Coordination of Action and Perception in a Surveillance Robot

James L. Crowley

LIFIA (IMAG)
Institut National Polytechnique de Grenoble
Grenoble, France

Abstract:

This paper describes a system in which twin hierarchies for navigation and perception are controlled by a knowledge based supervisor. At the top level, these hierarchies contain collections procedures which implement the "skills" of the system in locomotion and spatial reasoning. The structure of these hierarchies, and their integration is described.

Perception and navigation are controlled and monitored by a knowledge based "supervisor". The supervisor decomposes a mission plan into subgoals and then uses the perception and navigation procedures to accomplish these subgoals. The organization of the rule base for the supervisor is described, and a mission planning is illustrated with a simple example.

1 Introduction

This paper describes a technique for coordinating action and perception within a mobile surveillance robot. The surveillance robot employs an architecture composed of twin hierarchies for navigation and perception controlled by a knowledge based supervisor. Such an architecture has allowed us to explore he interface between heuristic and algorithmic programming techniques for perception and action. Our results illustrate that that most of the so called low-level tasks in perception and navigation are algorithmic in nature. On the other hand, at the "highest levels", the decisions as to which actions to perform is based on knowledge relevant to each situation. Such decisions are heuristic in nature, and the relevant knowledge is naturally encoded as production rules organized as contexts.

2 The System Architecture

The control of navigation and perception in an unknown environment present difficult problems. Earlier projects [Crowley 85a] in which this top level was implemented as an algorithmic process led to systems with rigid behavior. A major theme of this research is that control at the top level requires the application of a body of poorly organized knowledge. Thus we have developed a system architecture in which a production system controls the hierarchies of perception and navigation.

2.1 System Organization

The system organization is illustrated in figure 1. The system is composed of twin hierarchies for navigation and for perception. These hierarchies are organized as a set of levels according to the abstraction of the information which is processed.

Motors and Sensors: At the lowest level, each hierarchy asynchronously processes raw signals. In the navigation hierarchy, this processing involves closed loop control of the motors to maintain a specified velocity, as well as capture of proprioceptive sensor signals for estimating position and velocity. In the perception hierarchy, processing involves acquiring sensor signals and converting these to an initial symbolic representation in vehicle coordinates.

1.1 Scientific Issue: Coordinating Action and Perception

The research reported in this paper is part of an effort to develop a family of intelligent mobile surveillance robots. Such robots are a form of "intelligent agent", operating in and interacting with the real world.

The development of intelligent agents poses important scientific questions for robotics and artificial intelligence. Among these questions is the relation between heuristic knowledge and algorithmic "skills". In particular, where and how should the boundary occur between heuristic and algorithmic programming in an intelligent agent? A second problem concerns the relationship between planning and plan execution: How much of a plan can be developed in advance? How should the agent respond when the plan does not succeed?

To explore these issues we have posed these problems within a specific application domain: The planning and execution of surveillance missions for a military robot in an urban environment. Systems for dynamic world modeling and for navigation and locomotion have been developed for a surveillance robot within our laboratory [Crowley-Coutaz 86]. Knowledge based coordination of action and perception has been investigated using a simulated version of our surveillance robot and a data base of simulated urban environments. The simulation imitates the functional behavior of our surveillance equipped with a rich set of sensor in a rich and changing environment.

1.2 Solution: Knowledge Based Coordination

Our investigation has led to an architecture in which a production system sits at the top of twin hierarchies for perception and navigation. The lower levels of these hierarchies assure integrated control at the level of the vehicle, and integration of signals from the environmental sensors into a composite model of the surfaces in the immediate environment.

The composite model and the vehicle level controller are data driven processes at roughly the same level of abstraction within the two hierarchies. Above this level are a number of perception and navigation abilities which we refer to as the "action level" in the hierarchies. These action-level abilities correspond roughly to "skills" in a human. Attempts to implement these abilities as bodies of rules within the production system soon convinced us that such an approach was not appropriate. Skills are fundamentally algorithmic in nature and are better suited for programming in a traditional programming language.

The result is an architecture in which algorithmic procedures for navigation and perception tasks are "triggered" by rules within the production system. The navigation tasks at this level often require a considerable amount of information from the perception hierarchy. As a result a considerable amount of communication between navigation and perception occurs without the awareness of the production system.

