Artif Intell Rev
https://doi.org/10.1007/s10462-018-9631-5

Artificial intelligence test: a case study of intelligent vehicles

Li Li[1] · Yi-Lun Lin[2,5] · Nan-Ning Zheng[3] · Fei-Yue Wang[2,5] · Yuehu Liu[3] · Dongpu Cao[4,6] · Kunfeng Wang[2] · Wu-Ling Huang[2]

Abstract To meet the urgent requirement of reliable artificial intelligence applications, we discuss the tight link between artificial intelligence and intelligence test in this paper. We highlight the role of tasks in intelligence test for all kinds of artificial intelligence. We explain the necessity and difficulty of describing tasks for intelligence test, checking all the tasks that may encounter in intelligence test, designing simulation-based test, and setting appropriate test performance evaluation indices. As an example, we present how to design reliable intelligence test for intelligent vehicles. Finally, we discuss the future research directions of intelligence test.

Keywords Artificial intelligence · Intelligence test · Turing test · Simulation test

1 Introduction

Artificial intelligence (AI) usually refers to intelligence exhibited by machines. Nowadays, AI has transformed our lives in many aspects, from semi-autonomous cars on the roads to robotic vacuums in our homes. With no doubts, AI will continue to invade every area of our lives, from health care to education, entertainment to security, in the next 20 years.

✉ Fei-Yue Wang
feiyue.wang@ia.ac.cn

1 Department of Automation, BNRist, Tsinghua University, Beijing 100084, China

2 The State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

3 Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an 710049, China

4 Department of Mechanical and Mechatronics Engineering, University of Waterloo, Waterloo, ON N2L3G1, Canada

5 Qingdao Academy of Intelligent Industries, Qingdao 266109, Shandong, China

6 VIPioneers (HuiTuo) Inc., Qingdao 266109, Shandong, China

Published online: 12 April 2018

Some questions naturally arise. How to guarantee AI applications behave appropriately? Would self-driven car crash in some rare situations? Will a robotic cooker burn house? The urgent requirement of reliable artificial intelligence applications attracts increasing attention to these questions.

To answer such questions, we need to rethink what artificial intelligence is. Clearly, the definition given at the beginning of this paper is not precise. A more rigorous definition can be given as "Artificial intelligence is the intelligence (that is similar to or the same kind as human intelligence) exhibited by machines (in the same task)".

We can see that this new definition reveals the tight link between artificial intelligence and intelligence test. If and only if a machine finishes a set of specially designed tasks, we can say that this machine exhibits intelligence as human. This new definition is similar to Minsky's definition: AI is "the science of making machines capable of performing tasks that would require intelligence if done by [humans]" (Minsky 1968). The difference is that our definition focuses on the result (performing tasks); while Minsky's definition highlights the cause (the required intelligence). This definition belongs to the so-called behavior type AI definition proposed in (Russell and Norvig 2010).

Moreover, the choice of the designed tasks characterizes the kind of intelligence that this machine can have. Two sets of tasks may have no or few overlaps so that we cannot simply determine which one is more difficult. For example, an illiterate human may be a driver and a well-educated blinded human may not be able to drive.

Turing is the first researcher who realized the importance of intelligence test for developing artificial intelligence (Turing 1950). He proposed a test in which a human evaluator would judge natural language conversations between a human and a machine designed to generate human-like responses. If the evaluator cannot reliably distinguish the machine from the human, the machine is said to have finished the task and passed the test.

However, Turing test has several shortcomings and cannot be directly applied in many other applications which require reliable intelligence test for machines (Levesque 2014, 2017; Ackerman 2014; Schoenick et al. 2017). One example is intelligent vehicles that draw great attention from researchers, automobile manufacturers and the public in the last 10 years (Li and Wang 2007; Eskandarian 2012). In order to solve this problem, some initial attempts had been carried out recently (Broggi et al. 2013, 2015; Huang et al. 2014; Wagner and Koopman 2015; Li et al. 2017; Koopman and Wagner 2017; Watzenig and Horn 2017a, b; Zhao et al. 2017), but none of them give a clear portrait of the difficulties of intelligence test and explain the origins of these difficulties.

Facing such a predicament, some researchers claimed that machine-learning based autonomy is brittle and lacks 'legibility'. In contrast, more researchers believed that the field of autonomy is undergoing a machine learning revolution. They thought that the right time has come and we should combine advances in intelligent machine learning with intelligent machine testing of empirical autonomy applications.

Noticing that testing of intelligence is attracting more interests in recent studies, we survey the state-of-the-art achievements in this field in this paper. We account for the difficulties of intelligence test, highlight the role of tasks in intelligence test for all kinds of artificial intelligence, and discuss how to design reliable intelligence test for intelligent vehicles. We will not discuss the so-called strong (or hard) artificial intelligence which requires an intelligent machine to have an artificial general (full) intelligence and exhibit behavior as flexible as humans do (Ohlsson et al. 2017). Instead, we will focus on intelligence test for weak (or soft) artificial intelligence which requires an intelligent machine to solve specific problems as humans would do (Newell and Simon 1976; Kurzweil 2005). Furthermore, the recent progress in intelligent vehicles indicates that appropriate testing methods could help significantly improve the efficiency of intelligence test and thus increase the reliability of some intelligent machines. All the promising achievements urge us to put more efforts into this research field.

2 Difficulties of intelligence test

2.1 The description of tasks

The first difficulty of intelligence test is: *How to well define/describe the task for a certain kind of artificial intelligence*? A not well-defined task is usually hard to test.

The major shortcoming of Turing test is that the test is not quantitatively defined. Although Turing had inspirationally proposed a game between human and machine, there is no distinct game rule for the evaluator to determine which one will win the game. As a result, both human and machine has very little information to establish or update their strategy toward the success of the game. This certainly hinders the development of intelligent machine that can pass Turing test.

Let us review the great achievements of artificial intelligence that gained during the last 30 years. For example, IBM's Deep Blue won the chess game against human players in 1996–1997 (IBM 1997). Deepmind's AlphaGo won the Go game against human players in 2016–2017 (Silver et al. 2016, 2017b). Both chess game and Go game have much clearer definitions and descriptions than the imitation game proposed by Turing. For another example, in the celebrated ImageNet test (ImageNet 2016), we are required to just recognize the given objects but not every object. Such quantitative settings make the test focused so that we can easily track/analyze the try-and-test results and update our applications.

The comparison between Turing test and the above tests shed light on the importance of task description for intelligence test. However, it is often hard to provide a clear definition/description of the tasks that need to be finished for a certain intelligence test. Instead, many people just give a rough definition of the scenario, in which a lot of tasks needs to be finished. For example, the famous DARPA Ground Challenge 2004 requires the players to pass through a part of the Mojave Desert region of the United States but does not give a full description of the detailed task that the player may encounter (DARPA Grand Challenge and DARPA Urban Challenge 2004–2007; Buehler et al. 2009; Campbell et al. 2010). This often makes the intelligence test insufficient.

It is interesting to compare the representation systems of artificial intelligence (Srinivasan and Parthasarathi 2017; Licato and Zhang 2017) and the description of intelligence test tasks here. Both of them aim to give a 'levels of abstraction' and 'gradients of abstraction' for the problems of interests. Both of them embrace computer science terminology (e.g., typed variables, instantiated variables, ranges of valid values for variables, etc.) to describe the artificial intelligence systems that are made to reason or test. The major difference between these two concepts lies in their different focuses. The description of intelligence test cares only about the input and desired output of the system; while the representation systems of artificial intelligence are not only the input and desired output of the system, but also the way to model and describe the relationship between the input and desired output of the system (Cheng 2016).

In short, we need to first establish a set of detailed tasks for a special kind of artificial intelligence and such tasks could be quantitatively validated. Without such a basis, we cannot implement or test the intelligence exhibited by machines.

2.2 The validation of tasks

The above assumption naturally leads to the second difficulty of intelligence test: *How to guarantee that the machine acts accordingly for all the tasks that may encounter in a scenario*?

In general, we could view task validation as a decision problem that has been studied in computability (complexity) theory (Bradley and Manna 2007; Ding et al. 2013; Kroening and Strichman 2016). The input of the machine is the setting of tasks. If the machine passes a task, we assume it outputs "yes"; otherwise it outputs "no". We hope that the machine outputs "yes" for all possible inputs.

The complexity of decision problem varies significantly. Though few theoretical analysis had been made for intelligence test, we can easily find that some tasks are at least as hard as the nondeterministic polynomial time (NP) decision problems (Karp 1972). Till now, we still do not have the ranking standard to evaluate the complexity level of special kinds of artificial intelligence. We believe more and more research interests will be attracted to such a field in the near future.

For some relatively simple intelligence tests, if the scenario can be described in terms of discrete variables, we enumerate all the tasks that may occur and validate the performance of machine in each possible task. This is often troublesome and time-consuming, due to the famous combinatorial explosion problem. For example, a brute force validation reported in (Lamb 2016) had generated a 200-terabyte proof. If the scenario is described in terms of continuous variables, things may become worse, since we cannot enumerate all the combinations of variables due to their continuity.

One widely-used strategy to handle such problems is to sample the countless combinations of variables and just check the performance of the machine within these limited sampled tasks. If these representative test samples are appropriately selected, the machine which has finished all the sampled tasks is expected to behave well for all the remaining tasks, since the capability of the machine is built to be generalizable. For example, AlphaGo does not enumerate all the branches of Go game, if we view all the decision space of Go game as a decision tree. Instead, its build-in policy-network helps to filter many branches of the Go game tree and just sample a few nodes of this tree to train the machine (Silver et al. 2016, 2017b; Heule and Kullmann 2017). Competition between AlphaGo and human masters show that the policy-network based sampling strategy generally works well. However, AlphaGo still lost one game to Lee Sedol, due to incomplete training samples in 2016. The designers of AlphaGo used more samples to teach the machine to fix this problem and won all the official 60 games in 2017.

The sampling process can be guided by deterministic rules, or randomly data-driven, or even mixed. For example, researchers had proved that solving the Sudoku minimum number of clues problem is 16 via hitting set enumeration (Mcguire et al. 2014). Differently, at least partially randomly, data-driven adversarial decision-exploration and self-playing help build AlphaGo from a zero-knowledge beginner of Go game to a super Go master.

