Developing World Model Data Specifications as Metrics for Sensory Processing for On-Road Driving Tasks

Anthony Barbera[1], John Horst[1], Craig Schlenoff[1], Evan Wallace[1], and David W. Aha[2]

[1]Intelligent Systems Division
The National Institute of Standards and Technology
Gaithersburg, MD 20899
barbera@nist.gov, horst@nist.gov, schlenof@nist.gov, ewallace@nist.gov

[2]Intelligent Decision Aids Group
Naval Research Laboratory (Code 5515)
Washington, DC 20375
aha@aic.nrl.navy.mil

ABSTRACT

Building knowledge-intensive real-time intelligent control systems is one of the most difficult tasks that humans attempt. It is motivated by the desire to create an artificial reasoning system that displays intelligent behavior (i.e. that can act on the world and successfully accomplish activities that are only possible with the levels of knowledge processing exhibited by human beings). Measuring and evaluating the success of such systems is difficult - a system's observable behavior is not always indicative of its correctness or quality. This is especially true in complex real-time control systems such as autonomous on-road driving, which is the focus of our Defense Advanced Research Project Agency (**DARPA**) Mobile Autonomous Robot Software (**MARS**) On-Road Driving Project. We are performing a task analysis and developing performance metrics for autonomous on-road driving, and using the NIST Real-time Control System (RCS, now referred to as 4D/RCS) [1] design methodology and reference architecture to develop a task decomposition representation format for on-road driving task knowledge. This representation is used as the framework to further specify the world model entities, attributes, features, and events required for proper reasoning about each of the subtask activities. These world model specifications, in turn, are used as the requirements for the sensory processing system; they identify objects that have to be measured in the environment, including their resolutions, accuracy tolerances, detection timing, and detection distances for each subtask activity. We describe our project's task and world modeling knowledge, exemplify their application, and describe a set of performance metrics for validating sensory processing activities by evaluating the world model representations the system produces for each individual component subtask activity. In this way, taxonomies of autonomous capabilities can be developed and tested against these sensory processing and world model building performance metrics.

Keywords: On-road driving, performance metrics, sensory processing, task decomposition, finite state machines

1. Introduction

NIST has used the 4D/RCS [1] approach (Figure 1) for developing autonomous intelligent vehicle control systems for a number of years. These efforts support both off-road and on-road driving tasks. This work, whose goal is to approach human levels of performance in autonomous driving, has identified significant research and development areas. There is the behavior generation component that involves reasoning from real-time world model representations to conduct strategic and tactical behaviors for both off-road military missions and on-road civilian driving tasks. This includes planning alternate courses of action and alternate paths, evaluation of these plans, and selection of the most appropriate action through some type of value judgment. We are currently focusing our work in this area of reasoning, planning, and decision-making. However, a particularly challenging impediment that requires significant attention is the area of sensors and sensory processing algorithms to generate accurate, registered world maps and the recognition and classification of entities at sufficient resolution to populate a world model representation for the behavior generation component.

Complex real-time control systems are characterized by their components for (1) *sensory processing* for measuring entities and events of interest in the environment, (2) *internal world model processing* that derives world representations from sensory processing and task context internal states, and (3) *behavior generation processing* that reasons from this world model, develops alternate plans, and makes value judgments to select and execute the next appropriate output plan for accomplishing the goal tasks. Performance metrics are needed at the level of these three internal processing components that can be used to judge their quality and correctness.

The sensory processing components of the world model perform sensor fusion, feature and attribute detection, object classification, map building, etc. – all in the context of the present task activities. Our current work addresses how to develop these sensory processing components given the world model data specification.

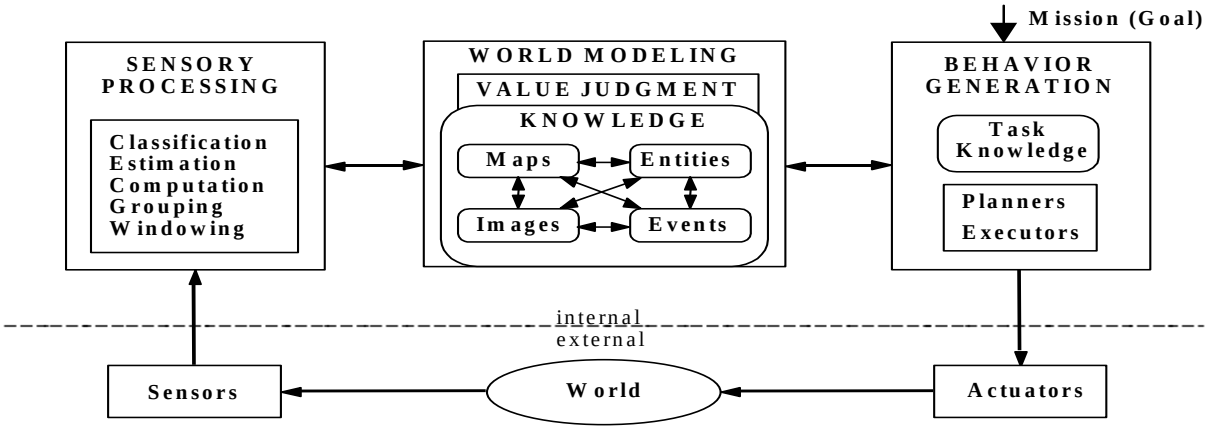


Figure 1. The basic internal structure of a 4D/RCS control loop. Sensory processing performs the functions of windowing, grouping, computation, estimation, and classification on input from sensors. World modeling maintains knowledge in the form of images, maps, entities, and events with states, attributes, and values. Relationships (e.g., class membership, inheritance, pointers to situations) between images, maps, entities, and events are defined by ontologies. Value judgment provides criteria for decision making. Behavior generation is responsible for planning and execution of behaviors.