In the section below we describe the system architecture. We then describe the organization of knowledge for the planning and plan execution systems.

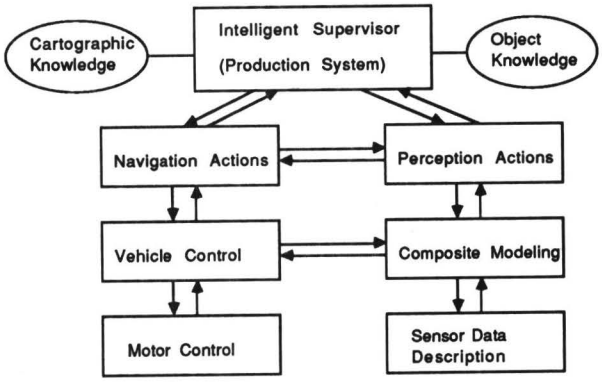

Figure 1. The System Architecture: Parallel Architecture for Navigation and Perception Controlled by an Intelligent Supervisor.

Vehicle Control and Composite Model: At an intermediate level, both hierarchies represent their information at a level of abstraction based on the vehicle and its environment. At the center of the navigation hierarchy is a vehicle level controller. This controller accepts asynchronous commands to move and turn the vehicle. The vehicle level controller also maintains an estimate of the position and velocity of the robot and shares this with the perception hierarchy. The perception hierarchy projects the description of new sensor signals into a common coordinate system, and uses the projected information to update a "unified" composite model of the environment. As a side effect of the update process, errors in the estimated position are detected and relayed to the vehicle controller.

Action Level: A collection of algorithmic processes operate on the information provided by the vehicle controller and the composite model. The navigation hierarchy includes algorithmic processes for such tasks as following a road, following a wall, or traveling towards a distant beacon. These processes necessarily depend on asynchronous access to perceptual information, made available by a set of interface procedures for the composite model. A significant amount of the coordination between action and perception occurs between algorithmic procedures at the level of actions. In human terms, these action level procedures correspond to learned "skills" such as driving a car, or understanding the environment in terms of rigid objects.

Supervisor Level: Controlling the twin hierarchies for navigation and perception involves selecting the appropriate perception and navigation actions in order to accomplish some set of high level goals. Such an activity is not easily organized as an algorithm; each situation implies a body of appropriate knowledge. Such knowledge is naturally expressed as rules, organized into "contexts". Within each context, rules are triggered by internal facts which represent things such as goals, external events, or descriptions of the environment.

In the following sections we describe briefly the structure of each part of the system.

3 The Perception System

The robot's sensors are divided into two broad classes: environmental sensors and surveillance sensors. Environmental sensors provide information for describing the structure of the local environment, while surveillance sensors are used to detect the presence of intruders. Surveillance sensors operate under direct control of the supervisor. Environmental sensors operate autonomously and continuously.

The data from the environmental sensors drives a process which constructs and maintains a composite model of the environment. The composite model describes geometric, dynamic and surface features within the local environment. The composite model serves as an interface between data driven and knowledge driven processes for perception.

3.1 Integrating Sensor Data: the Composite Model

The heart of perception system is a dynamically maintained data structure called the the composite model [Crowley 85] as well as with 3-D vision [Crowley 87]. The composite model is a geometric description; it does not contain labels for "recognized" objects. Interpretation of the structures within the composite model as known objects is accomplished by the supervisor using algorithmic procedures at the action level.

Parts of the matching and update processes for the composite model have complexities on the order of the square of the number of elements. To keep the cycle time fast, it is necessary to restrict the contents of the composite model to a few tens of primitives. Elements are quickly removed from the composite model when their presence is not reinforced by either the sensor signals or the needs of the task level processes. This purging is mediated by a recency mechanism. Older elements are purged from the composite model to restrict the number of elements to a fixed limit.

3.2 Action level Interface to the Composite Model

The contents of the composite model are never directly available to the other parts of the system. Instead the contents are accessible through a set of five interface procedures. Three of these processes are described in more detail in [Crowley 87].

