A Testbed for Evaluation of Architectures for Physical Agents

Dongkyu Choi, Michael Morgan, Chunki Park, and Pat Langley
Computational Learning Laboratory
Center for the Study of Language and Information
Stanford University, Stanford, California 94305
{dongkyuc, mmmorgan, chunki.park}@stanford.edu, langley@csli.stanford.edu

Abstract

Although cognitive architectures provide an excellent infrastructure for research stretching over various fields, their integration of multiple modules makes their evaluation difficult. Due to the lack of analytical criteria, the cost of demonstrations, and varying specifications among different architectures, developing evaluation methods is a challenge. In this paper, we describe a testbed for evaluation that revolves around an in-city driving environment. With its familiar but challenging missions cast in a rich setting, the testbed provides a uniform and competitive environment for evaluating cognitive architectures.

Introduction

Cognitive architectures combine ideas from psychology, computer science, and other fields to support research on general intelligence. They usually incorporate many different components that operate in combination to perform reasoning and problem solving, and learn new knowledge. For this reason, they provide a good infrastructure for research, but evaluation of these architectures is extremely difficult, partly due to the lack of analytical criteria and the cost of demonstrations. In addition, different architectures have different specifications that support distinct capabilities, further complicating systematic evaluation.

For these reasons, researchers in this field have mainly focused on empirical evaluation, measuring performance and learning across several different test domains. Although this provides a way to estimate the capabilities of individual architectures, each group of researchers uses its own favorite domains, often developed in-house, and it has been very difficult to compare the capabilities and performance of different architectures. There have been some efforts to resolve this issue, General Game Playing (Love et al., 2006) and RoboCup (Kitano et al., 1997) being excellent examples. However, the former encodes knowledge in strict logical terms, making its application to physical domains extremely hard. Also, the latter inherently focuses on low-level control, with participants engineering solutions to the particular game. These factors have kept them from being adopted widely by architectural researchers.

In this paper, we propose a testbed for empirical evaluation that provides a uniform framework to test agent architectures. The testbed also provides a competitive environment that supports side-by-side performance comparisons. Unlike General Game Playing, which has mostly been used for logical, deterministic, complete-information games, the testbed is designed specifically for architectures for physical agents. Also, thanks to the richness of driving in urban settings, the testbed lets agent developers focus on high-level strategies that are interesting, once the basic driving behaviors are programmed properly. In the following sections, we review and justify our motivations behind selecting the environment as our test domain, and we provide an overview of the testbed. Next, we explain various evaluation criteria available, after which we review related work and sketch our future plans.

Motivations for an In-City Driving Domain

Driving is a central activity in many people's lives. They commute to work, deliver packages, or shuttle people around, and engage in many other pursuits that involve complex, rich behavior. At the same time, there exist well-established rules and common sense for driving that constrain this behavior and make complex tasks tractable. For example, we drive on one side of the road, stop for pedestrians, and stay under the speed limit. Moreover, the domain is familiar to most people, making the tasks and the constraints on them comprehensible.

These reasons motivate our choice of in-city driving as a testbed. Driving involves many different goals, and drivers encounter a variety of situations. In particular, in-city driving tasks require tremendous amounts of attention, as compared to highway driving or even flying a small aircraft which involve interacting with many fewer objects. Therefore, we chose to focus on urban driving tasks as our research domain. Here, we have various objects, including pedestrians, traffic signals, and other vehicles moving in different directions, to name a few. A driving agent must attend to these objects while still maintaining safe and legal behavior. However, the domain does not require complicated manipulations, and necessary sensors are relatively simple.

This is why the in-city driving domain provides an interesting and rich environment to test different capabilities of agent architectures, while still being practical and feasible.

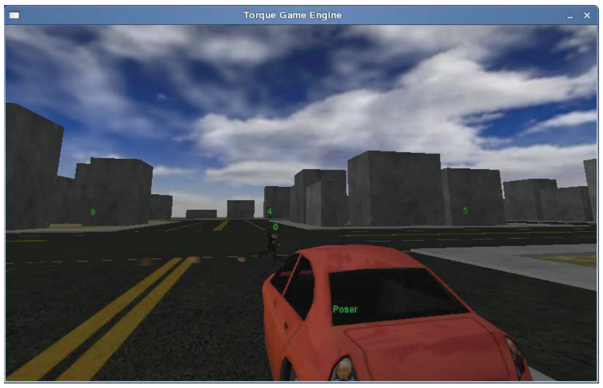

Figure 1: A screen shot of the in-city driving domain.