The sensory processing requirements of different driving tasks have significantly different resolutions, identification, and classification requirements that suggests that performance metrics should be defined on a task-by-task basis. For example, the task of driving the vehicle along a highway requires the sensor system to identify large objects moving nearby, their direction, speed, acceleration, positions in the lanes (which means the sensory processing system must identify road lanes), and state of the brake and turn signal indicator lights on these objects. There is little requirement for detailed recognition of object types or the need to see them at a distance or to read signs alongside or overhead of the road.

However, if our autonomous vehicle decides to pass a vehicle on an undivided two lane road, then an extraordinarily detailed world representation must be sensed that identifies additional entities (e.g., upcoming intersections, rail road crossings, vehicles in the oncoming lane out to very large distances, lane marking types, and roadside signs). This level of sensor capability and sensor data for world model processing probably does not exist today.

Thus, because sensor requirements and sensory-world model processing performance metrics are highly dependent on the particular driving task that the system is trying to accomplish, they should be designed to be task specific. Our goal is to first develop a list of required driving tasks, and then to identify the detailed world model entities, features, attributes, resolutions, recognition distances, minimum data update times, and timing for task stability for each of these decomposed subtask activities. This will provide the set of specifications that allow us to determine if particular sensor systems and sensor processing algorithms are sufficient to support particular driving tasks. Conversely, if the goal is to accomplish a particular set of driving tasks, this specification can be used to select the appropriate sensors and specify the required sensor processing requirements.

By defining in great detail the features, attributes, and classifications of entities required in the world to reason about and generate specific driving tasks, we will have a specification that can be used to (1) identify the requirements to the sensory processing researchers and (2) be used as testing performance metrics to evaluate the capabilities of various sensors and sensory processing algorithms.

2. Task Decomposition Knowledge

One of NIST's efforts in its DARPA MARS project is to provide a task analysis for autonomous on-road driving that can, among other things, serve as the basis for developing a number of performance metrics. This task analysis is based on earlier work performed by the Department of Transportation [3], although our context required many modifications, additions, and, in general, a focus on tasks relevant to autonomous driving.

Work at NIST over the past 25 years has led to the development of the Real-time Control System (RCS) [1], now referred to as the 4D/RCS methodology and reference architecture. It provides a formal approach for designing and implementing complex, intelligent, real-time control systems.

The 4D/RCS methodology uses a hierarchical task decomposition format [2] for representing domain knowledge. Hierarchies are the architectural mechanisms

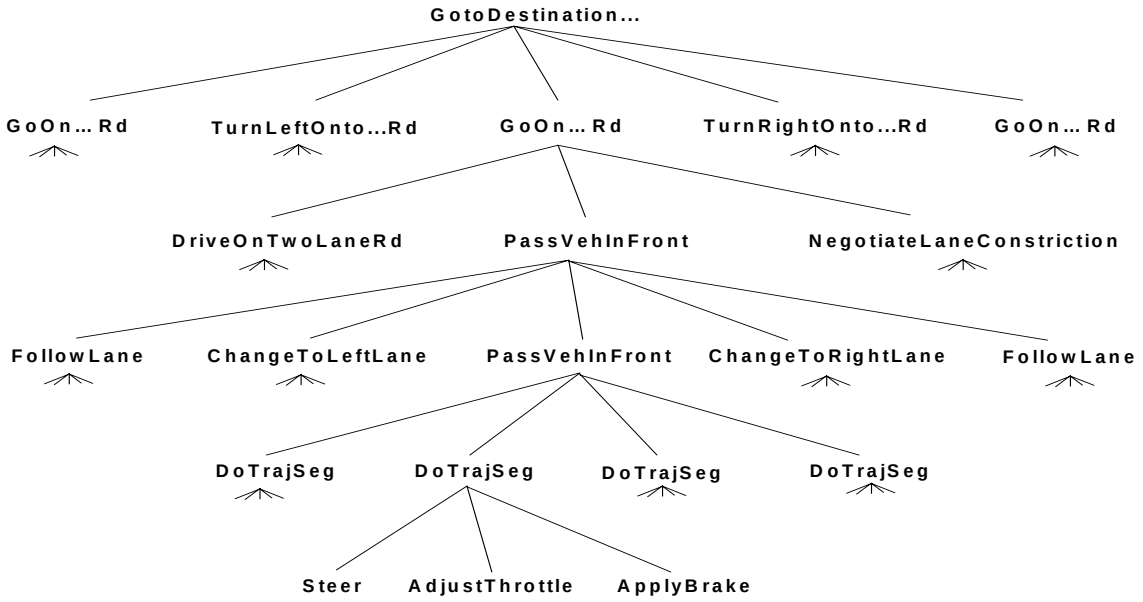


Figure 2. Example representation of a hierarchical task decomposition for the on-road driving task.