It should be pointed out that gaming is found to be a very effective task exploration tool which provides a good way to find the new samples for continuous learning and testing. Interestingly, Turing may be the first one to realize the power of gaming in artificial intelligence implementation and testing (Turing 1950). The emerging Generative Adversarial Nets (GAN) (Goodfellow et al. 2014) and the recently proposed parallel learning framework (Li et al. 2017) can all be viewed as applications of gaming based (adversarial) learning.

For some artificial intelligence applications, we will require the machine to pass all the representative tasks that will cover the whole task space. For example, we aim to test every possible extreme task an intelligent vehicle may encounter in practice (Zheng et al. 2004; Li et al. 2012, 2017; Huang et al. 2014; Wagner and Koopman 2015; Watzenig and Horn 2017a, b; Zhao et al. 2017), so as to avoid any severe accidents (A Tragic Loss 2016). However, no one can guarantee that AlphaGo will not lose a game anymore (Wang 2016a, b). How many sample tasks that are needed remains to be fathomed.

2.3 The design of simulation-based test

The desire to sample enough tasks forces us to resort to simulation-based intelligence test, since the time and financial costs of practical intelligence tests are often too high to afford. This leads to the third difficulty of intelligence test: *How to make the simulation-based test as "real" as possible*?

We could roughly categorize the simulating objects into three kinds: natural objects, man-made objects and human ourselves. Man-made objects are relatively easy to simulate because we usually know the exact math or physical disciplines that govern the behaviors of these objects. Some natural objects are difficult to simulate since they are much more complex to model. We usually introduce certain simplification and just reproduce the major features of these objects. For example, we assume that the arriving rate of vehicles follows certain distributions to test the performance of intelligent traffic control systems (Tong et al. 2015; Li et al. 2016a, b).

To mimic human behaviors is difficult. Actually, we meet a causal loop here: to test whether a machine behaves like a human, we need to set up simulation-based test; and to better simulate human that may interact with the machine, we need to well describe and simulate behaviors of human. This again requires us to judge whether the machine behaves like a human. The only possible solution to this dilemma is to build a spiral escalation process: the simulation will increase our knowledge about how to describe and simulate behaviors of human, and meanwhile, the gained knowledge helps better simulate human behaviors (Wang et al. 2016a; Li et al. 2017).

2.4 The setting of performance indices

In many applications, we have different goals of using intelligent machines. This leads to the fourth difficulty of intelligence test: *How to establish the appropriate test performance evaluation indices for tasks*?

The first kind of performance indices is to require the machine to behave like a human. A simple yet effective is to first observe how human operate in a certain task and then set up a criterion to measure how close artificial intelligent machine operations differ from human operations (Argall et al. 2009; Bagnell 2015; Kuefler et al. 2017). Therefore, the problem is transferred into finding an appropriate criterion that is able to robustly and smartly distinguish between intelligent machine operations and human operations, based on limited samples. Many researchers again resorted to the emerging Generative Adversarial Nets (GAN) (Ho and Ermon 2017; Merel et al. 2017), since we do not need to provide explicit rules to measure the difference. The implicit (dis)similarity between man-made and machine-made data will be automatically extracted and compared when GAN is correctly used. However, we have to admit that, for some applications, we still do not know how to set an appropriate quantitative criteria.

The second kind of performance indices is to reach the best performance. For example, in all chess games, we aim to build the machine that can beat all the other opponents rather than make it play like a human player. It is relatively easy to set the corresponding performance indices for such single-objective applications.

Unlike chess games in which players only aim to win, many intelligent applications have multi-objectives. For example, intelligent vehicles consider driving safety, travel speed, fuel consumption, and some other issues. Because different performance indices may lead to quite different implementations of intelligent machines, we should be very careful to set appropriate performance indices to balance different objectives.

In 2016–2017 Intelligent Vehicle Future Challenge hold in Changshu city of China, the time used by a participating vehicle to pass the given 10 tasks was taken as one of the standards of grading for intelligence level, since it is a nice synthetic criterion. Any traffic violation (e.g. running through a red light) will lead to a deduction of the final score. It is interesting that challenge participators have noticeably different preferences of the deduction values for each task. The judges had to hold a 3-h meeting to finally settle down the scoring rules.

Moreover, when the personal feeling is considered, it becomes even harder to set the appropriate performance indices. For example, personal preferences of driving may vary significantly from person to person (Classen et al. 2011; Butakov and Ioannou 2015; Lefèvre et al. 2015). To the best of our knowledge, few studies had established an accurate, flexible, and adjustable standard of grading for different personalizing aspects of driving.

3 Intelligence test for intelligent vehicles

Since it is impossible to summarize all the AI applications, we take intelligent vehicles as an example to present a framework of intelligence test and review the latest advance in this field.

3.1 The definition and generation of intelligence test tasks for vehicles

Most previous tests of intelligent vehicles did not provide a clear definition of driving intelligence. We can roughly categorize them into two kinds: scenario-based tests and functionality-based tests.

Scenario-based tests, such as DARPA Grand Challenge and DARPA Urban Challenge, just require an autonomous vehicle to pass a special region safely within a limited time (DARPA Grand Challenge and DARPA Urban Challenge 2004–2007; Buehler et al. 2009; Campbell et al. 2010). The number and the kind of traffic participants are not clearly defined. The scene and the driving environment is not explicitly given, either. This is mainly because researchers cannot enumerate all the possible settings of driving situations.

Functionality-based (ability-based) tests examine three components of driving intelligence: sensing/recognition functionality, decision functionality according to the recognized information, and action functionality with respect to the decision (Li et al. 2012, 2016a, b; Huang et al. 2014; Hernández-Orallo 2017). Special detailed functions (e.g., traffic sign recognition) will be further tested with specially designed tasks (GTSDB 2014). However, existing functionality-based tests are carried out separately and independently, which makes it impossible to get a comprehensive understanding of the intelligence level of vehicles and thus degrades the reliability of such tests.

Recently, a semantic relation diagram for driving intelligence was proposed in (Li et al. 2016a, b) to better define the intelligence of vehicles. Task atoms are on one side of this semantic relation diagram, while function atoms are on the other side of this semantic relation diagram. The links between these two sides denote that it usually requires an autonomous vehicle to perform several function atoms to fulfill any task atom. Moreover, various combinations of task atoms can be grouped into different kinds of driving scenarios. Meanwhile,

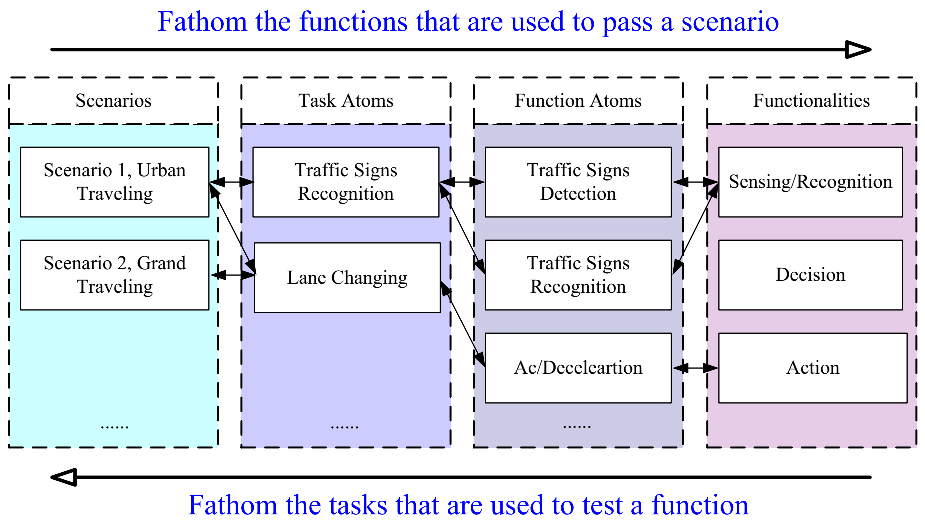
Fathom the functions that are used to pass a scenario
Scenarios
Task Atoms
Function Atoms
Functionalities
Scenario 1, Urban Traveling
Scenario 2, Grand Traveling
Traffic Signs Recognition
Lane Changing
Traffic Signs Detection
Traffic Signs Recognition
Ac/Deceleartion
Sensing/Recognition
Decision
Action
......
Fathom the tasks that are used to test a function

Fig. 1 An illustration of the semantic relation diagram for driving intelligence of autonomous vehicles

analogous to human drivers, the function atoms can also be grouped into three major categories: sensing/recognition functionality, decision functionality, and action functionality; see Fig. 1.

We can see that scenario-based tests only emphasize the left part of this semantic relation diagram; while functionality-based (ability-based) tests only emphasize the right part of it. So, this semantic relation diagram actually integrates the two major kinds of intelligent vehicle testing approaches. Moreover, if we transverse from the right side of the semantic relation diagram to the left side of the semantic diagram, we will generate the desired test task that is needed for some special functions (abilities). So, this semantic relation diagram not only defines the intelligence required to drive a vehicle but also gives the way of test task generation.

Based on this semantic relation diagram definition, a detailed test design can be simplified as a special temporal and spatial arrangement of task atoms. As shown in Fig. 2, each task can be taken as a rectangle. The left vertical boundary of this rectangle denotes the time that a task starts, and; the right vertical boundary defines the maximal allowable time when a task must be finished. The left horizontal boundary of this rectangle denotes the position that a task starts, and; the right horizontal boundary defines the maximally allowable position where a task must be finished. Since a vehicle may need to process and finish several task atoms simultaneously, the temporal-spatial range of a task may be overlapped with those of other tasks.

The number of task atoms, the difficulties of task atoms, and the numbers of concurrent task atoms all influence the difficulty of a particular task. Varying all these factors, we can sample and test tasks with different difficulty levels; see Fig. 2.

It is interesting to compare the above task definition and generation process with the so-called V-model which is frequently used for conventional automobile software development. V-model means Verification and Validation model. As shown in the right part of Fig. 3, it assumes that testing of the system is planned in parallel with a corresponding phase of development.

