capacity of the elements of the system for the management of a specific situation.
  E. Guarantee of the proper functioning of the system most of the time until the intervention of the driver, without tolerance to poor adaptation to situations of risk.
  F. Generation of informative and driver-oriented notifications.
  G. Adjustment capacity before the appearance and / or disappearance of information sources, sensors and actuators, during the operation of the system.

- Non-functional system requirements
  A. Necessary participation or minimal human intervention.
  B. Arrangement of the set of necessary data sources, in this case sensors and actuators, for carrying out learning tasks.

2. Final requirements

- Determine entities
  - Active entities: *the drivers* that can act on the effectors, have an autonomous, dynamic behavior and interact with the system. *The vehicles* that perform reasoning tasks on the perceived data, can act on the effectors, have an autonomous, dynamic and evolutive behavior.
  - Pasive entities: *the devices or resources* composed of sensors that perceive and transmit the data of the internal and external environment without acting on them. *The effectors or actuators* that change their value under the action of the user or the system.

- Define Context
  - Possible interactions: The active and passive entities in the environment interact with the system. In particular:
    - The sensors send the perceived data to the system.
    - Drivers and / or vehicles can change the state of the effectors.
    - The system notifies the driver about the unusual state of the devices, and any change of value outside the normal in the effectors made by the vehicle or by the same.
    - The system informs the driver about the general state of the driving process.

- Characterize environment
  - Inaccessible: the driving process contains many variables, so it is impossible to expect to perceive the entire system, the information is subject to delays, noise or disappearance; nevertheless, a set of basic or sufficient data is established that allow modeling the driving system in a precise manner.
  - Continuous: thanks to the idea of adjustment according to the context, which implies that the system is quite dynamic, makes that the number of states in which it can be is not limited; even if it were to define a certain number of abnormal situations.
  - Non-deterministic: an action does not have a single guaranteed effect.
  - Dynamic: the environment in which the driving process develops is constantly evolving, changing over time, according to the actions of the driver and the interaction with the environment.

Bibliography

1. National Highway Traffic Safety Administration. (2015). "Preliminary Statement of Policy Concerning Automated Vehicles", *connectedautomateddriving.eu*. Website: http://vra-net.eu/wiki/index.php?title=Automationlevels
2. Trommer, S.; Kolarova, V.; Fraedrich, E.; Kröger, L.; Kickhöfer, B.; Kuhnimhof, T.; Lenz, B.; Phleps, P. (2016). "Autonomous Driving—The Impact of Vehicle Automation on Mobility Behaviour", *ifmo.*
3. National Highway Traffic Safety Administration. "Automated Vehicles for Safety", *U.S. Department of Transportation.* Website: https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety
4. Slawiñski, E.; Mut, V.; Penizzotto, F. (2015). "Sistema de alerta al conductor basado en retroalimentación vibro-tactil". *Revista Iberoamericana de Automática e Informática industrial 12*.
5. Lisetti, C. L.; Nasoz, F. (2005). "Affective intelligent car interfaces with emotion recognition". *11th International Conference on Human Computer Interaction*.
6. (2015). "Que es y cómo funciona un coche automático". *autoBild.es.* Website: https://www.autobild.es/contenido-patrocinado/especial-toyota-que-es-como-funciona-coche-autonomo-262119
7. Litman, T. (2018). "Autonomous Vehicle Implementation Predictions", *Victoria Transport Policy Institute.*
8. Papa, E.; Ferreira, A (2018), "Sustainable Accessibility and the Implementation of Automated Vehicles: Identifying Critical Decisions," *Urban Science*, vol. 2, no. 1.
9. Ohn-Bar, E.; Manubhai, M. (2016). "Looking at Humans in the Age of Self-Driving and Highly Automated Vehile", *IEEE Transactions on Intelligent Vehicls,* vol. 1, no.1.
10. Eyben, F.; Wollmer, M.; Poitschke, T.; Schuller, B.; Blaschke, C.; Farber, B.; Nguyen-Thien, N. (2010). "Emotion on the Road—Necessity, Acceptance, and Feasibility of Affective Computing in the Car", *Advances in Human-Computer Interaction,* Volume 2010, Article ID 263593.
11. Konar, A.; Chakraborty, A. (2015). "Emotion Recognition. A Pattern Analysis Approach", *John Wiley & Sons, Inc.*
12. Li, W.; Sadigh, D.; Shankar, S.; Seshia, S. (2013). "Synthesis for Human-in-the-Loop Control Systems", *Technical Report no. UCB/EECS-2013-134.* Website: http://www.eecs.berkeley.edu/Pubs/TechRpts/2013/EECS-2013-134.html
13. Pantic, M.; Sebe, N.; Cohn, J.; Huang, T. (2005). "Affective Multimodal Human-Computer Interaction", *MM'05.*
14. Aguilar, J.; Aguilar, K.; Chavez, D.; Cordero, J.; Puerto, E. (2017). "Different Intelligent Approaches for Modeling the Style of Car Driving". *14th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*.
15. Beggiato, M.; Schleinitz, K; Krems, J.; Othersen, I. (2015). "What would drivers like to know during automated driving? Information needs at different levels of automation", *7th conference on driver assistance.*
16. Guespin -Michel, J. (2015). "Emancipation et pensée du complexe". *Editions du croquant*.
17. Aguilar, J. "Curso de Inteligencia Artificial Unidad VI: Aprendizaje", *Universidad de los Andes.* Website:http://www.ing.ula.ve/~aguilar/actividad-docente/IA/transparencias/Capt6_dian.pdf
18. Nigon, J. (2017). "Apprentissage artificiel adapté aux systèmes complexes par auto-organisation coopérative de systèmes multi-agents", *Université Paul Sabatier de Toulouse III.*

