Towards Intelligent Robots

P. Morignot and **E. Pollard**
INRIA Rocquencourt, Teams LifeWare & RITS, Domaine de Voluceau, B.P. 105, 78150 Le Chesnay, France
E-mail: Philippe.Morignot@inria.fr Evangeline.Pollard@inria.fr

Abstract

Artificial Intelligence (A.I.) and Robotics, as two main scientific domains related to Computer Science, both have a long history since their introduction. If these two domains were close 40 years ago (e.g., with the robot SHAKEY using the first task planner STRIPS), they seem to have followed separate paths since then, with specific academic forums for each. In this paper, we first review how A.I. is perceived by Robotics researchers (a library of efficient algorithms) and its opposite (an application domain). Facing this reality, we then advocate that a unifying notion is the one of *intelligent robots* (robots on one side and intelligence on the other), the main characteristics of intelligence for robots being to be able to survive in their environment. We propose the notion of multi-layered software architecture of robotic agents, as the key to reach this goal. By abstracting our discussion, we advocate that the key theoretical difference between the two scientific domains lays (1) in the uncertainty inherent to Robotics and not necessarily in A.I., and (2) in the difference between continuity and discreteness (and how to mix them). As an example of intelligent robot, we consider automated terrestrial vehicles (autonomous cars) from the Intelligent Transportation Systems domain (ITS), in which supervision (a Robotics notion) can be considered as an upper layer of their software architectures. Such high level common notion seems more fruitful and useful than merely improving a specific component (i.e., perception, path planning or control) by importing another A.I. algorithm.

Introduction

Artificial Intelligence (A.I.) and Robotics, as two main scientific domains related to Computer Science, both have a long history since their introduction, which can be traced back over the centuries for their initial ideas. Despite decades of research and engineering work in each community, the definition of each domain seems difficult to phrase, in order to cover the vast variety of work performed under each banner.

Let us attempt at using one definition of A.I.: Is relevant to A.I. any computer program which would be said "intelligent" if the same observed behavior would be so qualified when performed by a human. This wording leads to the imitation game, or Turing test, in which a human has to determine to whom it communicates with through a computer interface: a machine or a human (Turing 1950).

On the other hand, let us attempt at using one definition of Robotics: Robotics is the science of perceiving and manipulating the physical world through computer-controlled mechanical devices (Thrun, Burgard, and Fox 2006).

Stated as above, there seems to be a wide gap between the two scientific domains, except if we talk about *intelligent robots*, i.e., robots on one hand which would exhibit an intelligent behavior on the other hand. To push the idea to its extreme point, a test similar to the Turing's one could be defined for such intelligent robots: a human tester spends one hour with another human and one hour with an intelligent robot, in any order of appearance; And the human tester then has to determine who was the robot and who was the human. If he fails, then the robot is said to be intelligent. How far are we from this dream?

Fourty years ago (in the early 70s), these two scientific domains were closer. For example, the robot SHAKEY was a major advance both in Robotics and in A.I.: while presenting an interesting mobile robot, this work proposed the first A.I. task planner known as STRIPS (Fikes and Nilsson 1971). This started the scientific A.I. subdomain of task planning, which led 40 years later to the ICAPS conference series [1]. However nowadays, the two scientific domains have followed separate paths, leading to two different scientific communities with their own academic forums for each: General conferences in A.I. include IJCAI[2], AAAI[3] and PFIA[4]; General conferences in Robotics include ICRA[5], IROS[6] and ICARCV[7]. Similarly, in France, students must choose between M.Sc. degrees in Robotics or M.Sc. degrees in A.I., leading to two different kinds of Ph.D. work in laboratories specific to each domain[8].

[1]http://www.icaps-conference.org/
[2]http://ijcai.org/
[3]https://www.aaai.org/Conferences/AAAI/aaai14.php
[4]http://pfia2013.univ-lille1.fr/
[5]http://www.icra2014.com/
[6]http://www.iros2014.org/
[7]http://www.icarcv.org/2014/
[8]We are aware of two exceptions to this separation among French laboratories: the LAAS laboratory in Toulouse and the ONERA laboratory also in Toulouse both own teams explicitly dedicated to these two topics together.

