The Hierarchical Paradigm

A robotic paradigm is defined by the relationship between three primitives [1]:

- SENCE
- PLAN
- ACT

and by the way the sensory data is processed and distributed through the system.

The Hierarchical Paradigm (HP) is historically the oldest method of organizing intelligence in mainstream robotics. The paradigm dominated robot implementations from 1967 with the inception of the first AI robot, Shakey Figure 1 (see wikipedia entry: https://en.wikipedia.org/wiki/Shakey_the_robot), up until the late 1980's when the Reactive Paradigm emerged [1].

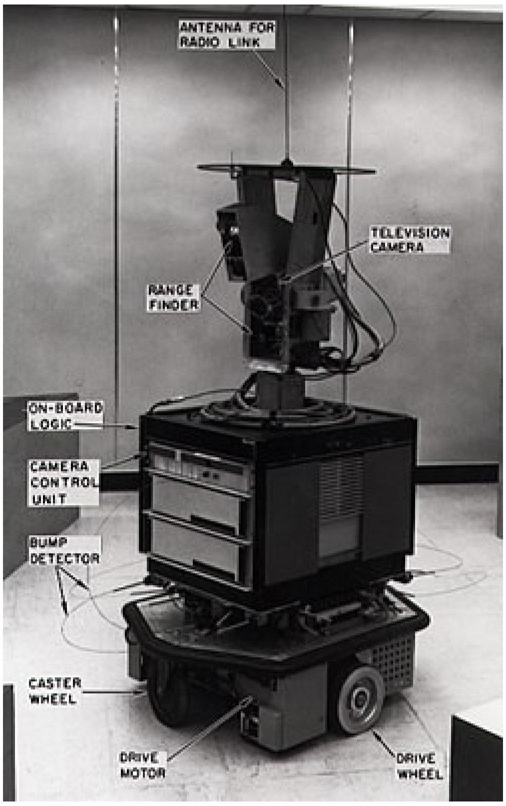

Figure 1: Shakey in 1972. Figure from wikipedia

The HP pattern is sequential with respect to the three primitives mentioned above. First the robot senses the world and constructs a global world map. Secondly, the robot plans all the actions needed to reach the goal. Thirdly, the robot acts to carry out the plans devised in the previous step. The sequence is repeated in a cycle. This is shown schematically in Figure 2.

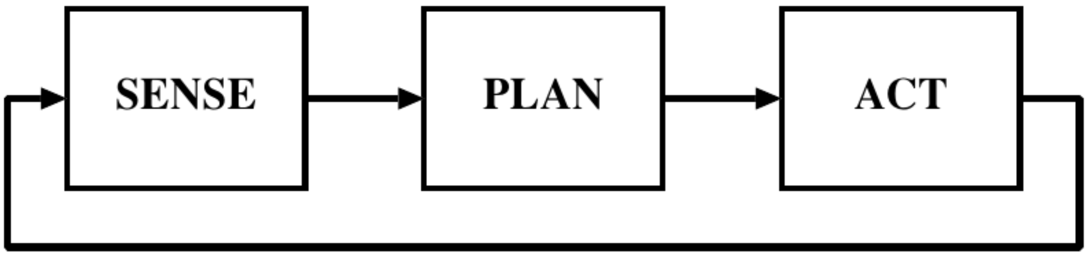

Figure 2: SENSE-PLAN-ACT organization of Hierarchical Paradigm

HP is monolithic in the sense that all sensor observations are fused into one global data structure which the planner accesses. This global data structure is generally referred as the world model.

Typically, the World Model contains information about

- the representation of the environment the robot is operating in. e.g. a map of the building
- sensing information
- cognitive knowledge that might be needed to accomplish a task.

A world model component can be characterized as either passive or active [2]. The former is more like a data store and may lack semantic understanding of the stored data. Hence, by itself it cannot perform physics related computations on the data it holds. Therefore, it cannot actively predict the state of the world given specific inputs. The active world model may incorporate kinematic and dynamic models of the objects it contains and be able to evolve beliefs of the world states when it is given a sequence of inputs [2].

Possibly the two best known architectures implementing the HP are the Nested Hierarchical Controller (NHC) and the NIST Realtime Control System (RCS) with its teleoperation version for JPL called NASREM [1].

1 Nested Hierarchical Controller

The NHC has components that are easily mapped to the aforementioned three primitives i.e. SENSE-PLAN-ACT. This is shown in figure 3. The first step is for the robot to gather observations from its sensors. By combining these observations, the robot forms the so called World Model data structure. The model can also contain *a priori* knowledge about the world. After the World Model has been updated, the robot can PLAN what actions it should take.

The PLAN activity is performed by three sub-modules namely:

- Mission Planner
- Navigator
- Pilot

As shown in figure 3 each of these three modules requires access to the World Model.

The Mission Planner locates where the robot is (by consulting the World Model and/or accessing an internally stored map or constructing the map and localizing simultaneously e.g. by using SLAM) and where the goal is. It also provides the translation of the this mission into terms that other functions can understand. The Navigator takes this information and generates a path (typically free of obstacles) from the current robot location to the goal. The generated path may be described as a set of waypoints or straight lines to be followed by the robot.