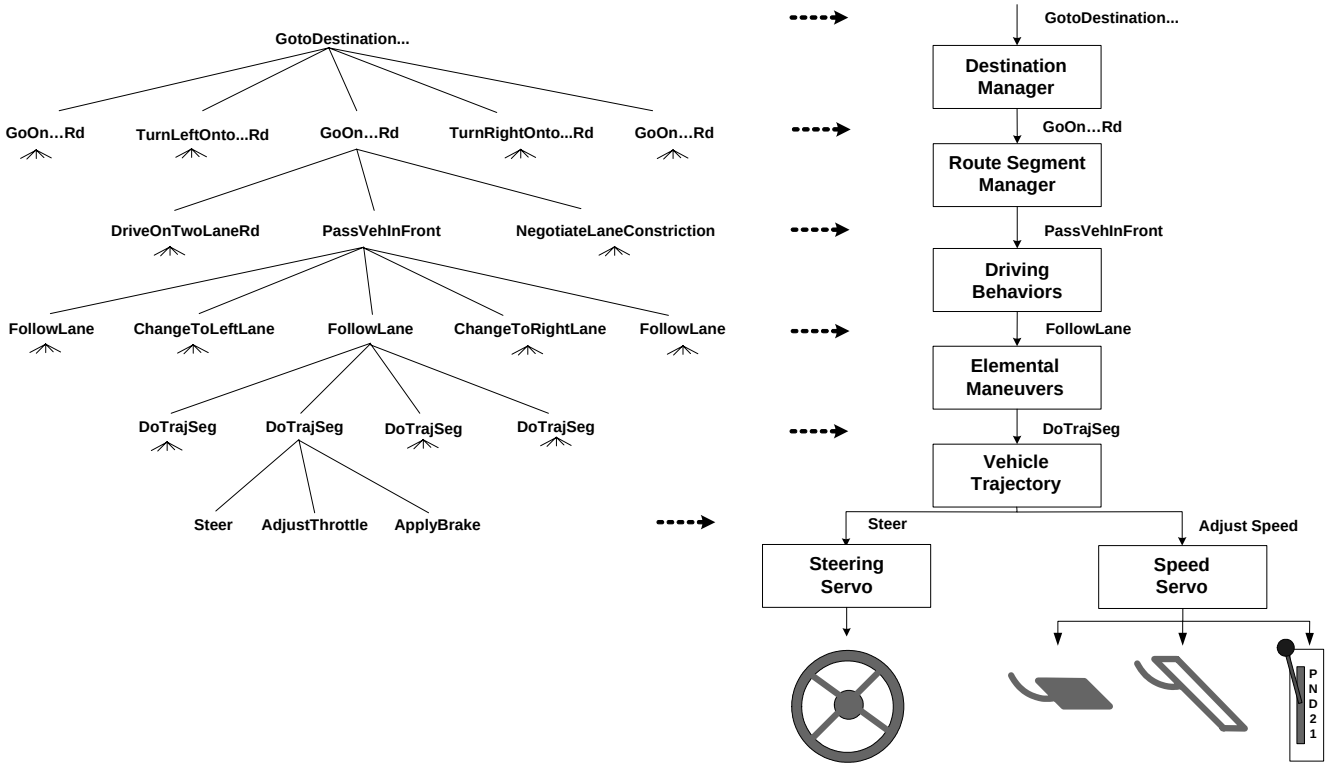


Figure 3. The RCS implementation creates a hierarchical organization of agent control modules (right side of figure) that will be the execution engine for the task decomposition (left side of figure). An agent control module is assigned to each actuator system to be controlled and an organizational structure is built up that mimics the same number of layers in the task decomposition representation. Each corresponding agent control module will accept the appropriate subtask command at the equivalent level in the task hierarchy and will determine which subgoal command will next be given to its subordinate, based on the rules encoded in the corresponding state table. For example, the subgoal command *PassVehInFront* to the Driving Behaviors agent control module will select the state table that contains all of the rules necessary to evaluate the present world state at this level of abstraction and in the context of passing the vehicle in front. It will send the appropriate subgoal command (i.e., *FollowLane*, *ChangeToRightLane*, or *ChangeToLeftLane*) for this present state to the Elemental Maneuvers agent control module.