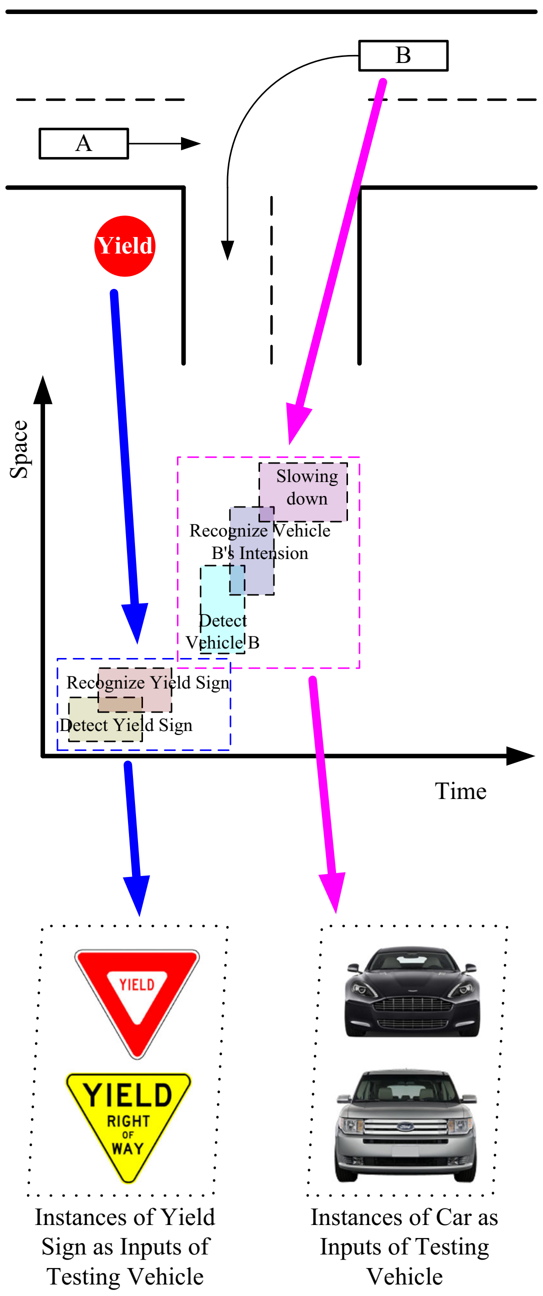
B
A
Yield
Space
Time
Slowing down
Recognize Vehicle B's Intension
Detect Vehicle B
Recognize Yield Sign
Detect Yield Sign
YIELD
YIELD RIGHT OF WAY
Instances of Yield Sign as Inputs of Testing Vehicle
Instances of Car as Inputs of Testing Vehicle

Fig. 2 An illustration of transforming a typical driving scenario into the corresponding temporal-spatial plot of the assigned tasks and generating sample instances of the related objects in simulation, according to the assigned temporal-spatial positions of tasks

The first phase of the V-model is the requirement phase which creates a system testing plan before development starts. The corresponding test plan focuses on meeting the functionality specified in the requirements gathering.

The second phase of the V-model is the high-level design phase which characterizes system architecture and design, providing an overview of the solution. Correspondingly, an

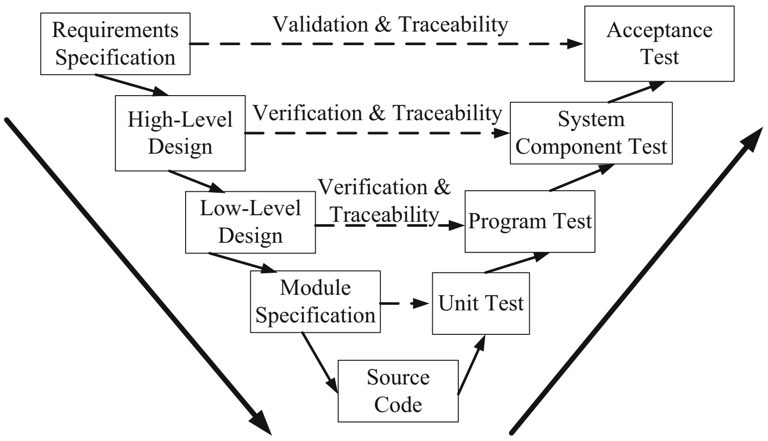

Fig. 3 An illustration of the V-model

integration test plan is created in this phase as well in order to test the pieces of the software systems ability to work together.

The third phase of the V-model is the low-level design phase which designs the actual software components, defines the operation rules for each component of the system, and sets the relationship between each designed classes. Correspondingly, component tests are created in this phase.

The fourth phase of the V-model is the module design phase which further decomposes the components into a number of software modules that can be freely combined. The bottom phase of the V-model is the coding phase where all design is converted into the code by developers. The dependences of different modules are minimized. Correspondingly, unit testing is performed by the developers on the obtained code to check the performance of modules.

If we combine the aforementioned test tasks generation method with the V-model, we can get a Λ–V-model as shown in Fig. 4. Since the definition of the up-level "scenario" is usually much more abstract than the definition of the low-level "task" and "function", we use the Greek symbol Λ to represent this top-down design. The phase-by-phase specification in the V-model is right a transverse from the left side of the semantic relation diagram to the right side of the semantic diagram.

3.2 The framework of intelligence testing system for vehicles

When test tasks are determined, we will build the testing system.

V-model is simple and easy to use for small system development where requirements can be straightforwardly understood. However, test designing happens before coding in the V-model. This makes V-model very rigid and inflexible for complex artificial intelligent system development.

As pointed out in (Boehm 1988; Raccoon 1997; Black 2009), we should take a spiral loop to find most challenging test tasks. Because learning and testing are two sides of the same coin, the architecture of such a powerful testing system should share a similar loop structure with some certain powerful artificial intelligence learning systems.

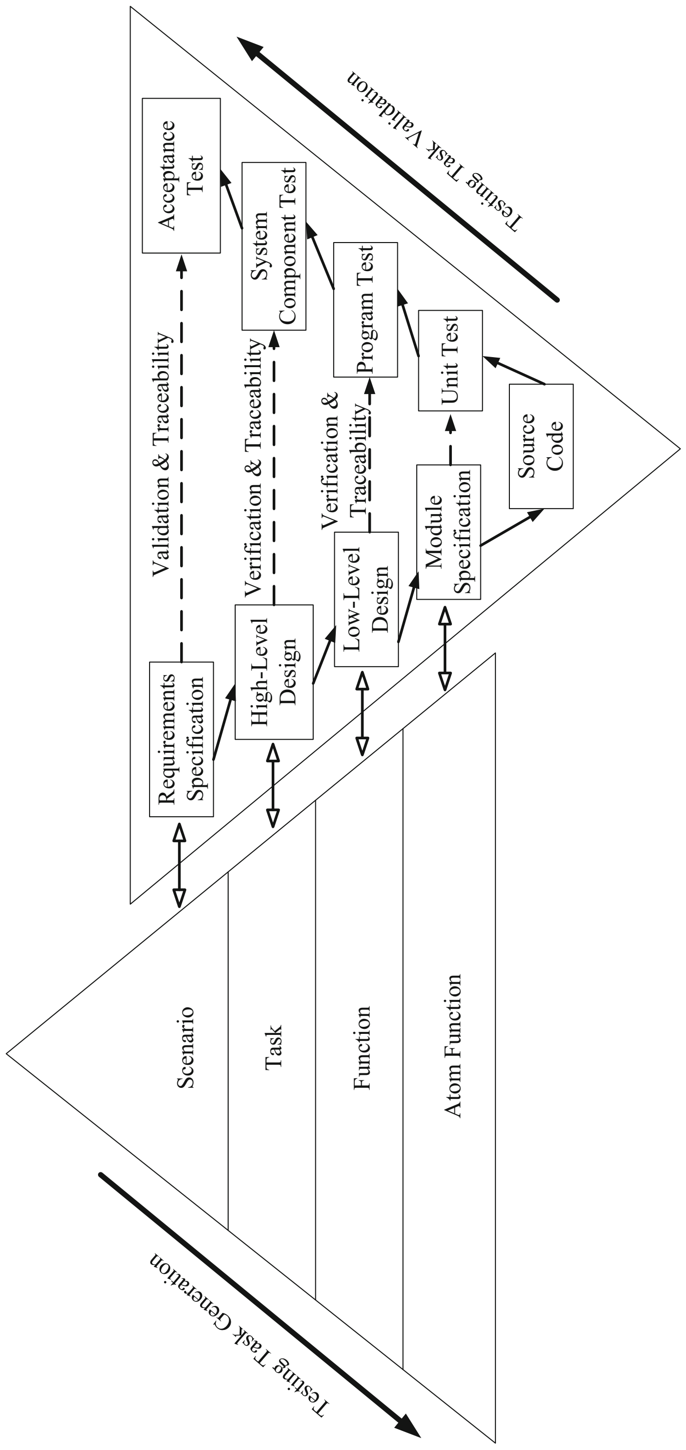

Fig. 4 An illustration of the Λ–V-model

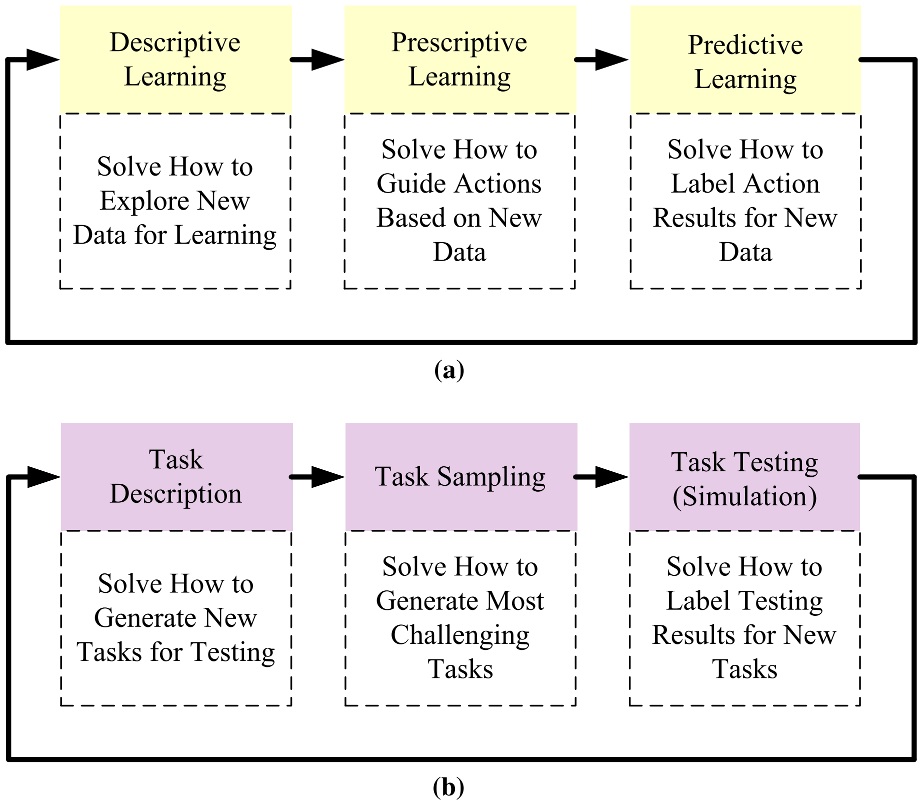
Descriptive Learning
Solve How to Explore New Data for Learning
Prescriptive Learning
Solve How to Guide Actions Based on New Data
Predictive Learning
Solve How to Label Action Results for New Data
(a)
Task Description
Solve How to Generate New Tasks for Testing
Task Sampling
Solve How to Generate Most Challenging Tasks
Task Testing (Simulation)
Solve How to Label Testing Results for New Tasks
(b)

