

Towards a layered view of control

Pravin Varaiya
Department of Electrical Engineering and Computer Science
University of California, Berkeley, CA 94720
varaiya@eecs.berkeley.edu

September 21, 1997

Abstract

I want to provide three perspectives on hierarchical architectures. The first focuses on how the space of automated highway system designs was structured. The second describes two hierarchically-layered designs. The third suggests a theoretical framework prompted by these two stories. I will end with a brief video that describes elements of one AHS design.

AHS design

In 1994, the U.S. DoT awarded a 7-year, \$200 M contract to develop an automated highway system (AHS) prototype. The contract was prompted by the realization that the nation's highway system had reached a performance plateau, and computer, communication and control technologies could be used to design an automated highway system with a quantum improvement in performance. The contract was awarded to a nine-member partnership called the National Automated Highway Systems Consortium or NAHSC.

In addition to the nine core members, the NAHSC comprises more than 120 associate participants representing nine stakeholder communities. The associate members ensure that the Consortium's designs reflect their interests. The associate members also help to promote awareness of automated highways to a wider public. An important event in that effort was a 4-day public demonstration of AHS technologies on interstate I-15 in San Diego in August 1997.

The first challenge facing the NAHSC team was to structure the "space of AHS designs" in a way that highlighted significant design alternatives. Six design attributes were selected.

1. Distribution of intelligence

Autonomous – all information within each vehicle

Cooperative – vehicles communicate info to each other

Infrastructure supported – hwy collects info and communicates to vehicles

Infrastructure managed – hwy also commands aggregate traffic

Infrastructure controlled – hwy commands individual vehicle

2. Separation policy

- Free agent – each vehicle maintains safe distance from neighbors
- Platoon – close spacing within platoon, safe distance between platoons
- Slot – each vehicle keeps within hwy-assigned time-slot on road

3. Mixed traffic

- Dedicated lanes with continuous physical barriers
- Dedicated lanes with gaps in physical barriers
- Dedicated lanes with virtual barriers (striped)
- Full mixing of automated and non-automated vehicles

4. Mixed vehicle classes

5. Entry/exit design

- Dedicated ramps to dedicated lanes
- Transition lanes next to dedicated lanes
- Today's entry and exit designs

6. Obstacle handling

- Manual sensing and avoidance
- Automatic sensing and manual avoidance
- Automatic sensing and automatic avoidance

A reflection on these attributes shows a preoccupation with:

- Control architecture—distribution of information and control authority among distributed agents
- Operating rules—how to coordinate actions of agents by physical restrictions or protocols
- Exception handling—how to deal with “faults”

Hierarchically-layered control

The use of layered or hierarchical frameworks for the design and implementation of algorithms for comprehending and controlling systems is quite common in practice although there is not much theory that explores the practice.

I discuss two examples. The first example is a video-based traffic surveillance system. The second example is a system designed to control the movement of vehicles on an automated highway. In section 3 we then attempt to formalize the properties of multi-layered frameworks. This formulation is still very tentative.

The first system called Roadwatch automatically produces descriptions of traffic along a highway. The system has been prototyped and tested [1, 2].

Roadwatch is organized in four layers.

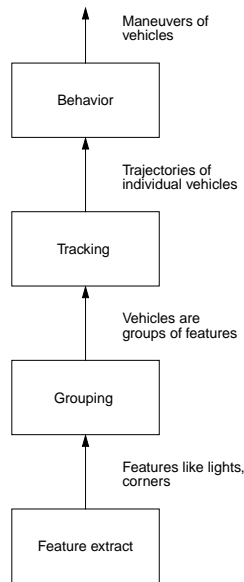


Figure 1: Roadwatch has four layers beginning with feature extraction and ending with classification of vehicle maneuvers

1. Layer 0. Low level signal processing produce a digitized image of the highway traffic scene at 15 frames per second.
2. Layer 1. Extract characteristic features semantically associated with ‘real’ vehicle corners, license plates, tail lights. Interpretation determines whether feature is real or ‘noise’.
3. Layer 2. Group features into vehicles. Rigid body dynamics implies features of same vehicle will not move relative to each other through successive frames and will be within a certain maximum distance from each other, whereas features belonging to different vehicles may exhibit relative motion.
4. Layer 3. Group vehicles into trajectories. Vehicles cannot accelerate or decelerate too rapidly.
5. Layer 4. Infer individual behavior from collective motion. If a particular vehicle is stopped (its trajectory shows zero speed), but if adjacent vehicles are moving, traffic flow theory allows one to infer that the vehicle is stopped because it is ‘disabled’

Thus underlying the four-layer algorithmic decomposition of Figure 1 is a hierarchy of theories: each framework has its own entities, operations that can be performed on those entities, and means of ‘embedding’ or ‘abstracting’ entities at one layer into entities at the higher layer. Conversely, an entity at a higher layer can be ‘realized’ (usually in a non-unique manner) by entities at a lower layer.

The control system proposed in [3] is organized in four layers.

1. Layer 0. ‘Open loop’ system of vehicle dynamics and sensors.

2. Layer 1. Suite of vehicle feedback laws that determine throttle, braking and steering actuator commands as a function of sensor readings. Each law is designed to carry out a certain maneuver selected at Layer 2.
3. Layer 2. Coordination layer. vehicle's controller communicates with its peers in neighboring vehicles to determine which maneuver (such as lane change, exit, entry) to execute so as to fulfill its goal (reaching its exit) while accommodating the requests from the roadside controller at Layer 3.
4. Layer 3. Link layer. Advises vehicles within its link (a distance of, say, 1 km) about speed and roadway conditions.