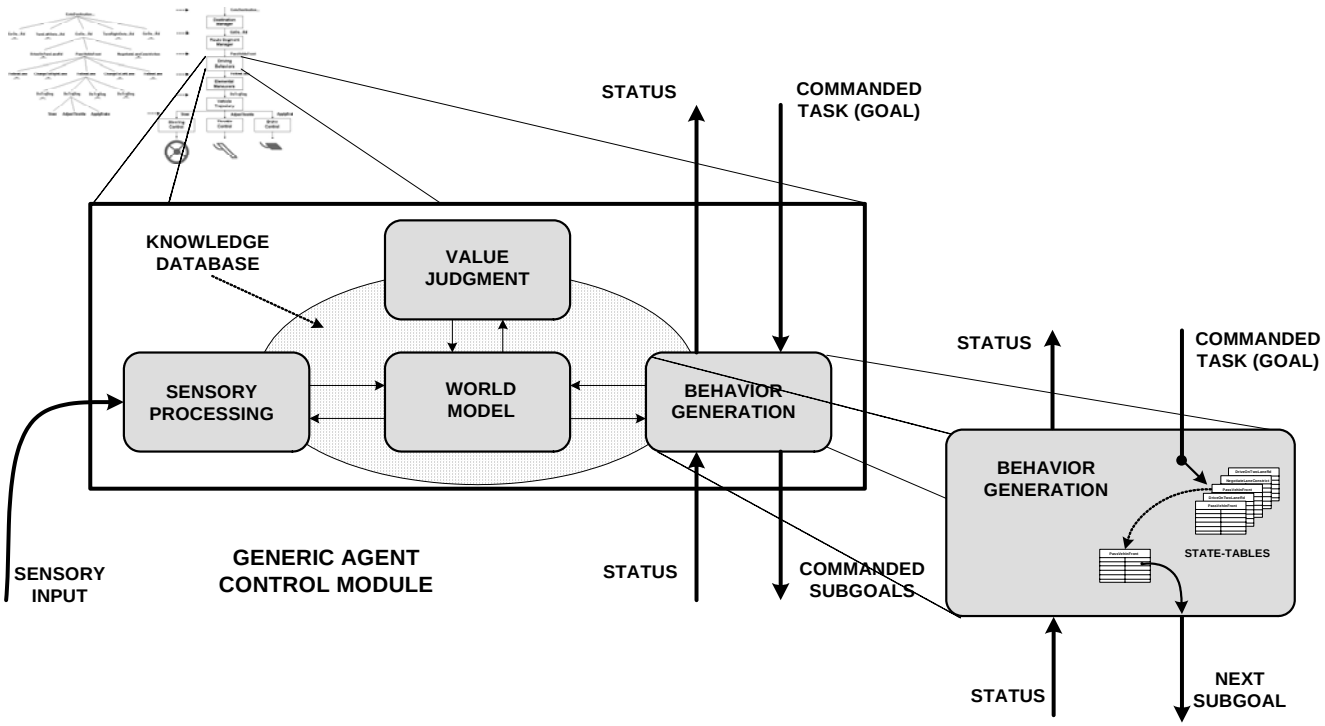
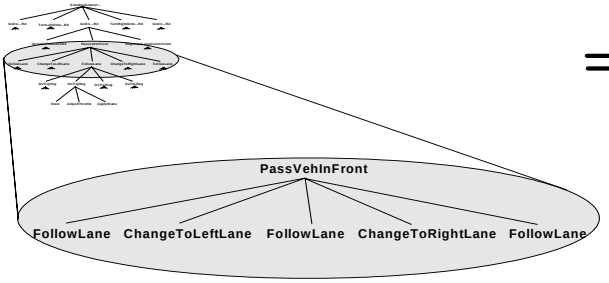


Figure 4. Every agent control module in the RCS hierarchy has the same processing structure of the generic agent control module. A module receives a commanded task (goal) that represents the present activity to be done at this level in the hierarchy at this instant. The Behavior Generation (BG) function uses this commanded task to look up and retrieve the state-table that contains the rules relevant to this activity. This sets the context for all of the processing at this module. Sensory Processing (SP) fills in world model data from the environment that is important to this particular task. If the situation requires planning activity, then the Value Judgment (VJ) function projects possible courses of action and performs some cost based analysis to determine a plan. As the situation creates matches to the rules in the BG's state table, the corresponding action part of the rule generates the next subgoal command to the subordinate agent control module.

used to "chunk" and abstract systems into manageable layers of complexity. The scenario descriptions of intelligent control system activities naturally evolve into a task decomposition representation because the scenarios are task sequences and can easily be discussed at many levels of abstraction, leading to well-defined levels within the task hierarchy. This provides a convenient framework for system designers and knowledge engineers to organize the information from the expert within an architecture that preserves the narrative character of the expert's scenarios, thus allowing the expert to easily review this representational format. Thus, a hierarchical task decomposition representational format is clearly well suited for this. Figure 2 shows an example of how a task decomposition hierarchy can be used to represent expert knowledge for the on-road driving task.

This task decomposition hierarchy also acts as a convenient structure in which to encode *semantic* knowledge from the expert. In the on-road driving task, semantic knowledge includes such knowledge items as the rules of the road, the rules that require the vehicle to drive more slowly on wet or icy roads or to allow larger following distances on wet roads, etc. Because each layer in the task decomposition represents a different abstraction level of the tasks, each layer also delineates levels of detailed task context for incorporating semantic knowledge relevant to that level of detail within a particular task's activities. We will exploit this very organized layering of the task knowledge into different levels of abstraction and task responsibility to aid us in performing a detailed analysis of the knowledge associated with finely partitioned task activities for the on-road driving activities.