Fig. 5 A comparison of **a** parallel learning loop (Li et al. 2017); and **b** testing loop for artificial intelligence

Let us take the recently proposed parallel learning framework (Li et al. 2017) as an example. As shown in Fig. 5a, parallel learning first applies descriptive learning to create the same (kind of) new data. This is just as Prof. Richard Feynman had said: "What I cannot create, I do not understand." Then, parallel learning applies prescriptive learning to make system evolve appropriately by special trying-and-testing and guide system with growing knowledge. Finally, parallel learning applies predictive learning to label data-action pair and leads the system to evolve in an unsupervised manner. The new action will generate new data and forms a loop in the end. The system will finally master the knowledge of choosing the appropriate actions for all the tested data. Such knowledge will be generalized to choose the actions for the untested data.

Check the inner mechanism of AlphaGo, we can find that it indeed does the same thing. The rules of Go game is first encoded (descriptive learning). The system sets up a deep neural network based policy network (prescriptive learning) to learn how to choose a move in the game (the action). The Monte Carlo sampling based self-playing (predictive learning) Browne et al. (2012) is used to determine whether the move (the action) is correct and how to update the policy network. Such a spiral loop makes the system become better and better.

Following a similar logic, an intelligent system for vehicle intelligence test explores the space of state, policy and state transitions in a loop as illustrated in Fig. 5b.

Task description part solves how to generate new tasks for testing. The main goal of this part is to set up and refine a methodology, which can guide to set up environments for the following tests. For tasks in every scenario, the descriptor will break it down into several task atoms, and then function atoms and functionalities. The connection between these elements will be described as well.

Given detailed descriptions of tasks, task sampling part will explore the policy space to choose challenging tasks. There were several ways to reach this goal (Zhao et al. 2017; Evtimov et al. 2017). However, none of the existing approaches is self-motivated.

To implement rapidly adaptive intelligence test, we consider challenging task sampling as a decision process which can be formalized as a 4-tuple (S, A, P, R). The state s_t in this decision process is the confidence we had on the performance of vehicle intelligence at time t, and the action a_t is the testing procedures that we choose to update our confidence. $\Pr_a\left(s_t, s'_{t+1}\right)$ denotes the probability that we choose a specific task will lead to another understanding level s' from state s, and the reward r_t gives how much confidence we gained at time t.

Under such setting, the long-term understanding of vehicle intelligence can be formalized as

$$V^{\pi}(s) = \mathrm{E}\left(\sum_{t=0}^{\infty} r_t | s, \pi\right). \tag{1}$$

The goal of task sampling part is to find an optimal policy π^* which can maximize the long-term understanding

$$\pi^* = \arg\max_{\pi} V^{\pi}(s). \tag{2}$$

With a detailed description of the task and sampling policy, testing (simulation) part can finally solve how to label testing results by actually generate the test scenarios and see how well the vehicle intelligence can perform. Two kinds of relationships need to be labeled during this procedure. One is the relationship between vehicle intelligence and its performance under certain environments. The evaluation of vehicle intelligence is the main output we want from an intelligent test system, and such results can help us sample better tasks in the next episode. Another is the relationship between the test and real environments. Differences of two environments and behaviors of subjects (e.g., the characteristic of traffic situations and features of vehicle dynamics) need to be paired, so the task description can be more detailed and realistic in the next loop.

The above framework of intelligence testing system for vehicles is designed based on the following considerations:

First, we can hardly know in advance whether intelligent vehicles will behave unless we test them. So, we cannot directly answer which task is most challenging. So, we need to gradually build our knowledge of testing from zero knowledge state and adopt a prescriptive learning style.

Second, testing can actually be viewed as a self-labeling (prediction learning) process. Since we do not know the outcome of a special test, we have to wait and let the results label whether the vehicle can pass the test or not.

Third, it requires huge an amount of resources and a long time to cover most of the functionalities that a vehicle intelligence should have. So, we need to find an efficient way to maximize the long-term understanding of vehicle intelligence.

We do not restrict the implementation details of such task sampling decision problem. We are now testing whether deep reinforcement learning needs to be used. We will write a dedicated paper to report the progress in the near future.

3.3 Parallel testing for vehicle intelligence test

When the detailed task is assigned, simulation-based tests can then be applied for tests of intelligent vehicles. Researchers began to show interests in accurately reproducing human behaviors (Wang et al. 2017b). While, currently, most efforts had been put into generating virtual image/video data as inputs of intelligent vehicles, since most information is collected by visual sensors (Gaidon et al. 2016; Santana and Hotz 2016; Liu et al. 2017).

Some approaches first accepted real 2D image/video data, then built the corresponding 3D object models in rendering engines, and finally generated 2D virtual image/video data as sensing inputs of intelligent vehicles (Gaidon et al. 2016; Richter et al. 2016; Greengard 2017). Some other approaches directly employed GAN to generate new virtual 2D image/video data from existing real 2D image/video data (Santana and Hotz 2016; Gatys et al. 2016; Liu et al. 2017). The latest approach mixed these two methods to produce more virtual data as "real" as possible and as "rich" as possible (Veeravasarapu et al. 2015; Wang et al. 2017a; Ros et al. 2016).

In this subsection, we propose a parallel system framework that combines real-world and simulation-world for vehicle intelligence test. As illustrated in Fig. 6, a vehicle intelligence test can be decomposed into three parts, the environment, the test planning part, and the test performing part. Following the logic we predicated in the last subsection, a parallel system can be built by connecting these three parts.

The loop of intelligence test in the parallel system starts from a real environment, which is an area with intersections, traffic signs and other elements of some specific driving scenarios. Depending on the mission, a task description, which is a directed acyclic graph (DAG) can first be initialized according to some prior knowledge. It breaks down the task into task atoms, function atoms, and functionalities atoms. Then, it establishes the connection between these atoms. The weights of DAG are estimations of confidence gained by performing a certain step.

Based on the description, an agent will be trained to plan the best schedule of tasks. For example, if there are two task atoms, traffic signs recognition and lane changing, the optimal agent will find that, the traffic signs recognition atoms can actually be neglected, since most of the confidences can be gained by performing the lane changing atom. Weighing the pros and cons of different routes in the DAG, the agent prunes some routes and picks important ones to perform. The most important tasks will be checked in the real environments and the less important ones will be tested in simulation.

Once the schedule is provided, a special task can be tested. Depend on the confidence of test accuracy and the importance of atom, we can calculate a weighted score based on the results in both real and simulative environments. Meanwhile, data generated in the real environment will be fed into the simulative environment, so the simulation can be improved continuously. The loop in the real system and the artificial system is asynchronous, and multiple loops can be performed in the artificial system while one loop in the real environment.

Comparing to traditional simulative environments, the parallel system for vehicle intelligence test has two main differences. First of all, the parallel system is not merely a reflection of the real system, but a combination of two systems with equal status. Things happened in both systems will affect each other and form a closed self-boosting loop. Second, the parallel system is a learning system which can evolve over time. Several key components in the