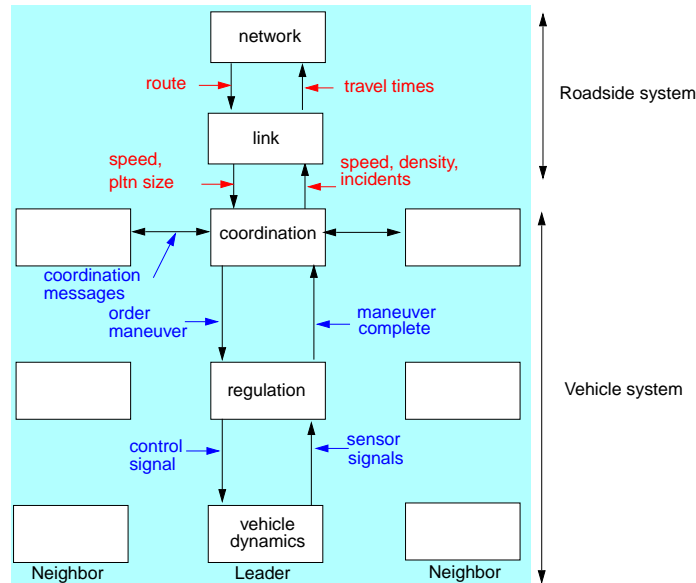


Figure 2: The four-layer control architecture starts with the open loop system and ends with traffic flow along a link.

As in the Roadwatch example we see that each layer of the control hierarchy involves different entities and algorithms that relate entities at one layer to entities at the adjacent layer. Moreover, there is a theoretical framework at each layer which provides a meaning to its entities, and relations between those frameworks that are operationalized in terms of those algorithms.

Figure 3 is a formalization of layered organization of control. Each layer incorporates the functions of ‘abstraction’, ‘reconstruction’, ‘intervention’ and ‘alarm’. Abstractions permit layer i entities to be represented (abstracted) into layer $i + 1$. Reconstructions goes in the opposite direction: a construct at layer $i + 1$ is given a more concrete representation at layer i . Intervention and alarm correspond to commands and abnormal events.

Each layer i can have several abstractions indexed k , located at layer $i + 1$. Each abstraction provides a framework or theory. The theory establishes rules for ‘grouping’ entities at layer i . The rules are determined by the theory. The theory provides the ‘syntactic’ rules for the

Figure 3: Each layer admits several frameworks or theories. Expressions of scenes at layer $i - 1$ are abstracted into expressions at layer i ; and an expression at layer i can reconstruct a scene at layer i .

relationship between entities at layer i . The theory is also equipped with ‘semantic’ rules that relate the ‘abstract’ entities of a theory to ‘concrete’ entities in the ‘real’ world.

A successful layered organization seems to conform to the principles of ‘self-containment’, ‘coherence’, and, perhaps ‘emergent behavior’. Pragmatic considerations suggests additional principles of ‘robustness’, ‘learning or adaption’, ‘scalability’, ‘technical change or heterogeneity’.

Principle of self-containment Each layer embodies theories that make it possible to discuss the design and implementation of that layer in isolation. Adjacent layers are encapsulated in well-defined interfaces. Each layer has a well-defined ‘language’ in which its theory or theories are expressed. The theory specifies how that layer behaves and how the adjacent layers provide ‘inputs’ and how they absorb ‘outputs.’ It is then possible to analyze the layer in isolation, to implement it, and to estimate the performance of that layer.

Principle of coherence Two aspects. First, the theories at the different levels must be consistent as much as possible. Second, the control system design that is based on these theories must lead to overall behavior that conforms to the purposes for which the system was constructed.

Principle of scalability The architecture must accommodate increasing the size of the distributed system. This means chiefly that the information channels and processing requirements that support the control actions are not overstressed with scale. That is to say the control must be distributed, while the burden of coordination remains bounded despite growing size.

Principle of robustness With growth in size, failures are inevitable. The architecture must tolerate failures and recover from them, localizing the impact of failures if possible.

Principle of emergent behavior With growth in size, unforeseen behaviors may emerge. These may be felicitous or harmful. The architecture must provide for taking advantage of the former and mitigating against the latter.

Principle of technological progress or heterogeneity With time, the performance of system elements that embody certain functions will improve. The architecture should accommodate such changes ‘locally’, that is it should not require that all elements be changed. In other words, the architecture should permit heterogeneous realizations of its functions.

Principle of learning or adaptation With changes in the environment or with better understanding of how the system behaves, there will be better theories at the different layers will emerge. The architecture must permit changes in the theory to have localized impact.

Meta principle of localization and incrementalism Deployment of a distributed system is a macro-change. However, changes in the distributed system should be capable of being incrementally introduced and its impact localized.

References

- [1] J. Malik and S. Russell, “Traffic surveillance and detection technology development: New traffic sensor technology final report,” tech. rep., UCB-ITS-PRR-97-6, Institute of Transportation Studies, University of California, Berkeley, CA 94720, January 1997.
- [2] T. Huang and S. Russell, “Object identification in a Bayesian context,” tech. rep., 1997. IJCAI-97, 1997.
- [3] P. Varaiya, “Smart cars on smart roads: Problems of control,” *IEEE Transactions on Automatic Control*, vol. 38, pp. 195–207, February 1993.