Table 1: An example of object information that an agent perceives in the environment.

```
((SELF ME SPEED 0.0 WHEELANGLE 0 THROTTLE 0
  LIMIT 25 BRAKE 0 SEGMENT S1734 HITS 0)
 (STREET FIRST) (STREET SECOND) (STREET B)
 (INTERSECTION S1812 STREET B CROSS FIRST
  DIST 5.73842)
 (SEGMENT S1734 STREET B DIST 88.986)
 (SEGMENT S1767 STREET FIRST DIST 19.7416)
 (BUILDING B1659 ADDRESS 8 STREET FIRST
  C1ANGLE 82.8411 C1DIST 33.6707
  C2ANGLE 58.9024 C2DIST 40.6854)
 (BUILDING B1682 ADDRESS 1 STREET B
  C1ANGLE 45.4169 C1DIST 35.197
  C2ANGLE 32.5563 C2DIST 49.2716)
 (LANE-LINE WHITE1 COLOR WHITE
  DIST 11.65 ANGLE -84.8146 SEGMENT S1767)
 (LANE-LINE YELLOW2 COLOR YELLOW
  DIST 5.814 ANGLE 4.83383 SEGMENT S1734)
 (PACKAGE P1 ADDR 5 STREET A DELIVERED N)
 (PACKAGE P2 ADDR 1 STREET B DELIVERED N))
```

Table 2: Sample actions that an agent can execute in the environment.

`(*cruise)`	:	`NO OP`
`(*gas %pedal)`	:	`push the gas pedal at the given percentage`
`(*brake %pedal)`	:	`push the brake pedal at the given percentage`
`(*straighten)`	:	`straighten the steering wheel`
`(*steer angle)`	:	`steer at the given angle`

Also, it is a very familiar environment to agent developers, as they encounter similar situations every day.

A Testbed for Evaluation of Architectures

Previously, we developed a two-dimensional version of an in-city driving domain (Choi et al., 2004) and used it as a testbed for our ICARUS agent architecture (Langley & Choi, 2006). Although the visualization was not up to the standards of modern video games, it provided a rich environment with various objects an agent can interact with, including streets with lane lines, sidewalks, buildings with addresses, traffic signals and drone cars. We gave the agent the task of driving a car in the city, pursuing goals like package delivery.

Although the domain provided an excellent testbed for our initial research purposes, we soon discovered its shortcomings, especially in its description of vehicle dynamics and the visualization of the environment. In this context, we started to design and implement a new in-city driving environment.

Implementation

To build this new testbed, shown in Figure 1, we used the *Torque Game Engine* (TGE), a three-dimensional game engine developed by GarageGames. It provides convenient and well-built templates for three-dimensional games, as well as robust networking capabilities, a C++-like scripting language, a terrain generating tool, and other powerful tools for game developers. Also, the game engine lets us develop new games in cross-platform computing environments including Linux, Mac OS X, and Windows.

The game engine provides a ready-made physics module for vehicles that not only has realistic default parameters but also allows user-defined settings. We used *TorqueScript*, a scripting language the game engine provides, for the interface between the new domain and our Lisp-based architecture, as well as several customized mission maps. The language supports a sufficient complement of functions, including mathematical, standard input/output, basic object manipulation, and other helper functions. Scripts written in this language do not need to be compiled with the main game engine, providing flexibility for many different applications.

Since our simulation is based on TGE, architectures can directly connect to the testbed in C++ language. However, the testbed also provides an interface for Lisp-based architectures. Through the interface, agent architectures can receive information on perceived objects in the environment and execute actions in the world. Table 1 shows examples of perceptions an agent can obtain from the world, and Table 2 gives some sample actions available in the environment.

Tasks Supported

Like people, an agent in the domain can work on many different tasks. We can have it simply drive around without hitting pedestrians as they cross streets, or it can pursue additional tasks, including:

- delivering packages to multiple addresses,
- picking up passengers and dropping them off at various destinations,
- chasing another car and issuing tickets like police,
- driving an ambulance with patients to hospitals, or
- joy-riding in a sports car as fast as possible.

Each of these tasks specifies a particular goal for the agent; they also involve different parameters for the mass, height, engine torque, and braking capability of the vehicle an agent will drive. The testbed provides an interface that supports easy changes of goal and parameters. The tasks are intended to test capabilities of agents ranging from sensory-motor control to high-level decision making and planning.

Evaluation Criteria

One should consider various dependent measures at different levels when evaluating integrated systems (Langley & Messina, 2004). The testbed provides basic and combined measures suitable for driving situations, and could allow comparisons with other architectures as well as baseline systems of the user's choice.

Whenever an agent drives a vehicle in the simulated city, the testbed stores key variables like the location, velocity, and heading at predefined intervals. In addition, it records the number of undesirable behaviors, such as:

- hitting pedestrians,
- collisions with other vehicles and buildings,
- driving faster than the posted speed limit,
- driving on the wrong side of road, and
- violations of traffic signals.

During post-processing, the basic driving behavior can be evaluated using these traces, providing measures of the agent's driving quality. The testbed supports one or more mission-specific evaluation criteria, such as:

- delivery time and the number of delivery errors,
- the time between pick up and drop off, along with the number of errors,
- the time needed to overtake other vehicles and the number of tickets issued,
- the time between pick up and drop off of patients at the emergency room, and
- average speed and the number of tickets received.

These criteria are designed to measure the agent's success during the missions in both continuous and discrete scales, providing additional evaluation measures for architectures.

