Reactive Paradigms

Dr. Divya Udayan J, Ph.D.(Konkuk University, S.Korea)

Department of CSE
Amrita School of Engineering, Amritapuri Campus,
Amrita Vishwa Vidyapeetham
Email: divyaudayanj@am.amrita.edu
Mobile: 9550797705

- The reactive paradigm is important to study

Because robotic systems in limited task domains are being constructed using reactive architectures.

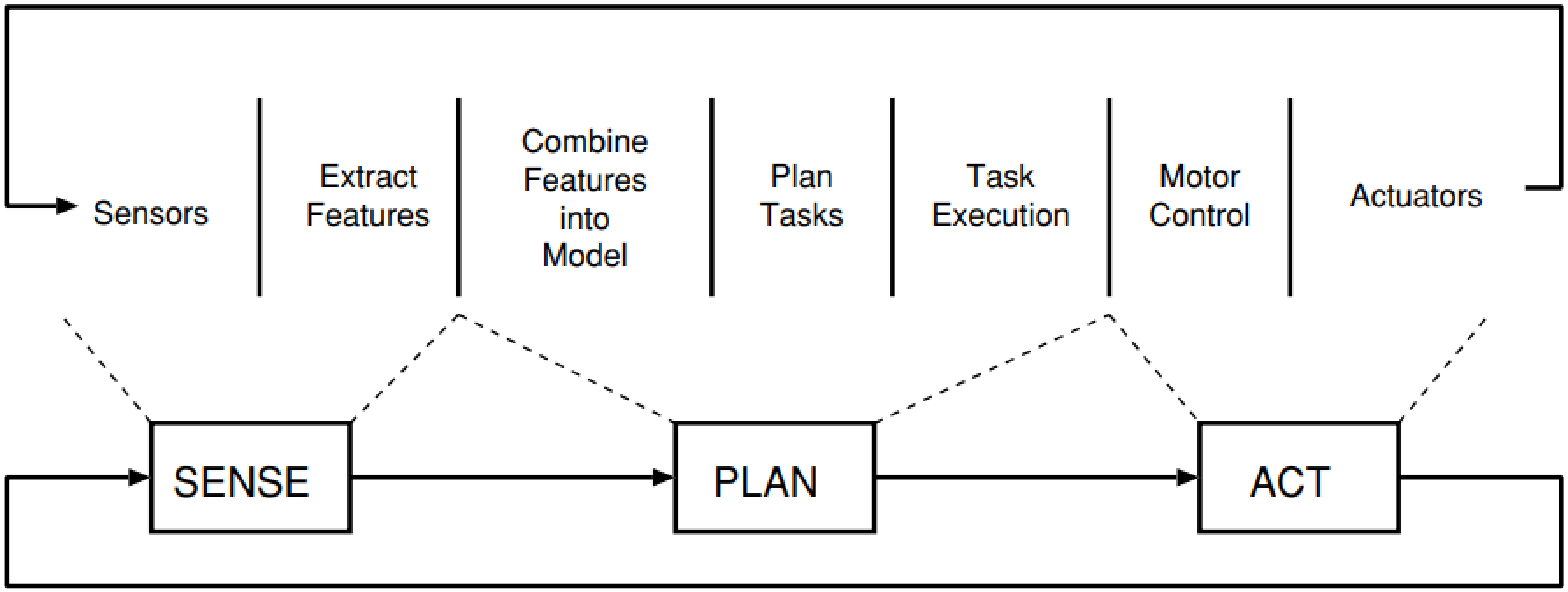


Figure Horizontal decomposition of tasks into the S,P,A organization of the Hierarchical Paradigm.