Given that the 4D/RCS methodology uses a task decomposition decision hierarchy to capture knowledge from the expert's narratives, it is straightforward to instantiate this into an implementation of a hierarchical architecture for agent control modules executing this task decomposition in a one-to-one fashion (Figure 3). This 4D/RCS implementation technique represents the knowledge in the implemented system in a manner that continues to be easily recognized by the domain expert. It maintains the layered partitioning of the task to create levels of abstraction, task responsibility, execution authority, and knowledge representation in a manner so as to greatly enhance the designer's ability to think about each of these layers separately. Each layer totally encapsulates the problem domain at one level of abstraction so all aspects of the task at this one layer can be analyzed without



PassVehInFront	
NewPlan	S1 FollowLane
S1 ConditionsGoodToPass	S2 ChangeToLeftLane
S2 ConditionsGoodToPass InPassingLane	S3 FollowLane
S3 ClearOfPassedVehicle SufficientReturnSpace	S4 ChangeToRightLane
S4 ReturnedToLane	S0 FollowLane Done
Input Situations	Output Actions

Figure 5. The task to "Pass a vehicle in front" is shown in both the task tree representation and the state table representation. The task knowledge for this particular on-road driving task is the set of subgoals, their sequence, and the conditions (i.e., current world situations) that cause each of these subgoals to be commanded. Here the sequence of subgoals is to "FollowLane", "ChangeToLeftLane", "FollowLane", "ChangeToRightLane", and "FollowLane". These are listed in the output action side (right side) of the state table. The conditions that trigger these output actions are current world situations such as "ConditionsGoodToPass", "InPassingLane, and "ClearOfPassedVehicle", etc.; these are listed in the input condition side (left side) of the state table.

overwhelming the designer. All of the system's interactions and co-ordinations within the context of this abstraction layer are contained here so that modifications and enhancements to it can be evaluated with respect to their completeness and potential interaction with other task activities at that same abstraction level. At each layer, all of the relevant sensory processing, world modeling, and behavior generation processing for that level of responsibility and authority is encapsulated. As such, the 4D/RCS approach provides a very well ordered representation of the tasks at various levels of finer and finer detail, clustered at each level in a task sensitive context. This is ideal for the manner in which we want to identify the performance metrics.

A generic agent control module (Figure 4) is used as the unit building block in our hierarchical implementation system. Finite State Machines (FSMs) cluster and order the task decomposition knowledge rules specific to a particular task goal for an agent control module. Part of the implementation procedure is to determine which rules apply to each particular subtask activity at each level in the hierarchy. This is a natural outcome of the task decomposition process.

A task is decomposed at one level into a sequence of simpler subtask *actions*, which may perform some processing at that level and/or send a command to the next subordinate level. The representation of this sequence can be in the form of a FSM that can be implemented as a state table, which is the ordered representation of the rules used to encode those sets of conditions that will yield the correct sequencing by executing the appropriate actions to accomplish that particular task.

State tables are also an extremely convenient representational format for the developer. They capture the relevant task sequencing and state knowledge at each control module for every task activity. As the need arises to evolve the system, the state table that contains the knowledge rule set that concerns the activity to be modified can be easily identified and retrieved. Potential conflicts that might arise in the execution are easily detected through inspection (because this is such a small set of rules) and avoided by ordering the rules using additional state variables. In this manner, the expert can provide additional task knowledge to resolve potential conflicts in specific task activities rather than require the system designer to devise some arbitrary and general conflict resolution mechanism. Figure 5 displays an example mapping of task decomposition knowledge into a state table.

3. World Model Knowledge

The FSMs described above are used to encode task decomposition knowledge. Each line of each state table uses some symbolic value to describe the present situation that must be matched to execute the corresponding output action of that rule. The processing required to determine whether a given situation is true can be thought of as a knowledge tree lying on its side, funneling left to right, from the detailed sensory processing branches until all of the values have been reduced to an appropriate situation identification encoded in a symbolic value such as "ConditionsAreGoodToPass" (see Figure 6). This lateral tree represents the layers of refinement processing made on the present set of world model data to conclude that a particular situation now exists (e.g., "ConditionsAreGoodToPass").