In this paper, we first consider A.I. and Robotics separately, and pinpoint the bridges between the two scientific communities. Although separated nowadays, the two scientific communities exchange algorithms, and ideas flow from one domain to the other, and back. We then advocate that the point where the two scientific communities seem to meet is the notion of software architecture of robotic agent: intelligent robots need embedding supervision, and A.I. algorithms need embodiement. By abstracting our discussion, we propose that the main difference between the two scientific communities seems to lay (1) in th notion of uncertainty and (2) in the difference between continuity and discreteness — a problem solved at least partially by the notion of software architecture. Finally, we describe an example of intelligent robot: autonomous terrestrial vehicles, i.e., Intelligent Transportation Systems (ITS).

This paper is organized as follows: Section 2 identifies Robotics in the way it perceives A.I.; Section 3 identifies A.I. in the way it perceives Robotics. Section 4 proposes an intermediate approach which unifies the two research communities through the notion of uncertainty, software architecture of robotic agents and the difference between continuity and discreteness. Section 5 proposes an example of intelligent robot: an autonomous terrestrial vehicle, i.e., an autonomous car. Finally we sum up our contribution and propose future research directions.

A.I. as perceived from Robotics

The introduction of one of the most widely known textbooks on Robotics (Thrun, Burgard, and Fox 2006) starts this way: *"Robotics is the science of perceiving and manipulating the physical world through computer-controlled mechanical devices. Examples of successful robotic systems include mobile platforms for planetary exploration, robotics arms in assembly lines, cars that travel autonomously on highways, actuated arms that assist surgeons. Robotics systems have in common that they are are situated in the physical world, perceive their environments through sensors, and manipulate their environment through things that move."* (excerpt from (Thrun, Burgard, and Fox 2006), page 3).

In planetary exploration, the environment is static and until now, there is no interaction with intelligent agents. Concerning industrial robots, they have a limited number of degrees of freedom and the environment is closed and designed to be safe. For lane keeping system, the problem is limited to lateral and longitudinal control, which is, even though complicated, really far away from what is called "autonomous driving". And finally, for assisted surgery, the robot is entirely controlled by humans: the robot does not take any decision. From these examples, it comes that this kind of robots are not confronted with changing environment and unpredictable intelligent agents: this is not the real world. It is a sub-part of what is called robotics, which has to deal with limited resources in term of energy, computational capacity, but, which cannot be considered as intelligent.

What do we call an intelligent robot? As any intelligent entity, its ultimate goal should be to ensure its survival in its environment. Developping this idea, an intelligent robot should be able to ensure its energy independence (know how to refill its energy resources). Then, it should be able to diagnose its own state and to evaluate its perception abilities (what can it do, what can't it do?). Towards survival, according to a mission (exploration, aid to individuals, *etc*), an intelligent robot should be able to react properly to an unknown/abnormal situation. Finally, it should be able to learn from experience. This list of goals is closed in spirit to classical models in Psychology, such as (Maslow 1954): a pyramid of needs is proposed for explaining the behavior of humans along the following dimensions: physiological (most urgent), safety, affiliation, achievement and learning (least urgent). All this suggests that Robotics needs A.I. at a high level — such an intelligent robot would pass the extended Turing test (see Introduction section) on the long term.

Unfortunately, A.I. is often only considered as a library of algorithms, in which Robotics researchers dig as necessary when their algorithms are not good enough for the tasks they plan to do. Many examples can be exhibited that follow this idea: The A* algorithm, a major advance in A.I. in the late 60s, can be interpreted as a best-first search on paths, i.e., given a heuristic function, A* computes a path from a root node to a potential solution node through the current node in some state space. It has been used as a major search algorithm whenever an A.I. researcher needs to an algorithm to search through a graph of possible states. But this algorithm is also widely used in Robotics: The idea is to discretize the environment (a map), and use this algorithm as a path finder: after all, the path from the root node to a solution node can be interpreted literally as a path through states as cells on a discretized map of some environment. More precisely, an A* node is an environment cell, with the root node being the start location cell in this map and a solution node being a target location cell on this same map. Therefore the A* algorithm can trivially be used as a path planner in Robotics (finding a collision-free path on a static grid map). This is an example of Robotics researchers using an A.I. algorithm for its efficiency and properties.

Similar examples can be drawn along these lines (see section 5 in the case of Intelligent Transportation Systems).

Robotics as perceived from A.I.

The most widely adopted textbook on A.I. nowadays (Russell and Norvig 2003) only dedicates two chapters to Robotics: Chapter 24 on "Perception" and chapter 25 on "Robotics". If it surely is interesting to enlarge the field of discourse from A.I. to Robotics, this chapter organization however seems to imply that Robotics could be considered as an application domain of A.I. with A.I. being the most important topic.