- Create parallel tracks of more advanced behaviors
- The parallel tracks can be thought of layers, stacked vertically.
- Each layer has access to sensors and actuators independently of any other layers.
- If anything happens to an advanced behavior, the lower level behaviors would still operate.
- human brain and breathing example

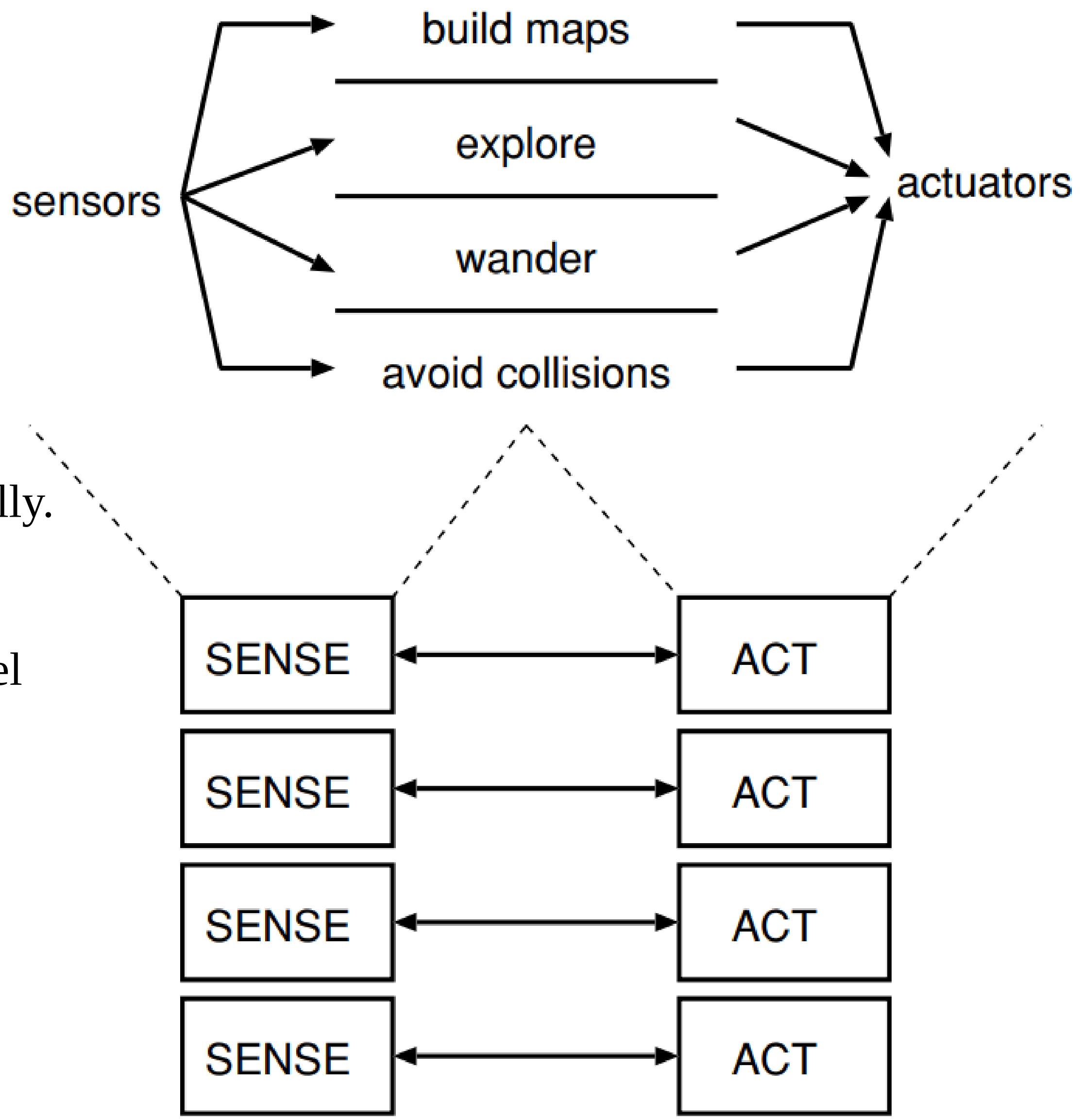


Figure Vertical decomposition of tasks into an S-A organization, associated with the Reactive Paradigm.