19. Picard, G.; Bernon, C.; Camps, V.; Gleizes, M-P. "ADELFE: Atelier de développement de logiciels a fonctionnalité émergente", *IRIT (Université Toulouse III) and L3I (Université de La Rochelle).*
20. Guivarch, V. (2014). "Prise en compte de la dynamique du contexte pour les systèmes ambiants par systèmes multi-agents adaptatifs", *Université Paul Sabatier de Toulouse III.*
21. Georgé J-P., Gleizes, M-P., Camps, V. (2011) Cooperation. In Giovanna Di Marzo SERUGENDO, Marie-Pierre GLEIZES et Anthony KARAGEORGOS, *Self-organising software: From natural to artificial adaptation*, page 193. Springer Science & Business Media
22. Wooldridge, M., Jennings, N. R. (1995). "Intelligent agents: Theory and practice". *The knowledge engineering review*, *10*(2), 115-152.
23. Ferber, J., & Weiss, G. (1999). "Multi-agent systems: an introduction to distributed artificial intelligence (Vol. 1)". *Addison-Wesley*.
24. Glize, P. (2001). "L'adaptation des systèmes à fonctionnalité émergente par autoorganisation coopérative". *Université Paul Sabatier, Toulouse III*.
25. Verstaevel, N., Bernon, C., Georgé J-P., Gleizes M-P. (2018). "A Self-Organized Learning Modelfor Anomalies Detection: Application to Elderly People". *12th IEEE International Conference on Self-Adaptive and Self-Organizing Systems, Trento, Italy.* (To appear next week)
26. Kaddoum, E. (2011). "Optimization under Constraints of Distributed Complex Problems using Cooperative Self-Organization". PhD thesis, *Université de Toulouse*, Toulouse, France. URL: ftp://ftp.irit.fr/IRIT/SMAC/DOCUMENTS/RAPPORTS/TheseElsyKaddoum_2011.pdf.
27. Verstaevel, N. (2016). "Self-Organization of Robotic Devices Through Demonstrations". Ph.D. thesis, *Université de Toulouse*.
28. Boes, J. (2014). "Apprentissage du contrôle de systèmes complexes par l'auto-organisation coopérative d'un système multi-agent". Ph.D. thesis, *Université de Toulouse III-Paul Sabatier*.
29. Bonnet, J. (2017). "Dynamic planning, multi-objectives and multi-criteria by a self-adaptative multi-agent system". Ph.D. thesis, *Université de Toulouse III-Paul Sabatier*.
30. Erdi, P. (2008). "Complex systems: The intellectual landscape". *Complexity Explained*.
31. Silberg, G., et al. (2016). "I see, I think, I drive (I learn)". *KPMG*.
32. Bengio, Y., Courville, A., Vincent, P. (2013). "Representation learning: A review and new perspectives", *IEEE transactions on pattern analysis and machine intelligence* 35, 8, 1798–1828.
33. Baluja, S. (1996). "Evolution of an artificial neural network based autonomous land vehicle controller," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 3, pp. 450-463.
34. Wohler, C., Anlauf, J.K. (2001). "Real-time object recognition on image sequences with the adaptable time delay neural network algorithm — applications for autonomous vehicles".
35. Drescher, G. L. (1991). "Made-up minds: à constructivist approach to artificial intelligence", *MITpress.*
36. Perroto, F. S. (2013). "A computational constructivist model as an anticipatory learning mechanism for coupled agent–environment systems", *Constructivist Foundations* 9, 1, 46–56.
37. Mazac, S. 2015. "Approche décentralisée de l'apprentissage constructiviste et modélisation multi-agent du problème d'amorçage de l'apprentissage sensorimoteur en environnement continu: application à l'intelligence ambiante". Ph.D. thesis, *Université Claude Bernard- Lyon I.*
38. Gueriau, M. (2016). "Systèmes multi-agents, auto-organisation et contrôle par apprentissage constructiviste pour la modélisation et la régulation dans les systèmes coopératifs de trafic". Ph.D. thesis, *Université de Lyon I Claude Bernard*.