Similarly, the topics list of the AAAI conference series only mentions "Robotics" as a single keyword, whereas any robotics conference opens to a dozen of topics at least, and not "Robotics" as the only one.

Another example is given by (Koenig, Likhachev, and Furcy 2004): these authors propose an improved version of the A* algorithm, and test their algorithm on a mobile robot. This is another example of A.I. work which is applied to Robotics.

As another example, the whole domain of task planning (see the ICAPS conference series) aims at building planners (not to be confused with *path*-planners in Robotics), which are computer programs capable of constructing a linear plan of actions using an initial state, goals and generic operators, all described in PDDL (Planning Domain Definition Language). A PDDL operator typically uses fluents (i.e., terms) such as *on(B,A)*, stating that block B is on block A in the blocks world domain (where the problem mostly is to find how to build towers of blocks on an infinite table). Problems arise when we consider the huge amount of wrong plans of actions which could be built by these task planners — a combinatorial explosion which leads to the mere existence of this scientific community. But now the question becomes: How does a task planner knows that block B is on block A? Perception, possibly vision algorithms, should be used here, in order to establish that there actually is a block labelled B which is stacked on another block labelled A. Such a perceptive task planner is rarely considered per se by this community, except if a task planner is embedded into a mobile robot capable of perceiving its environment (Fikes and Nilsson 1971) (Morignot et al. 2012). All the work in that scientific community mainly aims at being embedded into robotic agents at some point (e.g., spatial exploratory probes, automated terrestrial vehicles).

Middle Ground

The main points of the previous two sections can be summed up this way: On one hand, Robotics researchers import A.I. algorithms when needed by their current application. A.I. is considered as a library of algorithms into which Robotics researchers dig to improve one element of the behavior of their robot. Let's phrase this as the A.I.-as-algorithm-library approach. On the other hand, A.I. researchers propose new algorithms and apply them to robots as demonstrators. Let's phrase this as the Robotics-as-application-domain approach. How then can we merge the two previous approaches? Are A.I. researchers doomed to provide algorithms to Robotics researchers? Are Robotics researchers doomed to provide robotic platforms on which A.I. researchers can demonstrate their algorithms?

Uncertainty

In an ideal world with infinite computational resources, the entire world could be modeled with uncertainty. However, this is not possible and crucial choices must be done concerning the knowledge representation and random variables that are used. Concerning the knowledge representation, several options are offered, from classical probabilities (Thrun, Burgard, and Fox 2006) to more time consuming ones such as Dempster-Shafer representation (Murphy 1998) or Transferable Belief Model (TBM) (Nashashibi, Khammari, and Laurgeau 2008), which allows to deal with unknown and conflicted situations, and also to manage sensor reliability (Elouedi, Mellouli, and Smets 2004).

Robotics aims at dealing with uncertainty data to take the best decision as possible (even it is not always used in practice). As opposed to A.I., which often assumes that things are known, i.e. reasons at a symbolic level.

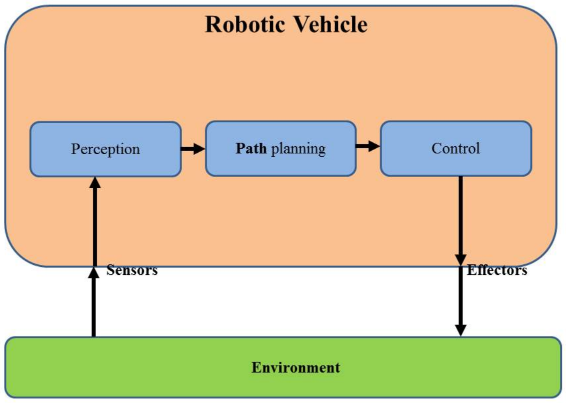

Figure 1: Software architecture of an intelligent transportation system (ITS).

Supervision through Software Architectures

The reknown chain of procedures in Robotics, e.g., automated vehicles, involves perception (i.e., turning data from sensors into symbolic percepts), path-planning (i.e., finding a collision free trajectory from a start location to a target location in an environment) and control (i.e., sending the right voltage to effectors, in order to reach a given posture) (see Fig. 1). If every previous module can incorporate A.I. algorithms (e.g., path-planning with an A* algorithm), we believe that the main lacking concept in the previous chain is supervision, as a way to reason over the previous 3 modules (e.g., providing them with parameters). This supervision layer appears as one way to bring reasoning to an automated vehicle, to take an example from the Intelligent Transportation Systems (ITS) domain.