Characteristics and connotations of reactive behaviors

- The primary connotation of a reactive robotic system is that it executes rapidly.
- The tight coupling of sensing and acting permits robots to operate in real-time, moving at speeds of 1-2 cm per second.
- Behaviors can be implemented directly in hardware as circuits, or with low computational complexity algorithms (O(n)).
- This means that behaviors execute quickly regardless of the processor.
- Behaviors execute not only fast in their own right, they are particularly fast when compared to the execution times of Shakey and the Stanford Cart.
- A secondary connotation is that reactive robotic systems have no memory, limiting reactive behaviors to what biologists would call pure stimulus-response reflexes.

Characteristics of Reactive Paradigm

- Robots are situated agents operating in an ecological niche.
- Behaviors serve as the basic building blocks for robotic actions, and the overall behavior of the robot is emergent.
- Only local, behavior-specific sensing is permitted.
- These systems inherently follow good software design principles.
- Animal models of behavior are often cited as a basis for these systems or a particular behavior.

Programming by behaviour

- Constructing a robotic system under the Reactive Paradigm is often referred to as programming by behavior, since the fundamental component of any implementation is a behavior.

Advantages

- Behaviors are inherently modular and easy to test in isolation from the system (i.e., they support unit testing).
- Behaviors also support incremental expansion of the capabilities of a robot. A robot becomes more intelligent by having more behaviors. The behavioral decomposition results in an implementation that works in real-time and is usually computationally inexpensive.
- If the behaviors are implemented poorly, then a reactive implementation can be slow. But generally, the reaction speeds of a reactive robot are equivalent to stimulus-response times in animals.
- Behaviors support good software engineering principles through decomposition, modularity and incremental testing.
- If programmed with as high a degree of independence (also called *low coupling*) as possible, and *high cohesion*, the designer can build up libraries of easy to understand, maintain, and reuse modules that minimize side effects.
- Low coupling means that the modules can function independently of each other with minimal connections or interfaces, promoting easy reuse. Cohesion means that the data and operations contained by a module relate only to the purpose of that module.