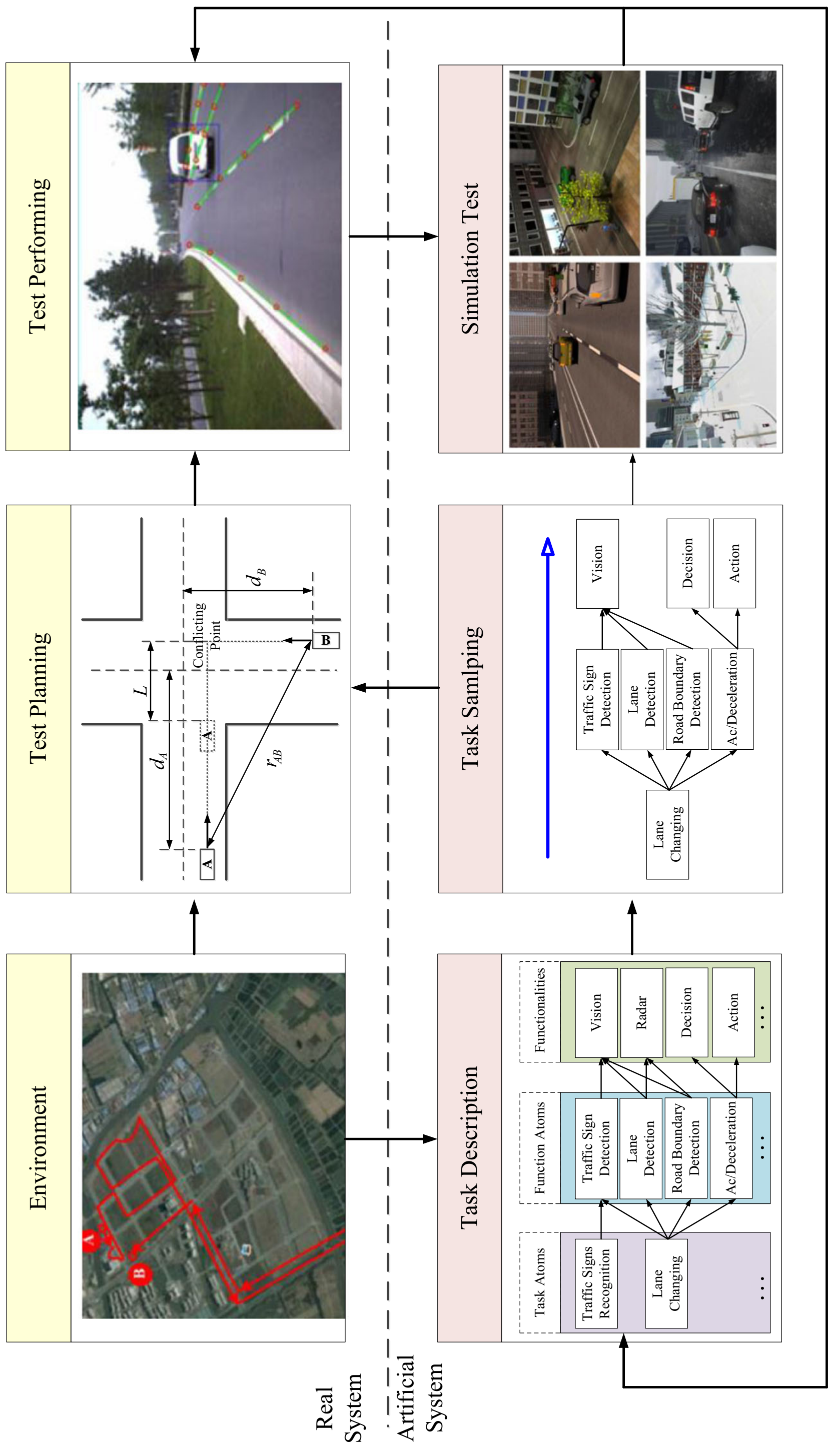

Fig. 6 A demonstration of parallel system for vehicle intelligence test

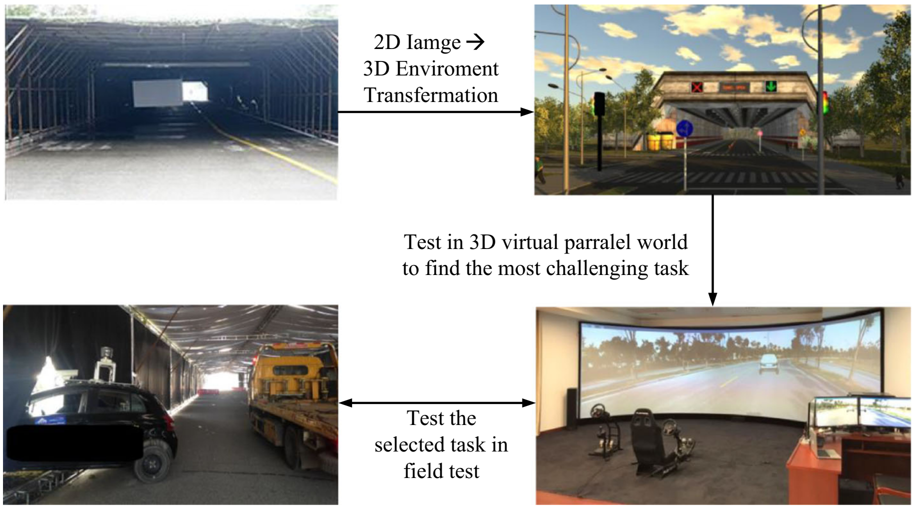

Fig. 7 A demonstration of using parallel system to find the most challenging task

artificial system (e.g., the task sampling agent and simulative environment) are data-driven instead of arbitrary models. Such designs make the parallel system more autonomous and quantifiable.

It should be pointed out that a prototype parallel intelligence testing system had already been built in Changshu city, Jiangsu Province, China and had successfully supported the 2016 and 2017 Intelligent Vehicle Future Challenge (IVFC). As shown in Fig. 7, some testing vehicles passed a number of relatively simple tasks but failed to do so when encountering the most challenging task that had been found in virtual tests in the virtual parallel world.

4 Discussions

4.1 Ethical problems

Most researchers, starting from Turing, have implicitly assumed that human will do the right things to finish the studied tasks and intelligent machines should learn to do the same right thing to finish the studied tasks. So, we only need to check whether intelligent machines do the same things as human, during intelligence test.

However, in some cases, even a human will feel difficult to know what should be done. One famous case is the so-called trolley problem that has mulled for about 50 years. Suppose a runaway trolley speeding down a track to which five people are tied. You can pull a lever to switch the trolley to another track to which only one person is tied. Would you sacrifice the one person to save the other five, or let the trolley kill the five people?

Trolley problems caused much debate that we do not want to discuss in this paper. If we think of humans as moral decision-makers and take artificial intelligent machines as moral agents that actually replace our capacities, we can hardly find a commonly accepted answer (Goodall 2014; Kumfer and Burgess 2015; Maurer et al. 2015; Thornton et al. 2017). If we assume that intelligent machines reason and act just what human had told them to do, the only decision-makers are human but not intelligent machines. In this paper, all such problems involved ethical decision making are not considered. As a result, we do not discuss how to design any intelligence test tasks for ethics, since we should pay to Caesar what belongs to Caesar–and God what belongs to God.

4.2 Real-time and automated evaluation of testing results

One major difference between Turing test and the new approach of intelligence test is the selection of the judge. Turing chose human to be the judge to arbitrate whether a machine has intelligence in Turing test; while many new intelligence testing systems use machines to arbitrate. This is not only because we have a more clear description of tasks in many recently studied intelligence test problems, but also because a human is unable to accurately examine many results of intelligence test without the help of machines.

Let us still use testing for intelligent vehicles as an example. To save time and money, several independent tasks of an intelligent vehicle are often linked along a special path of the vehicle and are tested sequentially in practice. For instance, a vehicle needs to finish 14 tasks in 2017 Intelligent Vehicle Future Challenge, including: (1) make U-turn, (2) pass the signalized T-intersection, (3) pass the non-signalized cross-intersection, (4) pass other vehicles, (5) pass the tunnel in which GPS is blocked, (6) recognize the stop sign dedicated for vehicles and behave appropriately, (7) pass another stop sign dedicated to school children, (8) give way to pedestrian, (9) make a right-turn, (10) pass the rural road, (11) give way to bicycle, (12) pass the working zone, (13) recognize the speed limit and behave appropriately, (14) park into the assigned berth; see Fig. 8 for an illustration.

Usually, we do not require the vehicle to stop after it passes a task. In order to achieve a real-time and automated evaluation of the testing results for each individual task, researchers

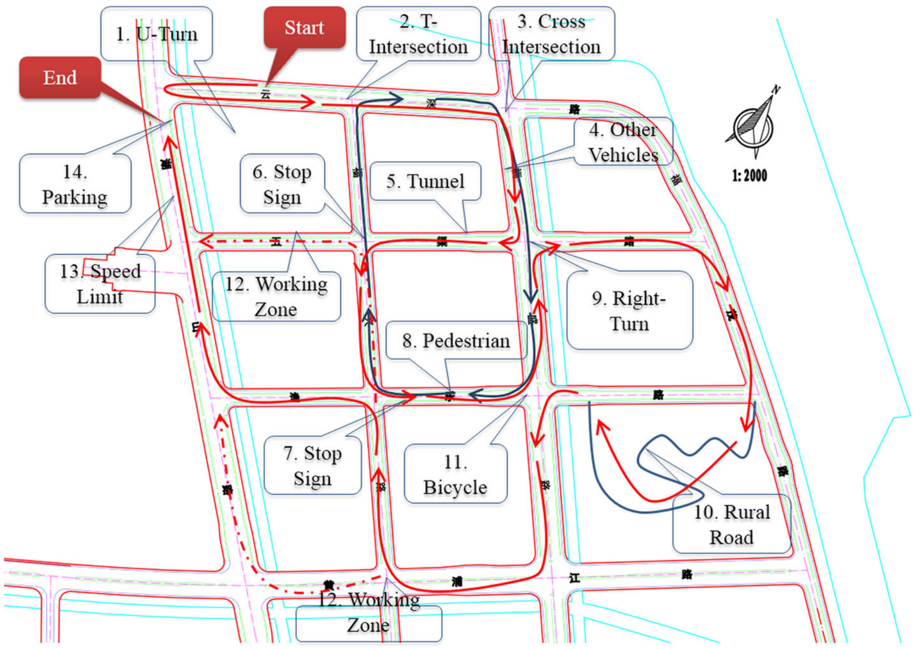

Fig. 8 An illustration of different test tasks for 2017 intelligent vehicle future challenge

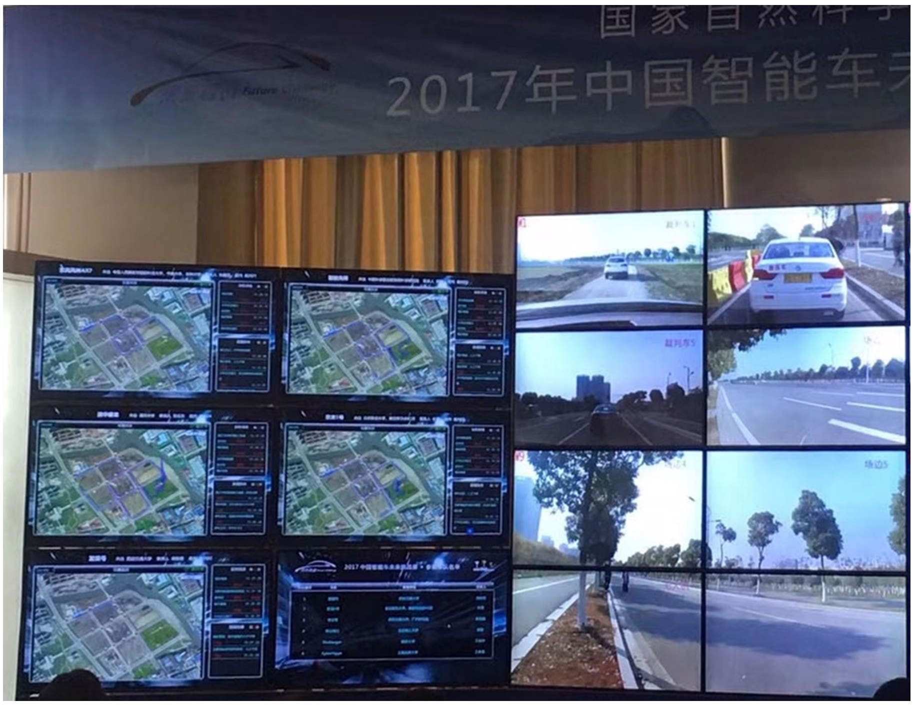

Fig. 9 A demonstration of the real-time automated evaluation system designed for vehicle intelligence tests of 2017 intelligent vehicle future challenge (IVFC)

had used vehicle-to-everything (V2X) communications to connect the onboard sensors and control center, share a number of information of vehicle (e.g., position, speed, ac/deceleration rate) and rapidly calculate the performance values of each task based on the collected information. Such a method reduces the burden of testing and becomes increasingly popular.