The identification of these layers of knowledge processing to evaluate to the situation value is done in reverse. We know that we cannot change into the oncoming

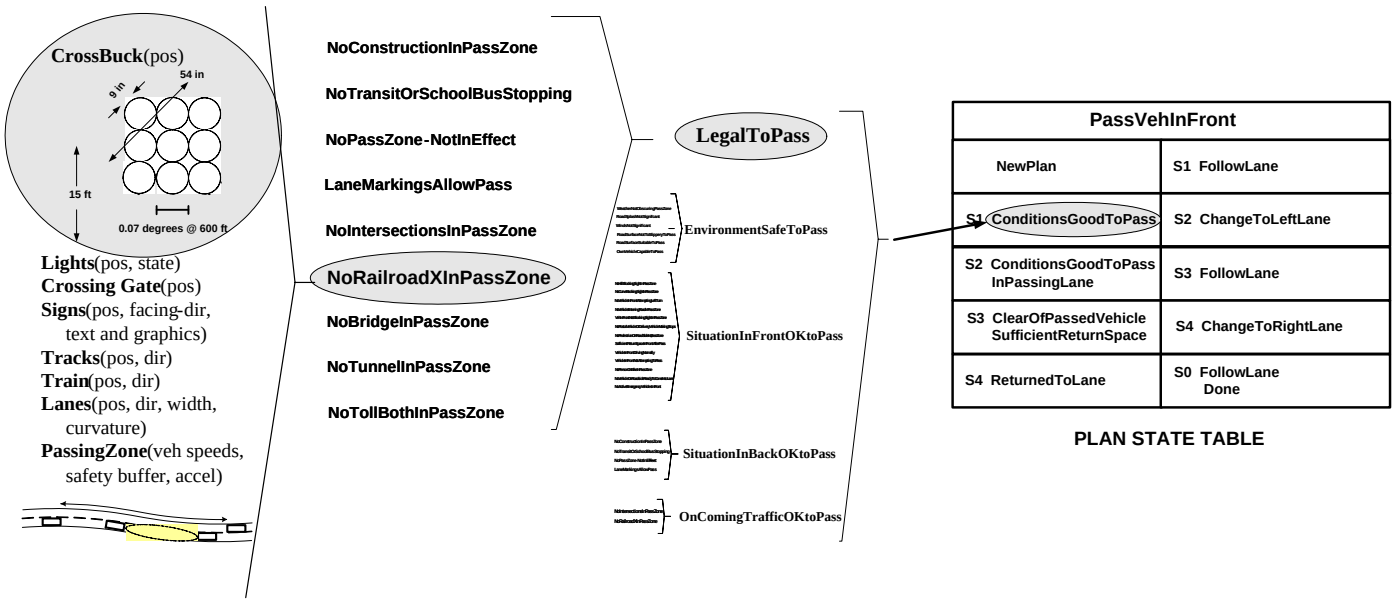


Figure 6. The "PassVehInFront" Plan StateTable encodes the task decomposition representation of its input conditions and corresponding output action subgoals. In this example, the next subgoal "ChangeToLeftLane" is chosen as the output action when the input condition of "ConditionsGoodToPass" is recognized. This figure illustrates how all of the dependencies on the world model data are derived. The high level group of situations that must be true for "ConditionsGoodToPass" to be true are identified. Here, one of these (LegalToPass) is further refined to identify all of the world model states that help define this situation. Similarly, we display detail on one of these world model states (NoRailroadXInPassZone) and the world entities, attributes, features, dimensions, and resolutions that help determine whether this state is true. One of these entities (CrossBuck sign) is further detailed in terms of the features, dimensions, and sensor resolutions required to recognize it within the distances required for the passing vehicle task.

traffic lane (the "ChangeToLeftLane" action) during the passing operation until "ConditionsAreGoodToPass". Now we have to determine what must be considered for this to be true. To do this, we review many different example scenarios to determine all of the pieces of knowledge required for all of these variations. The results are grouped by category into (in this example) five major evaluation areas. Thus, to be able to say that the "ConditionsAreGoodToPass", we first had to evaluate that each of the five sub groups were true (i.e., the conditions "LegalToPass", "EnvironmentSafeToPass", "SituationInFrontOKtoPass", "SituationInBackOKtoPass", and "OncomingTrafficOKtoPass").

In this example, we have clustered all of the rules of the road that pertain to the passing operation at this level of task detail into the "LegalToPass" sub group evaluation. We have itemized nine world states to be evaluated and named them with the identifiers such as

- "NoConstructionInPassZone",
- "NoTransitOrSchoolBusStopping", and
- "NoPassZone-NotInEffect",

These world states can now be further decomposed into the primitive world model elements we need to measure (e.g., vehicles, their speed, direction, location, lane markings, signs, railroad tracks) to determine whether these world states exist. These primitive world model elements then set the requirements for the sensory processing system we need to build to support these control tasks. Everything has been determined in the context of the individual tasks that we want the system to support.

4. Application Example

In this section, we summarize the RCS methodology, and detail an example mentioned throughout this paper pertaining to passing another vehicle on a two lane undivided road. Domain experts are consulted and play an integral part throughout this entire process. In the case of on-road driving, we are all domain experts, though many of the conditions we examine and the actions we perform are determined subconsciously.

1) **Scenario development with a domain expert:** For any task in on-road driving, we walk through detailed scenarios with domain experts to deeply understand the actions they take in certain situations, what conditions spawned those actions, and why they believed the actions were most appropriate in that situation. If possible, we try to immerse the domain expert in similar situations and have them talk through their behaviors. In the case of passing on a two lane undivided road, it is often beneficial to drive in a vehicle with the domain expert and to have them describe their process of determining when it was appropriate to pass. Specific conditions that spawn behaviors often change slightly depending on the

driver's personality and aggressiveness level, but we try to generalize the behavior to its fundamental components when encoding it in the control system.