Visible: Given two points, A and B, return the identity of the surface patch closest to point A which intersects the line. A return value of NIL indicates that the point B is Visible from the point A.

Correspond: Given a primitive element, P, with a particular position and orientation, as well as a tolerance for position and orientation, return the identity of the primitive element in the model which "best" corresponds to the primitive. A return value of NIL indicates that no such primitive can be found in the model.

FreePath: Given two positions A and B, as well as a tolerance, indicate whether the robot may safely travel from A to B without coming within the tolerance of an object. Return either NIL or the identity of the first object which the robot might strike.

FindPrimitive: Given a primitive element with a set of position independent attributes, and uncertainty tolerances for each attributes, return a list of all occurrences of the the primitive in the composite model.

FindObject: Given a description of an object as a composition of primitives, return a list of all such objects in the composite model.

3.3 Surveillance Sensors

The system described below employs three steerable sensors for detecting intruders: An infrared sensor, a micro-wave radar and an ultra-sonic ranging device. Operation of the surveillance sensors are implemented as action level procedures. The operation commands to the sensor includes a firing angle. The result of sensor operation is immediately returned to the supervisor. The supervisor can complete the information with either the composite model or with other sensors.

4 The Navigation System

The navigation hierarchy has an organization which is parallel to that of the perception hierarchy. Each level in this hierarchy involves an independent control loop. These loops can communicate asynchronously with the control variables for each loop specified by the next higher loop.

4.1 Motor Control

At the lowest level of the navigation hierarchy, motor control circuits control the state vectors of angular position and velocity for the motors. The controller also captures and integrates the encoder signals to make produce a cumulative encoder count. The motor controller provide asynchronous control, which makes it possible to maintain an estimate of the position of the robot as it moves, as well as to adjust the direction of travel independent of the speed.

4.2 Vehicle Level Control

The vehicle level controller coordinates the motor controllers to achieve a desired vehicle behavior, expressed in world coordinates. Navigation tasks which involve following linear structures such as roads or walls may be formulated as a process of independently controlling forward displacement and orientation and their derivatives [Wallace et. al. 1985]. The vehicle level controller must also maintain and make available an estimate of the position and orientation of the vehicle.

The command set at the level of the vehicle are :

```
MOVE      [v [, d [,a]]]
TURN      [v [, d [,a]]]
STOP
GetEstimatedPosition( x, y, angle)
CorrectEstimatedPosition( Δx, Δy, Δangle)
```

Commands may be received at any time and are executed immediately, superceding previous versions of the same command. The commands MOVE and TURN are implemented independently. The parameters of MOVE and TURN refer to displacement (d) and its first and second derivatives (v and a).

4.3 Action Level Procedures for Navigation

Control of local vehicle movements is inherently an algorithmic process. This level of control corresponds to the "local navigation" described in [Crowley 85]. Our system currently uses five such procedures. The first is based on the use of estimated position maintained by the vehicle level controller. The remaining three procedures are implemented using a form of direct pursuit. These last three tasks differ primarily in the way they use perception to select the target point.

Straight-Line Travel: The procedure is given a point in its local environment expressed in an external coordinate system. After verifying that a free path exists to the point, the robot turns toward the point and then travels in a straight line.

Pursue: The robot chases after a possibly moving target specified by the supervisor, maintaining a specified distance behind the target. The procedure calls the perception system to determine the position and velocity of the target. This function can also be used to travel towards a distant landmark or beacon.

WallFollowing: The robot travels a specified tolerance to the right or left of a wall for a specified distance. The target point is determined by projecting a target point in front of the vehicle, and then measuring the perpendicular distance to the wall, as well as the orientation of the wall at that point. These values are used to compute the target point. The procedure also uses function Freepath to assure that the path is not blocked by an obstacle. An early version of this procedure is described in [Crowley-Coutaz 86].

RoadFollowing: Based on the techniques described in [Wallace et. al. 1985]. The robot detects the center line of the road at a distance r, and uses the follow function to dynamically adjust its orientation. Road following is not currently implemented on our vehicle.