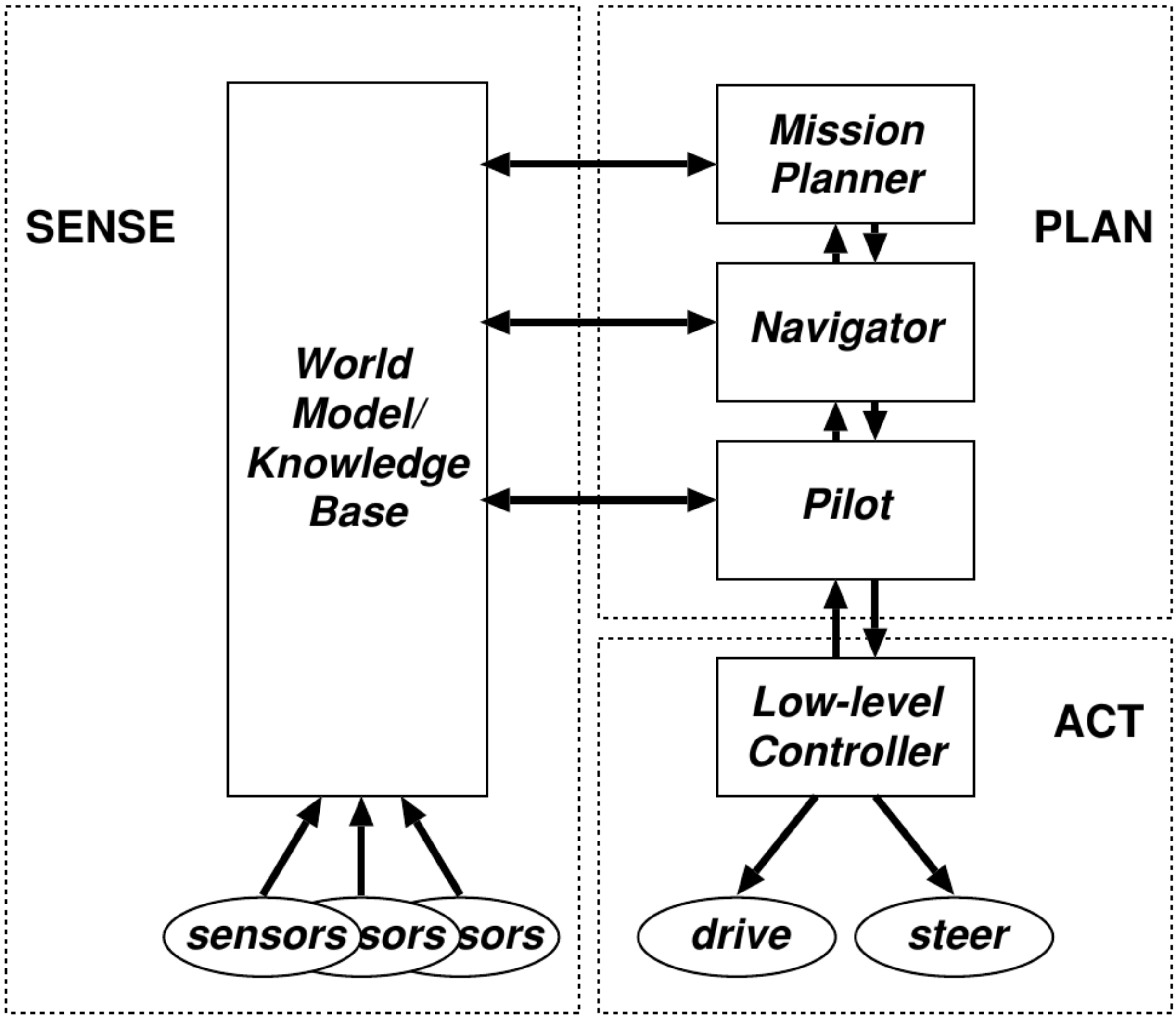

Figure 3: Nested Hierarchical Controller. Figure from [1].

The Pilot takes the first straight line or path segment and determines what actions the robot has to do to follow the path segment. For instance, the robot may need to turn around to face the waypoint before it can start driving forward. The Pilot module altogether is responsible for generating specific actions for the robot to do in order to reach its goal. These actions are subsequently translated into actuator control signals by the Low-Level controller in the ACT portion of the architecture.

After the Pilot gives the Low-Level Controller commands and the controller sends actuator signals, the robot polls its sensors again. The World Model is updated. However, the entire planning cycle does not repeat. Since the robot has a plan, it doesn't need to rerun the Mission Planner or the Navigator. Instead, the Pilot checks the World Model to see if the robot has drifted off the path sub-segment (in which case it generates a new control signal), has reached the waypoint, or if an obstacle has appeared. If the robot has reached its waypoint, the Pilot informs the Navigator. If the waypoint isn't the goal, then there is another path sub-segment for the robot to follow, and so the Navigator passes the new sub-segment to the Pilot. If the waypoint is the goal, then the Navigator informs the Mission Planner that the robot has reached the goal. The Mission Planner may then issue a new goal, e.g., Return to the starting place. If the robot has encountered an obstacle to its path, the Pilot passes control back to the Navigator. The Navigator must compute a new path, and sub-segments, based on the updated World Model. Then it gives the updated path sub-segment to the Pilot to carry out.

2 References

1. Murhy R. R., ***Introduction to AI robotics***, MIT Press, 2000.
2. Behere S., Torngren M. ***A functional architecture for autonomous driving***
3. Hierarchical control system ***https://en.wikipedia.org/wiki/Hierarchical_control_system***