Figure 9 gives a demonstration of the evaluation system designed by Tsinghua University and Qingdao *VIPioneers* company, for 2017 Intelligent Vehicle Future Challenge. The left screens show the real-time trajectories of 5 vehicles that were running in the Challenge and their ranks. The right screens show the real-time monitoring video data collected from the cameras that were installed inside the tested vehicles, the cameras that were installed inside the following arbitrator vehicles, and the roadside cameras. All the data were transferred to the testing center via various ways, including V2X communication, 4G wireless communication, and optical fiber communication.

In 2009–2015 Intelligent Vehicle Future Challenges, human judges determine how to evaluate the performance of intelligent vehicles. Such manual evaluation is tedious, time-consuming and prone to error. In Intelligent Vehicle Future Challenge 2017, most evaluations were done by machines based on the measured data collected from various resources. Comparisons show that the evaluations became more accurate and much quicker. For example, in the previous match, human judges stared at the dashboard to check whether the tested vehicle is speeding. Based on the high-resolution position information measured via BeiDou navigation satellite system (Wang 2016a, b), we can easily reconstruct the whole trajectory of the tested vehicle and determine when and where the vehicle is speeding.

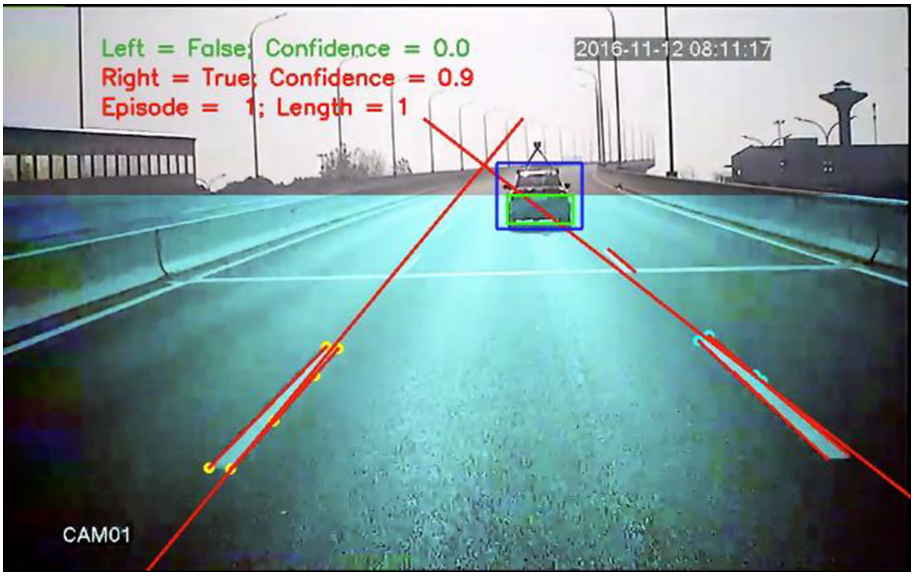
Left = False; Confidence = 0.0
Right = True; Confidence = 0.9
Episode = 1; Length = 1
2016-11-12 08:11:17
CAM01

Fig. 10 A demonstration of the automated evaluation system designed to lane departure warning

For another example, Fig. 10 gives a demonstration of the deep learning (LeCun et al. 2015; Goodfellow et al. 2016) based automated evaluation system designed to recognize whether the vehicle had crossed the lane boundaries (You 2017). This system used YOLO (Redmon et al. 2016; Redmon and Farhadi 2016) to recognize the tested vehicle, based on the video data collected from the judging vehicle that follows the tested vehicle all the way along. It can help catch each incorrect crossing of the lane boundaries during the long-time running tests and greatly relieve the burdens of human judges.

4.3 Human–machine integrated testing

However, we do not claim that we should remove human from tests of artificial intelligence. In the current stage, human participates in every aspect of artificial intelligence tests.

First, human experts are heavily involved in the description of test tasks. Indeed, every test is described by a certain kind of language that is established by human. Till now, we do not observe any artificial intelligent machine generates its own language. The capability of an intelligent machine and that of the corresponding testing system is constrained by human designers, too. So, we always resort to human experts to make substantive improvement for the design and tests of artificial intelligence.

Second, human experts also help to design the most challenging tasks in many intelligent applications, according to their experience and intuition that is gained through finishing the same tasks. For example, researchers inquired human drivers to set up different testing levels for different tasks for intelligent vehicles (Zheng et al. 2017).

Third, human experts usually monitor the testing process and take the final responsibility to guarantee that the testing results are correct. As shown in Fig. 8, the automated evaluation system designed for 2017 Intelligent Vehicle Future Challenge provides real-time visualization for human experts. This enables human experts to track the entire progress of testing, monitor whether the automated evaluation system works well, and gain an intuitive understanding of testing result. Such a hybrid-augmented intelligence (Zheng et al. 2017) setting helps combine both human and machines to better evaluate the performance of intelligent machines.

It should be pointed out that, till now, human's intelligence levels are tested via the tasks designed by human experts (Sternberg and Davidson 1983; Sternberg 1985; Mackintosh 2011; Rindermann et al. 2016; Ohlsson et al. 2017). Can we use some tasks that generated by machines via some technologies similar to what we had discussed above? We believe this interesting question will attract more attention in the near future.

4.4 Testing as a measurement of intelligence level

SAE International defines the six levels of driving automation, from no automation to full automation in 2016 (SAE J3016 2016). However, there is not a clear description of the corresponding test tasks. So, it becomes widely accepted that testing results for intelligent vehicles can be viewed as a measurement of intelligence level. Only if a vehicle passes all the tasks that are designed for a special level of driving automation, we can claim that this vehicle has such an intelligence level.

Intelligent machines are becoming smarter and smarter now. Now, intelligent machines had beaten all human players in Shogi, chess and Go games (Silver et al. 2017a, b). The AI 'Top Gun' beat the military's best pilots repeatedly. It is probably safe to say that all artificial intelligence researchers aim to design and implement some machines that beat human in certain kinds of tasks, since aeronautical engineers had shown that they can do something better than making machines fly so exactly like pigeons (Russell and Norvig 2010).

Maybe in the future, we should renew our definition of artificial intelligence as "Artificial intelligence is intelligence (that is similar to, or the same kind as, or even superior to human intelligence) exhibited by machines (in the same task)". At the current stage, human experts are still the major referring standard for tests of artificial intelligence. Sometimes in the future, the performance that an intelligent machine could achieve will serve as a new evaluating standard of intelligence level instead.

When we cannot enumerate all the test tasks, it becomes increasingly complex to set a fair measurement of intelligence for two different artificial machines dedicated for the same purpose. For example, in Go game, researchers used the Elo rating scores (Elo 1978; Coulom 2008; Silver et al. 2017b) that were computed from evaluation games between different players, because conventional static rating systems do not consider time-varying strengths of players. When the information that we can observe from the results is limited, things become even harder. As shown in the recent algorithms designed for the poker game, analyzing results indicated that we need to build special algorithms to drill the useful guide so as to boost the intelligent machines (Moravčík et al. 2017; Brown and Sandholm 2017). We believe that more research efforts will be put into this research direction.

4.5 Explainable testing of intelligent machines

It should be also pointed out that, just like Turing had done 67 years ago, we focus on the outside behaviors of human/machine rather than the inside mechanism that generates the outside behaviors. If a machine has passed all the tasks according to its outside behaviors, we admit its intelligence in this special field. However, we usually know neither what the best way to finish all these tasks is, nor how human finish these tasks.

Nowadays, intelligent algorithms and machines become more and more complex. Someone is calling them 'black box', since it becomes harder to interpret what these algorithms and machines are doing. However, intelligent machines coded in simple rules seem do not work as well as some state-of-the-art ‘black boxes’. Actually, if we assume that the latest machine learning technology has “the ability to learn from testing results and improve itself automatically without being explicitly programmed, we may find that these machines will be naturally hard to interpret. Otherwise, we can turn them back to explicit codes.

To the best of our knowledge, few studies give a widely-accepted generalizable way to combine inside mechanism design with outside behavior validation of artificial intelligence. We think this new direction may bring some interesting findings in the near future.

4.6 Testing as an essential part of artificial intelligence software development process

Because artificial intelligence is coded and implemented on computers, we need to highlight the importance of software development of artificial intelligence. The lack of reproducibility and readability has already hindered the development of AI techniques, since researchers can hardly rely on an implementation that can hardly be proofed or understood to further their research.

A proper design of AI development loop can help to alleviate such situation. Test-driven development (TDD) has already been widely adopted in modern software development process. The basic idea of TDD is to organize the development cycle as a repetition of a very short development cycle: First turn the requirements into very specific test cases, and then improve the software to pass the tests. In such development process, the reliability can be guaranteed if we set the test properly, and the readability of software can be improved as well, since it is organized as the collection of simple components to each fulfill a specific requirement.