Support for Agent Capabilities

Our urban driving testbed also supports a variety of functional capabilities that are relevant to research on cognitive architectures, but that are seldom studied for lack of an appropriate environment. These include:

- pursuing multiple goals, such as reaching a destination rapidly, driving safely, and obeying traffic laws;
- managing limited resources, ranging from physical quantities like gasoline to the agent's perceptual attention;
- encoding, using, and learning spatial knowledge in terms of places and routes that connect them;
- storing and retrieving episodic memories about events the agent has encountered and activities it has pursued;
- representing and exhibiting emotions like frustration and anger, along with the physical responses they elicit.

These capacities in turn suggest additional ways to evaluate agent architectures that move beyond the simple metrics outlined above, but we will reserve discussion of them for another paper.

Related Research

There have been some efforts to provide testbeds for empirical evaluation of architectures. The World Modelers Project (Langley et al., 1981), one of the first such attempts, provided a simulated physical environment that aimed to support research on integrated intelligent systems at a time when robotic platforms were still unreliable.

More recently, RoboCup (Kitano et al., 1997) has provided a popular testbed for artificial intelligence and robotics research, and it has promoted research and competitions for a decade. However, as the competition has grown larger, the recent focus has been more on winning techniques specific to the game, rather than on the fundamental research issues that were originally pursued.

TacAir-Soar (Jones et al., 1999) showed extensive capabilities of a cognitive architecture, and confirmed the possibility of using physical simulations to evaluate such systems. But the testbed was designed for military training exercises and has not become widely used in the AI community.

General Game Playing (Love et al., 2006) is a more recent testbed that has hosted competitions since 2005. With its logical description language, the testbed provides complete specifications of games to players, letting them focus on decision-making processes. It supports a broad class of games, but it is inappropriate for complex physical settings that involve continuous space, many interacting entities, and partially observable environments.

Directions for Future Work

Our work is still in its early stages, and we plan to improve the in-city driving domain and develop it into a uniform testbed that developers can use to evaluate their cognitive architectures in a competitive environment. We intend to implement many additional features, the most important being support for multiple agents. The current testbed supports only single agent missions, which restricts the scope of evaluation. With multiple agents simultaneously running in the world, possibly controlled by different architectures, we can introduce more competitive missions. Racing to flags, playing hide and seek with other agents, and chasing each other are some examples.

The great advantage of in-city driving is that it supports many alternative tasks in a single physical environment. This should let researchers evaluate the generality of their architectures with relatively little effort. However, to support such evaluations, we must specify more formally the set of driving tasks outlined earlier and the metrics associated with them. This will be another priority for our future work.

Conclusions

Cognitive architectures provide an important path toward developing intelligent systems, but their integrated nature make their evaluation impractical with analytical methods. In response, we have proposed an in-city driving testbed that supports empirical evaluation of such architectures.

This domain is a familiar but challenging environment for physical agents. Although it is rich with many objects and many possible tasks, it is reasonably constrained, with a small set of manipulators and relatively simple sensors. In-city driving supports a variety of achievement and maintenance goals that interact to produce complex behavior. With multiple missions and corresponding evaluation criteria, it provides a promising testbed for empirical evaluation of architectures. We hope to package the environment and make it available online in the near future.

Acknowledgements

This material is based on research sponsored by DARPA under agreement numbers HR0011-04-1-0008 and FA8750-05-2-0283. The U. S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Discussions with Negin Nejati and Tolga Konik contributed to the ideas we have presented in this paper. Authors would also like to thank GarageGames for allowing the free use of its products, including Torque Game Engine 1.4.2, BTCP Car Pack, and Urban Pack 1.0.0 for our development.

References

Choi, D., Kaufman, M., Langley, P., Nejati, N., & Shapiro, D. (2004). An architecture for persistent reactive behavior. *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi Agent Systems* (pp. 988–995). New York: ACM Press.

Jones, R. M., Laird, J. E., Nielsen, P. E., Coulter, K. J., Kenny, P., & Koss, F. (1999). Automated intelligent pilots for combat flight simulation. *AI Magazine, 20*, 27–41.

Kitano, H. Kuniyoshi, Y. Noda, I. Asada, M. Matsubara, H. & Osawa, E. (1997). Robocup: A challenge problem for AI. *AI Magazine*, *18*, 73–85.

Langley, P., Nicholas, D., Klahr, D., & Hood, G. (1981). A simulated world for modeling learning and development. *Proceedings of the Third Conference of the Cognitive Science Society* (pp. 274–276). Berkeley, CA.

Langley, P., & Messina, E. (2004). Experimental studies of integrated cognitive systems. *Proceedings of the Performance Metrics for Intelligent Systems Workshop*. Gaithersburg, MD.

Langley, P., & Choi, D. (2006). A unified cognitive architecture for physical agents. *Proceedings of the Twenty-First National Conference on Artificial Intelligence*. Boston: AAAI Press.

Love, N. C., Hinrichs, T. L., & Genesereth, M. R. (2006). *General game playing: Game description language specification* (Technical Report LG-2006-01). Logic group, Department of Computer Science, Stanford University, Stanford, CA.