2) **Develop the task decomposition hierarchy:** Before we can encode the knowledge needed to pass on a two lane undivided road, we must understand and build an initial, overall task decomposition hierarchy for on-road driving. This is an iterative process, and the task decomposition hierarchy often changes as new on-road driving scenarios are explored. Changes in the task decomposition hierarchy are much more frequent in the beginning, and gradually reduce in frequency as more scenarios are explored. This passing scenario is one of many scenarios that has been used to develop this task decomposition hierarchy.

3) **Determine the conditions that cause you to perform an action and the sub-actions that are needed to perform that action:** In the case of passing, the actions that need to be performed are fairly straightforward: change to left lane, follow left lane for some period of time, and change to right lane. This is shown in Figure 5. However, the conditions of when the vehicle should start this sequence of actions and when it should progress from one action to the next is much more difficult to understand.

   Let's examine the conditions when one would initiate a passing operation. In speaking with domain experts, one could decompose the conditions (that must be true to pass) into two categories: namely, that our autonomous vehicle desires to pass and that the conditions are good to pass. Only when both of these conditions are true will it initiate the passing operation. Through continued interrogation and "what-if" scenarios, we determined five conditions that must be true when it is "good to pass": 1) it is legal to pass, 2) the environmental weather and visibility conditions are conducive to passing (often related to weather conditions), 3) the situation in front of our vehicle is OK to pass (other vehicles, pedestrians, and objects in front of us do not hinder our ability to pass), 4) the situation behind our vehicle is OK to pass (the vehicle behind us is not passing or tailgating us), and 5) oncoming traffic allows us to pass safely (we have time to get around the vehicle in front of us). Each of these five sub-conditions would be recursively decomposed until we identify the objects in the environment, and their pertinent attributes, that impact the decision of whether to perform this passing action.

4) **Use the previous step to define the concepts that must be captured in the system's underlying knowledge base, and structure the knowledge base to ensure maximum efficiency for the application:** The objects and attributes discovered in the previous step sets the requirements for the knowledge base that underlies the system. Following through with the scenario of passing on a two lane undivided road, in order to evaluate the conditions mentioned in the previous step, the knowledge base must contain concepts such as:

   - other vehicles, including their speed, direction, location, and possibly intention;
   - pedestrians, including their speed, direction, location, and possibly intention;
   - lane markings, along with the type of lane marking;
   - weather conditions and visibility; and
   - signs, including the text on the sign.

   Once these concepts are captured in the knowledge base, they can be structured in such a way to ensure maximum system efficiency.

5) **Carefully evaluate all of the above objects and attributes in the context of the appropriate tasks to define resolutions, distances, and timing of the measurement of these items by the sensory processing system:** As shown in Figure 6 with the identification of the railroad crossing buck sign, we must define the sizes, shapes, relative locations, and angles to the road, distances at which they have to be identified (thereby setting resolution requirements), etc. This will yield the sensory processing specifications in terms of the world model elements that must be measured and generated. These same specifications become the performance requirements on sensory processing during test and evaluation.

5. Example Of Sensory Processing Metrics

In this section, we will examine some detailed examples of requirements for sensory processing, following through with our passing example. In particular, we will determine what it requires of the vehicle sensors to decide, at any given time and speed, if it is legal to pass.

As shown in Figure 6, for a passing operation to be legal, there cannot be:

- any construction in the passing zone,
- a transit or school bus stopping in the passing zone,
- a no-passing-zone sign in the passing zone,
- lane marking that prohibit passing,
- intersections in the passing zone,
- a railroad crossing in the passing zone,
- a bridge in the passing zone,
- a tunnel in the passing zone, or
- a toll booth in the passing zone.

Therefore, the sensory processing system must detect these items, or indicators that these items are approaching, at a distance that allows the vehicle to pass safely. In this analysis we make a few assumptions:

- the vehicle can accelerate comfortably at 1.65 m/s^2,
- our vehicle is positioned approximately one second behind the vehicle in front of it (i.e., our vehicle will be at the preceding vehicle's current position in one second traveling at constant velocity),
- our vehicle will begin merging back into its original lane when it is one car length in front of the vehicle it is passing,
- the merging operation that brings our vehicle back into its original lane will take one second, and
- the average length of a vehicle is five meters.

All of these values are variables, and can easily be changed depending on the exact situation.

With these assumptions, we calculated the distance that our vehicle would travel during a passing operation, how long it would take to travel that distance, and its final velocity assuming both vehicles have initial speeds of 13.4 m/s (30 mph), 17.9 m/s (40 mph), and 26.8 m/s (60 mph). Table 1 displays the results (we assume un-occluded visibility).

Table 1: Pertinent values for passing operation at various speeds.