Example: An abstract follow-corridor behavior

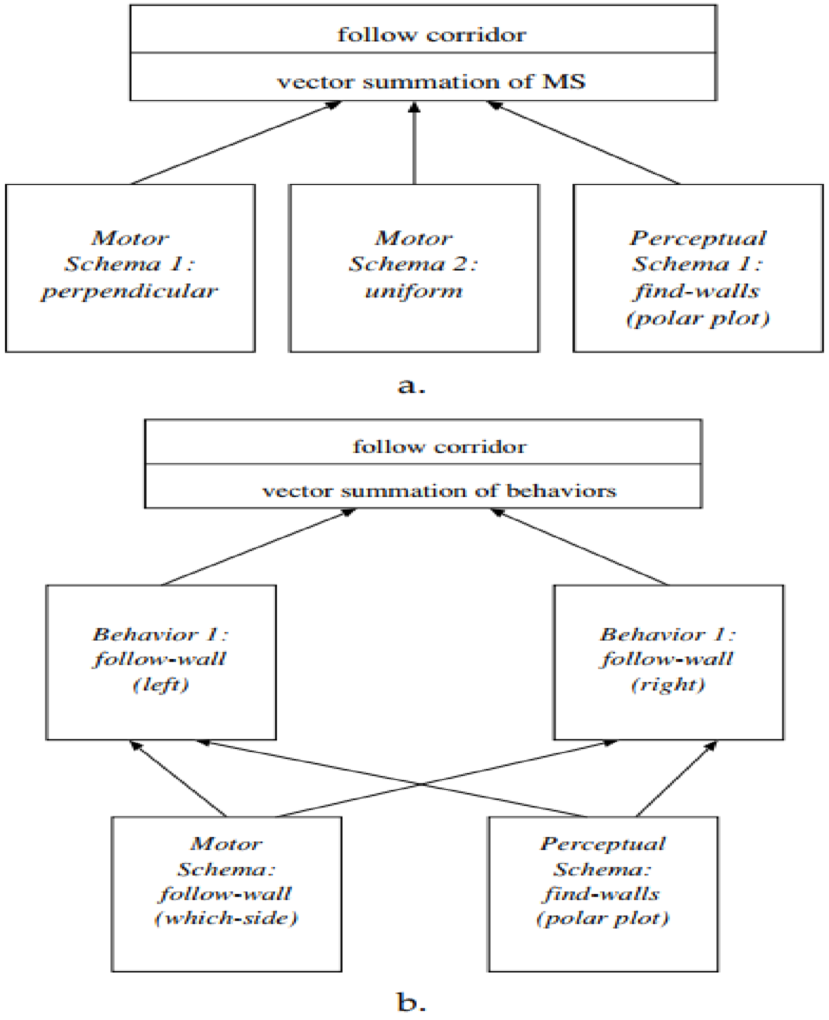
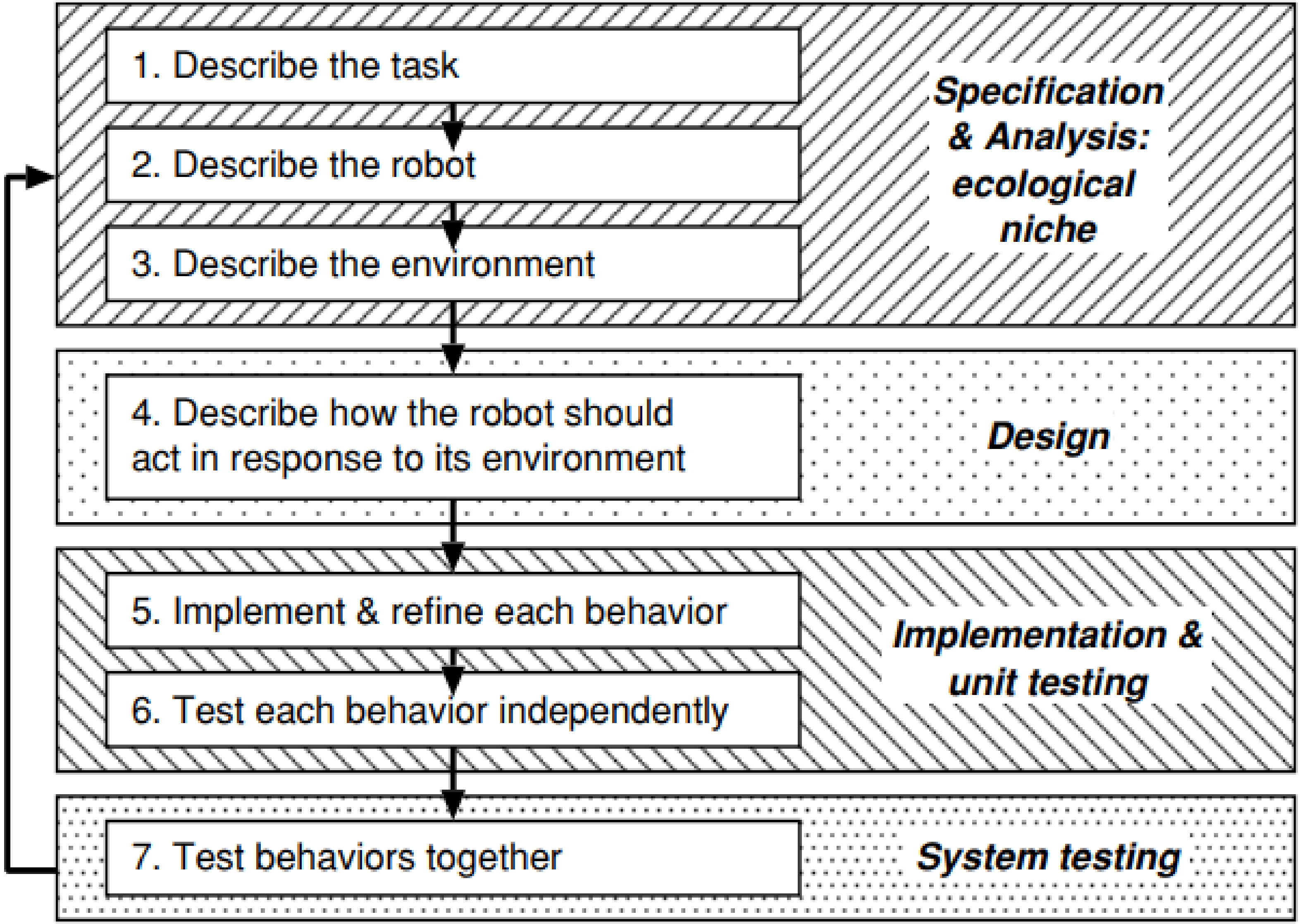