This missing supervision layer is provided by the notion of software architecture of a robotic agent. We define a software architecture as a structure to organize the various algorithms inside a robotic agent. It should encompass very fast loops at the perception module or the control module, and potentially slow (eventually very slow) reasoning algorithms.

Many such software architectures have been proposed: Three layers (Alami et al. 1998), three layers with a sequencor (Gat 1998), two layers (Hayes-Roth et al. 1995), among others.

For example, Hayes-Roth et al. propose a two-layer software architecture for robotic agents, one layer encapsulating perception and control, and another layer encapsulating situation assessment, task planning and plan monitoring (Hayes-Roth et al. 1995) (see Fig. 2). One main characteristics of this software architecture is that the reasoning layer runs in parallel with the acting layer. This way, a potentially slow (eventually very slow) reasoning module such as task planning can finish its computation while the robotic agent is performing its task in the environment, thus preventing its motion to be blocked because of too slow reasoning compu-

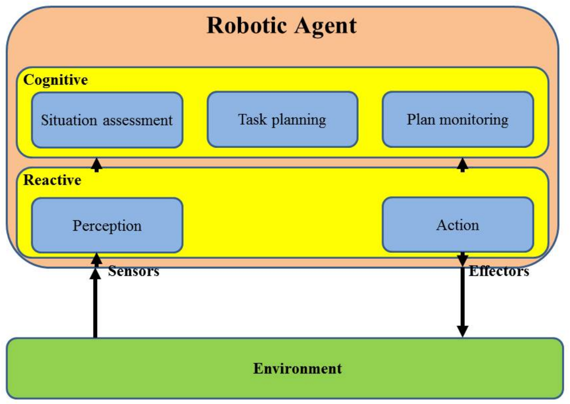

Figure 2: Two-level software architecture (Hayes-Roth et al. 1995).

tation.

In the robotics subdomain of ITS, Pollard et al. propose an intermediate approach towards full autonomy based on control layers, and self-assessment of the sensor states, both being based on dedicated ontologies (Pollard, Morignot, and Nashashibi 2013). This way, every algorithm running on an automated vehicle can be described by a concept of such ontology, and reasoning over these concepts can take place (through SWRL inference rules, in this example) — a minimal reasoning encoding supervision is hence represented with a minimal computation cost.

Continuity vs. discreteness

The previous discussion on uncertainty, a first class concept in Robotics, leads to the notion of probability, as a way to represent uncertain knowledge and reason on errors and noisy data. As such, a probability is a real number between 0 and 1, i.e., based on the set of real numbers R. Now reasoning for supervision typically involves symbols (e.g., for the previous ontologies), i.e., a finite subset of the set of integers N. The question then becomes: How can one merge a computation on R with a computation on N? $N \subset R$ but the opposite is false (R cannot be enumerated, see Cantor's proof). The only remaining possibility towards merging R (continuity) and N (discreteness) is to discretize an interval of real numbers, to a coarse or fine grain.

This opposition between continuity and discreteness is not specific to A.I. and Robotics. For example, in Operations Research, the simplex algortithm is used to (hopefully) find a solution to a problem expressed as a linear cost function and linear inequalities over variables. The variables of this model are expressed on the set of real numbers R. Let us call this method a reasoning over continuous variables. But if we want to switch to discrete variables, i.e., variables expressed on the set of natural numbers N, then the problem becomes NP-complete. Let us call this approach a reasoning on discrete variables. To solve this problem, this community uses a branch and bound algorithm, which launches a simplex algorithm at each node of a search tree (relaxed solution), forces some variables to become integers and launch a simplex algorithm again, while propagating the cost value from previous branches to cut on the current branch if necessary. Since variables are progressively assigned to integer values, the branch and bound algorithm converges to a discrete solution, i.e., at the end all variables are set to integers and not real numbers (we hide away the worst-case problem of variables exactly having the value 0.5, 1.5, 2.5 and so on in a relaxed solution). In that scientific domain either, as for A.I. and Robotics, merging continuous search and symbolic search is solved by adding symbolic reasoning on top of a continuous reasoning. This is analoguous to software architectures (see previous section), in which an upper layer is used for symbolic reasoning and a lower layer is used for continuous algorithms (perception and control).