The development of AI software can be profited from such development methodology, if some critic problems are solved. Despite the unclear definition of requirements which can be handled by the method we proposed in the last section, the major problem is the lack of testing and debugging tools. Software testing had already taken an essential part of software development. Almost all state-of-the-art commercial software developing tools provide thorough support for testing at different phases (Huizinga and Adam 2007; Ammann and Jeff 2017). However, most current software/toolbox for building artificial intelligence lacks convenient testing tools and debuggers. We wonder software/toolbox for building artificial intelligence could be viewed as Software 2.0 (Karpathy 2017). We expect more attention could be drawn to this important issue.

4.7 Life-long learning and life-long testing

Researchers are developing more and more powerful testing methods of artificial intelligence, just like what they had done for design methods of artificial intelligence. However, all the changes take time to complete. Similar to the evolutionary history of machine learning, it seems that machine testing will take a relatively long time to become strong enough to characterize what a truly intelligent machine should be. We cannot give a precise prediction of the time when an intelligent vehicle can drive in all kinds of situations. So, we borrow the term “life-long” from life-long learning (Chen and Liu 2016) and name this evolution process as “life-long testing”.

Moreover, it should be emphasized that we should always take the design and testing of intelligent vehicle as a whole. The knowledge of testing will be fed back to the design part of intelligent vehicle and will be used to further improve the intelligence of intelligent vehicles.

Such a spiral loop helps make intelligent vehicle into practice in every automobile lab and manufactory.

In precision machining industry, we continuously employ low-level machines to build more precise high-level machines. About 400 years ago, we can only make some simple gadgets. Now, we had achieved a great success and become able to make many complex things like CPU and GPU. Similarly, in artificial intelligence research field, smart machines are used to build even smarter machines now. Fortunately, we are now witnessing such a great change in artificial intelligence development.

4.8 Testing as an economical opportunity

The ongoing artificial-intelligence revolution brings changes in enormous social lives and economic opportunities (Harari 2017). Humans are pushed out of some part of the job market by intelligent machines (Fagnant and Kockelman 2015; Fisher et al. 2016). For example, some aggressive researchers advocated to totally replace human drivers in the near future.

Meanwhile, AI generates a wide range of new jobs, including some new jobs for tests of AI. Using crowdsourcing (Wang et al. 2016b), we can hire a number of human to label the video data collected in streets and plot the bounding boxes of vehicles/pedestrians, since we need ground truth data to train the artificial intelligent systems for environment recognition and autonomous driving. Several companies in China had hired a lot of retired people to do such jobs and gained gigabytes of useful in return. We hope that, in the future, many people who had been replaced by intelligent machines could join the building process of more intelligent machines. This also requires us to build more flexible and powerful software, like Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) (von Ahn et al. 2003; George et al. 2017).

Crowdsourcing also leads to new risks of AI developing and testing. Tencent company had recently announced a critical vulnerability of Google's TensorFlow. Such vulnerability allowes hackers access to AI code being written by programmers, jeopardize the training data, or confuse the testing results (Liao 2017). So, we have to make far more efforts to make distributed tests of artificial intelligence into practice.

5 Conclusions

In this paper, we discuss four major difficulties of carrying out the test of artificial intelligence, with a special emphasis on the role of task in intelligence test. We also present our experiences in designing reliable intelligence test for intelligent vehicles.

We explain our design of intelligence test by analogy with the structure of machine learning framework. The origin of this similarity lies in the fact that learning and testing are indeed two faces of artificial intelligence. From this viewpoint, we explain why a parallel system framework for vehicle intelligence test is needed. Such a framework should have two important features. First, the whole testing should be formulated as a loop between three parts: task description, task sampling and task testing (simulation). This formulation allows us to gradually build our knowledge of testing results and automatically finds the most challenging tasks to test. Second, the simulation tests should be executed in a mirror system so that we can produce more virtual data as "real" as possible and as "rich" as possible. This will help us reduce both the time and financial costs of testing.

However, the evolution of artificial intelligence only helps to reduce human participation from some parts but not the core of artificial intelligence test. We still do not have an intelligent machine can self-test, self-boost and upgrade without the help of human. The singularity of AI (Vinge 1993; Kurzweil 2005) is yet to come.

Acknowledgements This work was supported in part by the National Natural Science Foundation of China under Grants 91520301 and 61533019, and the Beijing Municipal Science and Technology Project (No. D171100000317002).

References

A Tragic Loss (2016) https://www.tesla.com/blog/tragic-loss. Accessed April 2018
Ackerman E (2014) A better test than Turing. IEEE Spectr 51(10):20–21
Ammann P, Jeff O (2017) Introduction to software testing, 2nd edn. Cambridge University Press, Cambridge
Argall BD, Chernova S, Veloso M, Browning B (2009) A survey of robot learning from demonstration. Robot Auton Syst 57(5):469–483
Bagnell JA (2015) An invitation to imitation. Technical Report, CMU-RI-TR-15-08, Robotics Institute, Carnegie Mellon University
Black R (2009) Managing the testing process: practical tools and techniques for managing hardware and software testing. Wiley, Hoboken
Boehm BW (1988) A spiral model of software development and enhancement. IEEE Comput 21(5):61–72
Bradley AR, Manna Z (2007) The calculus of computation: decision procedures with applications to verification. Springer, Berlin
Broggi A, Buzzoni M, Debattisti S, Grisleri P, Laghi MC, Medici P, Versari P (2013) Extensive tests of autonomous driving technologies. IEEE Trans Intell Transp Syst 14(3):1403–1415
Broggi A, Cerri P, Debattisti S, Laghi MC, Medici P, Molinari D, Panciroli M, Prioletti A (2015) PROUD—public road urban driverless-car test. IEEE Trans Intell Transp Syst 16(6):3508–3519
Brown N, Sandholm T (2017) Safe and nested subgame solving for imperfect-information games. https://arxiv.org/abs/1705.02955. Accessed April 2018
Browne CB, Powley E, Whitehouse D, Lucas SM, Cowling PI, Rohlfshagen P, Tavener S, Perez D, Samothrakis S, Colton S (2012) A survey of monte carlo tree search methods. IEEE Trans Comput Intell AI Games 4(1):1–43
Buehler M, Iagnemma K, Singh S (eds) (2009) The DARPA urban challenge. Springer, Berlin
Butakov VA, Ioannou P (2015) Personalized driver/vehicle lane change models for ADAS. IEEE Trans Veh Technol 64(10):4422–4431
Campbell M, Egerstedt M, How JP, Murray RM (2010) Autonomous driving in urban environments: approaches, lessons and challenges. Philos Trans R Soc A 368(1928):4649–4672
Chen Z, Liu B (2016) Lifelong machine learning. Morgan & Claypool Publishers, San Rafael
Cheng PCH (2016) What constitutes an effective representation? In: Jamnik M, Uesaka Y, Elzer Schwartz S (eds) Diagrammatic representation and inference: proceedings from the 9th international conference, diagrams 2016, vol 9781. Lecture notes in computer science. Springer, Berlin
Classen S, Nichols AL, McPeek R, Breinerd JF (2011) Personality as a predictor of driving performance: an exploratory study. Transp Res F Traffic Psychol Behav 14(5):381–389
Coulom R (2008) Whole-history rating: a Bayesian rating system for players of time-varying strength. In: Proceedings of international conference on computers and games, pp 113–124
DARPA Grand Challenge, DARPA Urban Challenge (2004–2007) http://archive.darpa.mil/grandchallenge/. Accessed April 2018
Ding Z, Jiang C, Zhou MC (2013) Design, analysis and verification of real-time systems based on time Petri net refinement. ACM Transactions in Embedded Computing Systems 12:4:1–4:18. https://doi.org/10.1145/2406336.2406340
Elo AE (1978) The rating of chessplayers, past and present. Arco Publishing, New York
Eskandarian A (ed) (2012) Handbook of intelligent vehicles. Springer, Berlin
Evtimov I, Eykholt K, Fernandes E, Kohno T, Li B, Prakash A, Rahmati A, Song D (2017) Robust physical-world attacks on machine learning models. https://arxiv.org/abs/1707.08945. Accessed April 2018
Fagnant DJ, Kockelman K (2015) Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. Transp Res A Policy Practice 77:167–181
Fisher DL, Lohrenz M, Moore D, Nadler ED, Pollard JK (2016) Humans and intelligent vehicles: the hope, the help, and the harm. IEEE Trans Intell Veh 1(1):56–67
Gaidon A, Wang Q, Cabon Y, Vig E (2016) Virtual worlds as proxy for multi-object tracking analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4340–4349

Gatys LA, Ecker AS, Bethge M (2016) Image style transfer using convolutional neural networks. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 2414–2423

George D, Lehrach W, Kansky K, Lázaro-Gredilla M, Laan C, Marthi B, Lou X, Meng Z, Liu Y, Wang H, Lavin A, Phoenix DS (2017) A generative vision model that trains with high data efficiency and breaks text-based CAPTCHAs. Science. https://doi.org/10.1126/science.aag2612

Goodall NJ (2014) Ethical decision making during automated vehicle crashes. Transp Res Rec 2424:58–65

Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. Proc Adv Neural Inf Process Syst 27:2672–2680

Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge

Greengard S (2017) Gaming machine learning. Commun ACM 60(12):14–16

GTSDB, The German Traffic Sign Recognition Benchmark and the German Traffic Sign Detection Benchmark (2014) http://benchmark.ini.rub.de/?section=home&subsection=news. Accessed April 2018

Harari YN (2017) Reboot for the AI revolution. Nature 550:324–327

Hernández-Orallo J (2017) Evaluation in artificial intelligence: from task-oriented to ability-oriented measurement 48(3):397–447

Ho J, Ermon S (2017) Generative adversarial imitation learning. https://arxiv.org/abs/1606.03476. Accessed April 2018

Huang WL, Wen D, Geng J, Zheng NN (2014) Task-specific performance evaluation of ugvs: case studies at the IFVC. IEEE Trans Intell Transp Syst 15(5):1969–1979

Huizinga D, Adam K (2007) Automated defect prevention: best practices in software management. Wiley, Hoboken

IBM, Deep Blue - Overview (1997) IBM Research. http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/. Accessed April 2018

ImageNet (2016) http://image-net.org. Accessed April 2018

Karp RM (1972) Reducibility among combinatorial problems. In: Miller RE, Thacher JW (eds) Complexity of computer computation. Plenum Press, New York, pp 85–103