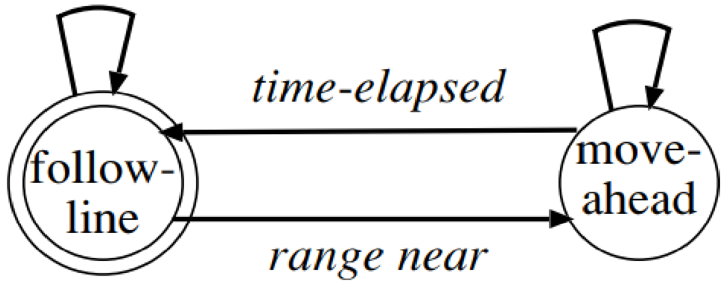
Figure Class diagrams for two different implementations of `follow_corridor`: a.) use of primitive fields, and b.) reuse of fields grouped into a `follow wall` behavior.



Steps in designing a reactive behavioral system, following basic software engineering phases.

Finite State Automata

- *Finite state automata (FSA)* are a set of popular mechanisms for specifying what a program should be doing at a given time or circumstance.
- The FSA can be written as a table or drawn as a *state diagram*, giving the designer a visual representation.



a.

$$M : K = \{\text{follow-line, move-ahead}\}, \Sigma = \{\text{range near, range far}\},$$
$$s = \text{follow-line}, F = \{\text{follow-line, move-ahead}\}$$

q	σ	$\delta(q, \sigma)$
follow-line	range near	move-ahead
follow-line	range far	follow-line
move-ahead	time remaining	move-ahead
move-ahead	time elapsed	follow-line

b.

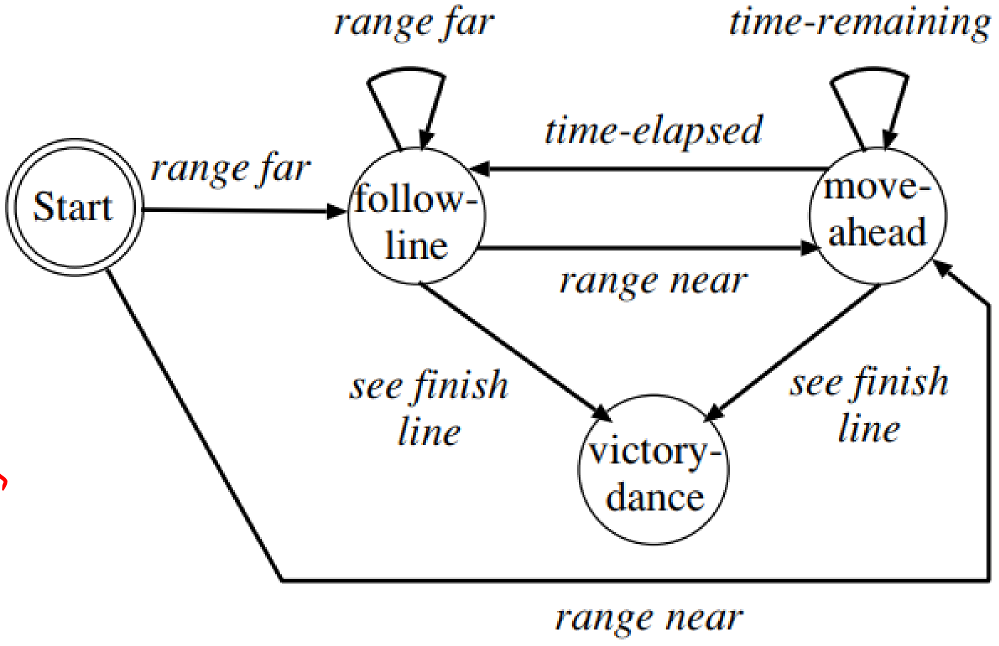
Figure A FSA representation of the coordination and control of behaviors in the UGV competition: a.) a diagram and b.) the table.