Speed (m/s)	Time to Complete Pass (s)	Distance Traveled in Pass (m)	Final Velocity at End of Pass (m/s)
13.4	6.32	117.8	23.9
17.9	6.81	159.3	29.1
26.8	7.68	253.9	39.5

For the "no railroad crossing in passing zone" requirement, there are multiple markings that can indicate a railroad crossing is upcoming, such as a crossbuck just before the railroad crossing, or railroad signs at pre-defined distances before the railroad crossing. Table 2 displays the specification of how far before a railroad crossing a warning sign should be placed, what size the sign must be, and what size the letter on the signs must be, according to the Manual of Uniform Traffic Control Devices (MUTCD) [4].

Table 2: Specifications for railroad crossing signs.

Speed (m/s)	Distance from Railroad Crossing (m)	Sign Dimensions (m x m)	Letter height (m)
13.4	99	0.450 x 0.450	0.125
17.9	145	0.450 x 0.450	0.125
26.8	236	0.450 x 0.450	0.125

Considering that the railroad warning sign is a pre-defined distance before the railroad crossing, we can subtract that distance from the full passing distance shown in Table 1 to identify the forward distance our sensors must to be able to sense. These distances are shown in Table 3.

This sets the specification for how far a sensor must be able to "see" to determine if there is a railroad crossing sign in the passing zone. However, we can take this one step further and determine what the resolution of the sensors must be to read the sign. The following paragraphs examine

Table 3: Sight distance for a railroad warning sign.

Speed (m/s)	Passing Distance (m)	Warning Sign Distance (m)	Sensor Sign Distance (m)
13.4	117.8	99	18.8
17.9	159.3	145	14.3
26.8	253.9	236	17.9

the requirements of the sensor itself. We ignore the software that performs the character and object recognition task in this discussion, although we recognize that it is at least as important as the specifications for the sensors.

For a sign that needs to be read (ie, its shape and/or color do not convey its meaning), we assume that a 20x20 array of pixels hits on each letter is required to recognize it. Using simple trigonometry based on the distance to the sign and the size of its letters as shown in Table 2, we can determine that a camera with resolutions of about 0.02 degrees is needed for all three cases above.

In some cases, a warning sign is not present and the sensors must rely on recognizing a crossbuck that is immediately before the railroad crossing. In this case, we assume that we need an array of 5x5 pixel hits on the crossbuck to recognize it by shape, and that the size of the crossbuck is the standard 900x900 mm in total dimensions, as specified by the MUTCD manual. Based on this information, we would need a sensor with a resolution as shown in Table 4. Similar calculations could be performed for all other items the sensor would need to sense when determining if it is legal to pass at any given time and speed.

Table 4: Sensor sight distance for crossbuck.

Speed (m/s)	Sensor Resolution (degrees)
13.4	0.1042
17.9	0.0711
26.8	0.0406

6. Summary

Our goal is to produce a taxonomy of on-road driving behaviors which can be further analyzed to produce the set of specifications that identify the world model entities, features, attributes, resolutions, recognition distances, and locations for each separate driving task. These specifications can be used as the basis of performance metrics for sensory processing world model building. This requires the representation of two sets of domain knowledge. One is the task decomposition knowledge that defines the sequences of subtask activities for every aspect of every type of driving task. This task decomposition knowledge is encoded into ordered sets of production rules clustered by the context of the individual driving tasks. These rules consist of input conditions (present world situations) that, when matched, cause the output of the appropriate sub-task goals. The second set of domain knowledge is the detailed world state descriptions and evaluation functions required to produce the world situation

symbolic values that are used as the input transition conditions by the task decomposition rules.

We described how the 4D/RCS methodology and reference architecture was used to define the task decomposition and the resulting state tables of production rules. We then described how the input conditions of these rules were further evaluated to derive all of their dependencies on all of the corresponding world model states and primitive world entities, features, and attributes. Still using the context of the individual driving tasks, the appropriate recognition distances were factored in to attain a specification of the requirements for the sensory and world model processing necessary for each separate driving task behavior. These requirements now serve as both a requirements list for the development of the sensory and world model processing for different on-road driving tasks as well as the performance metrics against which they can be measured to assess the correctness of their operations. Finally, we anticipate that additional metrics can be developed to measure the performance characteristics of the behavior generation component (i.e., its planning and value judgment operations); this is a topic for future research.

Acknowledgements

This work was supported by DARPA's Mobile Autonomous Robotics program (PM. D. Gage). We thank Jim Albus, Elena Messina, and our other project colleagues for their continuing support of this work.

References

1. Albus, J. and et.al., "4D/RCS Version 2.0: A Reference Model Architecture for Unmanned Vehicle Systems," NISTIR 6910, National Institute of Standards and Technology, Gaithersburg, MD, 2002.

2. Albus, J. and Meystel, A., *Engineering of Mind*, John Wiley & Sons, Inc. 2001.

3. McKnight, J. and Adams, B., *Driver Education Task Analysis. Volume 1. Task Descriptions*, Human Resource Research Organization, Department of Transportation, National Highway Safety Bureau 1970.

4. U.S.Department of Transportation, F. H. A., *Manual on Uniform Trarffic Control Devices (MUTCD 2000) Millenium Edition* 2000.