The choice of the appropriate local navigation procedure depends on the the local environment. Such choice is best stated as a set of heuristic rules, depending on knowledge about the environment and the mission of the robot. In our surveillance robot, each route in the cartographic knowledge base contains a suggested local navigation mode for traveling along the route. The supervisor reads the suggested navigation mode from the cartographic data base and calls the appropriate local navigation procedure. When a navigation procedure is unable to continue, the procedure halts and notifies the supervisor.

5 The Supervisor for a Surveillance Robot

The behavior of the supervisor is "goal oriented"; The supervisor is given a mission as a sequence of high level tasks to accomplish. The mission is decomposed into a sequence of lower level tasks by a planning stage. During execution, the plan is adapted and modified as needed in order to accomplish the mission goal.

The supervisor is implemented as a production system using the OPS-5 language [Brownston et. al. 1985]. The supervisor's rule base is organized as contexts according to the tactical situation and the current task. The right hand side of OPS-5 rules may include function calls in common lisp. This provides the interface to algorithmic tasks for navigation and perception as well as access to the cartographic and object data bases.

A mission for the surveillance robot is specified as a set of abstract navigation and surveillance tasks. The supervisor decomposes the mission into a sequence of sub-tasks to verify that sufficient information is available to accomplished the mission, and to verify that the robot can satisfy the stated constraints on time and fuel consumption. If the robot is unable to perform the mission, the human operator is informed and requested to re-specify the mission.

5.1 Cartographic and Object Knowledge

Much of the knowledge which is needed to plan and execute navigation tasks is cartographic in nature. Such knowledge is organized naturally as a network of places and routes. A cartographic data base, composed of places and routes is implemented as a network of LISP structures accessible from LISP functions. The supervisor can recall the contents of a place or route by calling the LISP functions GetPlace or GetRoute. The result is the creation of an OPS-5 object containing the attributes of the route or place. We have recently begun experiments with automatic construction of rules which suggest sequences of routes between non-adjacent places. These rules "cache" the results of planning for frequently traveled routes.

A place contains the following fields:

Name: An alpha-numeric name for the place.
Location: A cartesian location for the place.
Tactical Zone: The tactical situation at the place.
Adjacent Places: A list of pairs (Place, Route) for adjacent places

A route is composed pf:

Name: A name for the route.
NavigationMode: A suggested navigation procedure for this route.
Length: The length of the route.
Maximum Speed: The maximum speed for the route.
Efficient Speed: An efficient speed for the route.
Tactical Zone: The current tactical zone.
Adjacent Places: The pair of places connected by the route.

Each place and each route in the cartographic knowledge base are labeled with an attribute which identifies the tactical zone for that

place or route. The tactical zone, in turn, is an important factor in determining the robot's behavior. Each zone contains a risk factor, which is multiplied by the time spent in the zone, and used as a constraint in mission planning.

5.2 Operational Knowledge

The supervisors behavior is organized as a set of modes. Modes are specified to the robot during mission planning. The current set of modes include:

Efficient: The robot travels at a rate which optimizes the consumption of fuel. No surveillance activity occurs. The optimum speed for each route is available as an attribute of the route in the cartographic knowledge base. The robot will not enter dangerous tactical zones while traveling in efficiently mode.

Rapid: The robot travels as fast as possible. No surveillance activity occurs. The maximum speed is an attribute in the description of the route in the cartographic knowledge base. The robot will not enter dangerous tactical zones while traveling in rapid mode.

Discrete: The robot seeks to avoid detection. It will not enter dangerous or hostile zones. It travels slowly in unknown zones to minimize noise. It travels efficiently in friendly zones. In discrete mode the robot maintains a surveillance activity, paying particular attention to corners. If an intruder is detected, the robot will seek to move so that a barrier is between it and the intruder. It will notify base of the presence of the intruder, and then either monitors the intruder in one of the detection modes described below or plan a path to avoid the intruder depending on the mission instructions.

Reconnaissance: In friendly zones the robot travels efficiently with no surveillance. In unknown, forbidden, hostile, or dangerous zones, the robot travels at a speed which most efficient for surveillance. In the case of a detection the robot notifies the base and enters one of the detection modes described below.

5.3 Mission Specification and Planning

A mission is specified to the robot as a set of pseudo-natural language sentences which describe tasks to be accomplished. The tasks are either navigation tasks, surveillance tasks, or combinations of both. The following is a sample mission definition.