$q \rightarrow$ any state

$K \rightarrow$ list of states

$s \rightarrow$ start state

$\sigma \rightarrow$ input | behaviors releases



a.

$$M\colon K = \{\text{follow-line, move-ahead}\}, \Sigma = \{\text{range near, range far}\},$$
$$s = \text{follow-line}, F = \{\text{follow-line,move-ahead}\}$$

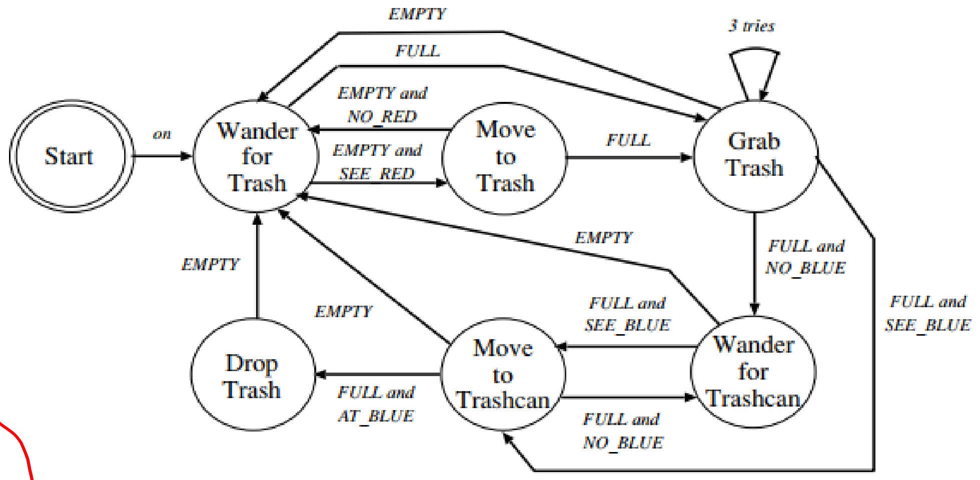
q	σ	$\delta(q,\sigma)$
follow-line	range near	move-ahead
follow-line	range far	follow-line
move-ahead	time remaining	move-ahead
move-ahead	time elapsed	follow-line

b.

Figure An alternative FSA representation of the coordination and control of behaviors in the UGV competition: a.) a diagram and b.) the table.

δ

Transition function- which specifies what state the robot is in after it encounters an input stimulus



a.

K = {wander for trash, move to trash, grab trash, wander for trash can, move to trash can, drop trash }, Σ = {on, EMPTY, FULL, SEE_RED, NO_BLUE, SEE_BLUE, AT_BLUE}, $s = Start$, $F = K$

The set of stimulus or affordances That can be recognized by the robot is represented by Σ

σ

q	σ	$\delta(q, \sigma)$
start	on	wander for trash
wander for trash	EMPTY, SEE_RED	move to trash
wander for trash	FULL	grab trash
move to trash	FULL	grab trash
move to trash	EMPTY, NO_RED	wander for trash
grab trash	FULL, NO_BLUE	wander for trash can
grab trash	FULL, SEE_BLUE	move to trash can
grab trash	EMPTY	wander for trash
wander for trash can	EMPTY	wander for trash
wander for trash can	FULL, SEE_BLUE	move to trash can
move to trash can	EMPTY	wander for trash
move to trash can	FULL, AT_BLUE	drop trash
drop trash	EMPTY	wander for trash

b.

The stimulus is represented by arrows. Each arrow represents the releaser for a behaviour. The new behaviour triggered by the releaser depends on the state the robot is in.

Figure A FSA for picking up and recycling Coke cans: a.) state diagram, and b.) table showing state transitions.

FSA table is an extension of behavioral table

$M = \{K, \Sigma, \delta, s, F\}$