Example: The ITS domain

From the proposed definition of an intelligent robot (see second section), the most advanced work in progress for intelligent robots seems to be autonomous driving, i.e., intelligent terrestrial vehicles in the ITS scientific domain. An autonomous car is a very complex robot (see Fig. 3, refining Fig. 1) driving in a urban jungle. It is equipped with many sensors: Proprioceptive sensors (acceloremeter, gyrometer, odometers, *etc.*) provide information about the vehicle itself such as its velocity or lateral acceleration. On the other hand, exteroceptive sensors, such as video camera, laser or GPS devices, provide information about the environment surrounding the vehicle or its location. In addition, intelligent vehicles are connected to the other vehicles as well as with the infrastructure through communication: V2V (vehicle to vehicle), V2I (vehicle to infrastructure) and even Vehicle to Pedestrian. Additionally, even though really specific to driving applications, for legal reasons, it has to take into account the human will into the decision loop and to interact with the driver.

At each robotic level, A.I. can be used if it is not too time consuming for the limited robot resources. For example, Li and Nashashibi import genetic algorithms (a whole subfield of A.I.) for merging maps (Li and Nashashibi 2012): Given one grid map of some environment, how to translate/rotate a second grid map so as to maximize the number of pixels that match between these two maps? Several approaches to this problem have been proposed, but these authors import an A.I. algorithm into Robotics for proposing a new solution. These authors use their map merging algorithm to localize an automated vehicle with respect to another one (two vehicles in a row).

As another example, Pérez et al. import fuzzy logic (a strong contribution to A.I.) for controlling an automated vehicle (Pérez et al. 2011): A set of fuzzy rules defines the behavior of the automated car in many situations, and fuzzy logic is used to represent and activate these fuzzy rules in order for the automated vehicle to exhibit a behavior regarding the given goal location and obstacles on the way. These authors use their approach to make an automated car autonomously drive smoothly on a roundabout, for example.

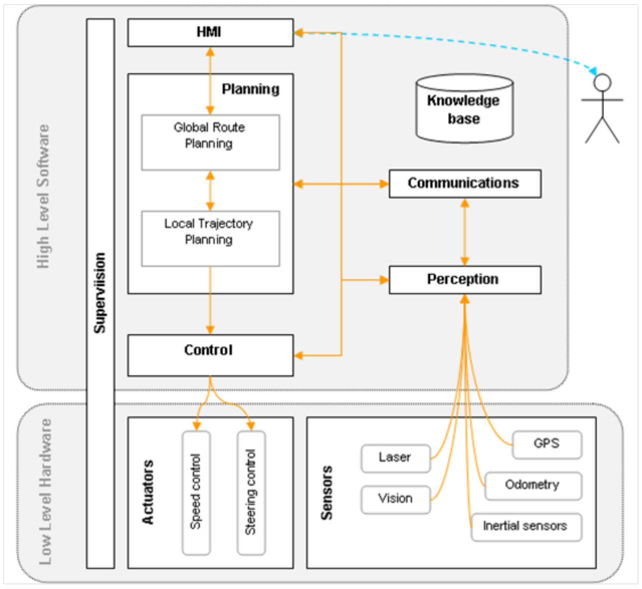

Figure 3: Automation loop for ITS.

Through these examples, it appears that A.I. techniques can be relevant to solve low level robotics problem concerning one element of the perception, planning or control loop. However, our claim is that A.I. techniques can be used at a higher level, which is called ”Supervision” (see Fig. 3) in order to bring intelligence into the decision process. For example, reaching a full autonomous driving mode in all situations seems difficult in practice due to weather conditions and sensor limitations for instance. An autonomous car, in addition to perceiving its environment, should also self-assess its own perception abilities, in order to give the control partially back to the driver or to stop if safety is not guaranty due to sensor/condition limitations. In (Pollard, Morignot, and Nashashibi 2013), two ontologies are used: one to represent the automation spectrum and the other to define situation assessment. Then, inference rules are used (integrating the uncertainty on the considered situation element estimation) in order to determine the maximum automation level allowed by the system abilities. This symbolic reasoning aims at what we call ”intelligence” or a behavior which will allow the robot to better survive in its environment.

Recalling the definition given in Section 2, some other works related to A.I. supervision layer can be cited. Among them, in (Armand, Filliat, and Ibanez-Guzman 2013), Armand et al. propose to model stop intersections with Gaussian processes, which can be seen as a part of learning from experience. However, it is limited to the driver behavior and no work is proposed on the hazardousness of trajectories and the reaction to abnormal/unknown situations is still crucially missing.