Karpathy A (2017) Software 2.0. https://medium.com/@karpathy/software-2-0-a64152b37c35. Accessed April 2018

Koopman P, Wagner M (2017) Autonomous vehicle safety: an interdisciplinary challenge. IEEE Intell Transp Syst Mag 9(1):90–96

Kroening D, Strichman O (2016) Decision procedures: an algorithmic point of view, 2nd edn. Springer, Berlin

Kuefler A, Morton J, Wheeler T, Kochenderfer M (2017) Imitating driver behavior with generative adversarial networks. In: Proceedings of IEEE intelligent vehicles symposium, pp 204–211

Heule MJH, Kullmann O (2017) The science of brute force. Commun ACM 60(8):70–79

Kumfer W, Burgess R (2015) Investigation into the role of rational ethics in crashes of automated vehicles. Transp Res Rec 2489:130–136

Kurzweil R (2005) The singularity is near. Viking Press, New York

Lamb E (2016) Maths proof smashes size record: supercomputer produces a 200-terabyte proof—but is it really mathematics? Nature 534(7605):17–19

LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444

Lefèvre S, Carvalho A, Gao Y, Tseng HE, Borrellia F (2015) Driver models for personalised driving assistance. Veh Syst Dyn 53(12):1705–1720

Levesque HJ (2014) On our best behavior. Artif Intell 212:27–35

Levesque HJ (2017) Common sense, the Turing test, and the quest for real AI. MIT Press, Cambridge

Li L, Wang FY (2007) Advanced motion control and sensing for intelligent vehicles. Springer, New York

Li L, Wen D, Zheng NN, Shen LC (2012) Cognitive cars: a new frontier for ADAS research. IEEE Trans Intell Transp Syst 13(1):395–407

Li L, Huang WL, Liu Y, Zheng NN, Wang FY (2016a) Intelligence testing for autonomous vehicles: a new approach. IEEE Trans Intell Veh 1(2):158–166

Li L, Lv Y, Wang FY (2016b) Traffic signal timing via deep reinforcement learning. IEEE/CAA J Autom Sin 3(3):247–254

Li L, Lin Y, Zheng NN, Wang FY (2017) Parallel learning: a perspective and a framework. IEEE/CAA J Autom Sin 4(3):389–395

Liao R (2017) Tencent discovers major loopholes in Google's AI platform TensorFlow. https://technode.com/2017/12/18/tencent-tensorflow/. Accessed April 2018

Licato J, Zhang Z (2017) Evaluating representational systems in artificial intelligence. Artif Intell Rev. https://doi.org/10.1007/s10462-017-9598-7

Liu MY, Breuel T, Kautz J (2017) Unsupervised image-to-image translation networks. https://arxiv.org/abs/1703.00848. Accessed April 2018

Mackintosh NJ (2011) IQ and human intelligence, 2nd edn. Oxford University Press, Oxford

Maurer M, Gerdes JC, Lenz B, Winner H (eds) (2015) Autonomous driving: technical, legal and social aspects. Springer, Berlin

Mcguire G, Tugemann B, Civario G (2014) There is no 16-clue sudoku: solving the sudoku minimum number of clues problem via hitting set enumeration. Exp Math 23(2):190–217

Merel J, Tassa Y, TB D, Srinivasan S, Lemmon J, Wang Z, Wayne G, Heess N (2017) Learning human behaviors from motion capture by adversarial imitation. https://arxiv.org/abs/1707.02201. Accessed April 2018

Minsky ML (ed) (1968) Semantic information processing. MIT Press, Cambridge

Moravčík M, Schmid M, Burch N, Lisý V, Morrill D, Bard N, Davis T, Waugh K, Johanson M, Bowling M (2017) DeepStack: expert-level artificial intelligence in heads-up no-limit poker. Science 356:508–513

Newell A, Simon HA (1976) Computer science as empirical inquiry: symbols and search. Commun ACM CACM Homepage 19(3):113–126

Ohlsson S, Sloan RH, Turán G, Urasky A (2017) Measuring an artificial intelligence system's performance on a verbal IQ test for young children. J Exp Theor Artif Intell 29(4):679–693

Raccoon L (1997) Fifty years of progress in software engineering. ACM SIGSOFT Softw Eng Notes 22(1):88–104

Redmon J, Farhadi A (2016) YOLO9000: better, faster, stronger. https://arxiv.org/abs/1612.08242. Accessed April 2018

Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. https://arxiv.org/abs/1506.02640. Accessed April 2018

Richter SR, Vineet V, Roth S, Koltun V (2016) Playing for data: ground truth from computer games. In: European conference on computer vision, pp 102–118

Rindermann H, Becker D, Coyle TR (2016) Survey of expert opinion on intelligence: causes of international differences in cognitive ability tests. Front Psychol. https://doi.org/10.3389/fpsyg.2016.00399

Ros G, Sellart L, Materzynska J, Vazquez D, Lopez AM (2016) The SYNTHIA dataset: a large collection of synthetic images for semantic segmentation of urban scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3234–3243

Russell S, Norvig P (2010) Artificial intelligence: a modern approach, 3rd edn. Pearson Education Limited, London

SAE J3016 (2016) Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems. SAE, Warrendale

Santana E, Hotz G (2016) Learning a driving simulator. https://arxiv.org/abs/1608.01230. Accessed April 2018

Schoenick C, Clark P, Tafjord O, Turney P, Etzioni O (2017) Moving beyond the Turing test with the Allen AI science challenge. Commun ACM 60(9):60–64

Silver D, Huang A, Maddison CJ, Guez A, Sifre L, van den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lillicrap T, Leach M, Kavukcuoglu K, Graepel T, Hassabis D (2016) Mastering the game of Go with deep neural networks and tree search. Nature 529(7587):484–489

Silver D, Hubert T, Schrittwieser J, Antonoglou I, Lai M, Guez A, Lanctot M, Sifre L, Kumaran D, Graepel T, Lillicrap T, Simonyan K, Hassabis D (2017a) Mastering Chess and Shogi by self-play with a general reinforcement learning algorithm. https://arxiv.org/abs/1712.01815. Accessed April 2018

Silver D, Schrittwieser J, Simonyan K, Antonoglou I, Huang A, Guez A, Hubert T, Baker L, Lai M, Bolton A, Chen Y, Lillicrap T, Hui F, Sifre L, van den Driessche G, Graepel T, Hassabis D (2017b) Mastering the game of Go without human knowledge. Nature 550:354–359

Srinivasan B, Parthasarathi R (2017) A survey of imperatives and action representation formalisms. Artif Intell Rev 48(2):263–297

Sternberg RJ (1985) Beyond IQ: a triarchic theory of human intelligence. Cambridge University Press, Cambridge

Sternberg RJ, Davidson JE (1983) Insight in the gifted. Educ Psychol 18(1):51–57

Thornton SM, Pan S, Erlien SM, Gerdes JC (2017) Incorporating ethical considerations into automated vehicle control. IEEE Trans Intell Transp Syst 18(6):1429–1439

Tong Y, Zhao L, Li L, Zhang Y (2015) Stochastic programming model for oversaturated intersection signal timing. Transp Res Part C 58:474–486

Turing AM (1950) Computing machinery and intelligence. Mind 59(236):433–460

Veeravasarapu VSR, Hota RN, Rothkopf C, Visvanathan R (2015) Simulations for validation of vision systems. Comput Sci. https://arxiv.org/abs/1512.01030

Vinge V (1993) The coming technological singularity: how to survive in the post-human era. In: Landis GA (ed) Vision-21: interdisciplinary science and engineering in the ear of cyberspace. NASA Publication, CP-10129, Washington, pp 11–22 von Ahn L, Blum M, Hopper NJ, Langford J (2003) CAPTCHA: using hard AI problems for security. In: Proceedings of international conference on the theory and applications of cryptographic techniques, pp 294–311

Wagner M, Koopman P (2015) A philosophy for developing trust in self-driving cars. In: Meyer G, Beiker S (eds) Road vehicle automation 2. Lecture notes in mobility. Springer, Cham. https://doi.org/10.1007/978-3-319-19078-5_14

Wang FY, Zhang JJ, Zheng X et al (2016) Where does AlphaGo go: from church-turing thesis to AlphaGo thesis and beyond. IEEE/CAA J Automatica Sin 3:113–120. https://doi.org/10.1109/JAS.2016.7471613

Wang L (2016) Directions 2017: BeiDou's road to global service. GPS World

Wang FY, Wang X, Li L, Li L (2016a) Steps toward parallel intelligence. IEEE/CAA J Autom Sin 3(4):345–348

Wang X, Zheng X, Zhang Q, Wang T, Shen D (2016b) Crowdsourcing in ITS: the state of the work and the networking. IEEE Trans Intell Transp Syst 17(6):1596–1605

Wang K, Gou C, Zheng N, Rehg JM, Wang FY (2017a) Parallel vision for perception and understanding of complex scenes: methods, framework, and perspectives. Artif Intell Rev 1:1–31

Wang X, Jiang R, Li L, Lin Y, Zheng X, Wang FY (2017b) Capturing car-following behaviors by deep learning. IEEE Trans Intell Transp Syst. http://ieeexplore.ieee.org/document/7970189/

Watzenig D, Horn M (2017a) Automated driving: safer and more efficient future driving. Springer, Cham

Watzenig D, Horn M (2017b) Automated driving: safer and more efficient future driving. Springer, Cham

You J. (2017) Deep learning based lane departure detection for automated vehicles. Bachelor Thesis, Tsinghua University

Zhao D, Huang X, Peng H, Lam H, Leblanc DJ (2017) Accelerated evaluation of automated vehicles in car-following maneuvers. IEEE Trans Intell Transp Syst. http://ieeexplore.ieee.org/document/7933977/

Zheng NN, Tang S, Cheng H, Li Q, Lai G, Wang FY (2004) Toward intelligent driver-assistance and safety warning systems. IEEE Intell Syst 19(2):8–11

Zheng NN, Liu ZY, Ren PJ, Ma YQ, Chen ST, Yu SY, Xue JR, Chen BD, Wang FY (2017) Hybrid-augmented intelligence: collaboration and cognition. Front Inf Technol Electron Eng 18(2):153–179