Mission Start: Time = 9:00, Fuel = 20, Maximum Risk =100
1) If detection of person use mode warn.
2) If detection of car use mode monitor
3) Go to place A in mode discrete
4) Go to place B in mode surveillance.
5) Remain at place B in mode surveillance.
6) At time=10:00 Return to place Base in mode surveillance.

The mission is planned by assembling a sequence of actions for accomplishing each task. Navigation tasks in a known environment are best planned using graph search over a network of decision points. Surveillance tasks are easily inserted in such a framework.

A backward chaining search procedure is used to develop a tree of alternative mission plans. This search procedure is based on a version of the GraphSearch algorithm described in [Nilsson 80]. The algorithm has been modified to use rule based knowledge at the two critical stages of generating the list of successive actions, and selecting the next potential action to investigate. A similar modification may be found in a paper by Mostow [Mostow 83].

Each action carries a vector of constraint values representing elapsed fuel consumption, time, risk, and distance. The selection rules for considering an action are based on the sum of the current constraint vector plus the constraint vector that would be generated for straight line travel in efficient mode in f The current rule base tries first to minimize fuel consumption, and thus prefers efficient travel. By changing rules, paths which preferentially minimize time, risk, distance traveled or linear combinations of constraints can also be constructed.

A major use of the planning process is to signal impossible tasks. If a specified task can not be completed within the allowed constraints, the operator is notified. The problem can be remedied by increasing the value of the maximum constraint or by changing the mission definition.

The first navigation task in the mission given above was: "Go to place A". This task generates the set of sub-tasks:

StraightLine to Place 0 in mode efficient
FollowRoad to Place 1 in mode efficient
StraightLine to Place 2 in mode efficient
StraightLIne to Place 3 in mode efficient
StraightLine to Place A in mode surveillance

Although, the mission plan contained a navigation mode, the final choice of navigation modes is determined by the tactical situation. Thus although navigation mode efficient was specified for travel to place A, mode surveillance is automatically chosen for travel between Place 3 and Place A because this is marked as forbidden. Such a change in navigation mode may also occur if the tactical situation for the current route or place changes during mission execution.

6 Summary and Conclusion

This paper has described a system which coordinates action and perception in a mobile surveillance robot. An architecture was presented in which twin hierarchies for navigation and for perception are controlled by a production system. Much of the functionality of the hierarchies is provided by action-level procedures at the highest levels of the hierarchies. The procedures for navigation, in particular, make heavy use of procedures within the perception hierarchy. An important source of coordination between action and perception is thus provided by these procedures.

The task level control in the system is provided by a supervisor implemented as a production system. The supervisor determines the choice of navigation and perception procedures according to the mission plan and the local environment. The mission plan is developed by the supervisor in a planning stage before the mission begins. This planning stage serves both to decompose the mission into sub-goals which can be translated into procedure calls to the task level procedures and to verify that the mission can be accomplished within the stated constraints.

Bibliography

[Brownston85] Brownston, L., et. al., Programming Expert Systems in OPS-5, Addison Wesley, Reading Mass., 1985.

[Crowley 85] Crowley, J. L. , "Navigation for an Intelligent Mobile Robot", IEEE Journal of Robotics and Automation, Vol 1 (1) March 1985.

[Crowley-Coutaz 86] Crowley, J. L. and J. Coutaz, "Navigation et Modelisation pour un Robot Mobile", Technique et Sciences en Informatique, Octobre/Novembre 1986. (in French)

[Crowley 87] Crowley, J. L. "Using a Composite Surface Model for Perceptual Tasks", 4rh IEEE Conference. on Robotics and Automation, Raleigh, N.C. , April, 1987.

[Nilsson 80] Nilsson, N. J. Principles of Artificial Intelligence, Tioga Press, 1980.

[Mostow 83] Mostow, D. J. "Machine Translation of advice into Heuristic Search Procedures", in Machine Learning: An A I Approach, ed. R. Michalski, et. al., Tioga Press, 1983.

[Wallace et. al 85] Wallace, R. W. et. al., "First Results in Road Following", IJCAI 85, Los Angeles, August 1985.