Conclusion

In this paper, we advocate that A.I. and Robotics meet on the construction of intelligent robots, to which we propose a definition as having a long term goal of survival in its environment. For this, we review the way A.I. is considered by Robotics researchers (a library of efficient algorithms) and the way Robotics is considered by A.I. researchers (an application domain). Facing this reality, we propose that the two scientific communities meet on the notion of software architecture of intelligent agents, i.e., both win by considering the other domain at a high level, and not only as a library of algorithms for specific components or as an application domain for specific algorithms. Then we focus on a scientific domain, Intelligent Transportation Systems, for which survival of a vehicle in a urban environment is a goal of sharp interest (risk of killing the car’s passengers). In this domain, we advocate that supervision, as an upper layer in a software architecture, is crucial towards building intelligent robots — and not only improving each component of the vehicle separately.

Future work involves merging constraint programming, as one paradigm for reasoning, and Robotics, in a way close to merging evolutionnary algorithms and Robotics (Doncieux et al. 2011).

Acknowledgments

The authors thank the members of the RITS project-team at INRIA Rocquencourt.

References

Alami, R.; Chatila, R.; Fleury, S.; Ghallab, M.; and Ingrand, F. 1998. An architecture for autonomy. *International Journal of Robotics Research (Special Issue on “Integrated Architectures for Robot Control and Programming”)* 17(4).

Armand, A.; Filliat, D.; and Ibanez-Guzman, J. 2013. Modelling stop intersection approaches using gaussian processes. In *16th International IEEE Conference on Intelligent Transportation System - ITSC*.

Doncieux, S.; Mouret, J.-B.; Bredeche, N.; and Padois, V. 2011. Evolutionnary robotics: Exploring new horizons. In *Studies in Computational Intelligence, New Horizons in Evolutionnary Robotics*, volume 341, 3–25. Springer.

Elouedi, Z.; Mellouli, K.; and Smets, P. 2004. Assessing sensor reliability for multisensor data fusion within the transferable belief model. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 34(1):782–787.

Fikes, R., and Nilsson, N. 1971. Strips: A new approach to the application of theorem proving. *Artificial Intelligence* 2:189–208.

Gat, E. 1998. Three-layer architecture. In et al., D. K., ed., *A.I. and Mobile Robots*. AAAI Press.

Hayes-Roth, B.; Pfleger, K.; Morignot, P.; and Lalanda, P. 1995. Plans and behavior in intelligent agents. In *Stanford University, Knowledge Systems Laboratory, Tech. Report*.

Koenig, S.; Likhachev, M.; and Furcy, D. 2004. Lifelong planning a*. *Artificial Intelligence* 155(1-2):93–146.

Li, H., and Nashashibi, F. 2012. A new method for occupancy grid maps merging: Application to multi-vehicle cooperative local mapping and moving object detection in outdoor environment. In *12th International Conference on Control, Automation, Robotics and Vision (ICARCV'12)*.

Maslow, A. 1954. *Motivation and Personality*. New York: Harper's and Row.

Morignot, P.; Soury, M.; Leroux, C.; Vorobieva, H.; and Hède, P. 2012. Generating scenarios for a mobile robot with an arm. case study: Assistance to handicapped persons. In *Eleventh International Conference on Control, Automation, Robotics and Vision (ICARCV'12)*.

Murphy, R. 1998. Dempster-shafer theory for sensor fusion in autonomous mobile robots. *Robotics and Automation, IEEE Transactions on* 14(2):197–206.

Nashashibi, F.; Khammari, A.; and Laurgeau, C. 2008. Vehicle recognition and tracking using a generic multisensor and multialgorithm fusion approach. *International Journal of Vehicle Autonomous Systems* 6(1):134–154.

Pérez, J.; Milanes, V.; Onieva, E.; and Godoy, J. 2011. Longitudinal fuzzy control for autonomous overtaking. In *IEEE International Conference on Mechatronics (ICM'11)*, 15–21.

Pollard, E.; Morignot, P.; and Nashashibi, F. 2013. An ontology-based model to determine the automation level of an automated vehicle for co-driving. In *16th International Conference on Information Fusion (FUSION'13)*.

Russell, S., and Norvig, P. 2003. *Artificial Intelligence: A Modern Approach (2nd ed.)*. Upper Saddle River, New Jersey: Pearson Education Inc.

Thrun, S.; Burgard, W.; and Fox, D. 2006. *Probabilistics Robotics*. Cambridge, Massachusetts: MIT Press.

Turing, A. 1950. Computing machinery and intelligence. *Mind* 59(236):433–450.