ELSEVIER

Artificial Intelligence 100 (1998) 5–85

Artificial Intelligence

Interpreting a dynamic and uncertain world: task-based control

Richard J. Howarth [1]

School of Cognitive and Computing Sciences, University of Sussex, Falmer, Brighton BN1 9QH, United Kingdom

Received 13 May 1995; revised 23 July 1997

Abstract

In this paper we show that it can be beneficial to have a high-level vision component that guides the reasoning of the whole vision system when interpreting a dynamic and uncertain world. This guidance is provided by an attentional mechanism that exploits knowledge of the specific problem being solved. Here we develop a general framework for such an attentional mechanism and its application to understanding dynamic scenes. This attentional mechanism can enable a vision system to perform a given domain task while expending minimal resources. We have developed a component that uses Bayesian networks combined with a deictic representation to select what, when and how to use processed data from a fixed camera. We apply two forms of Bayesian network, which (1) create a dynamic structure to reflect the spatial organisation of the data and (2) measure task relatedness. Together these give attentional focus making the reasoning performed relevant to the task. © 1998 Elsevier Science B.V.

Keywords: High-level computer vision; Surveillance; Attention; Event reasoning; Visual behaviour

1. Introduction

In the modern world there is an increasing use of surveillance, resulting in the need for automatic or semi-automatic methods of processing the dynamic input data. Surveillance concerns more than just observation. In addition to having some intelligence and knowledge, the processing performed is often much more purposive, complying to some known task. There is still along way to go before such systems can be fully automated, although the work described in this paper is a step in this direction. A visual

[1] Email: richardh@cogs.susx.ac.uk or howarth@dcs.qmw.ac.uk.

PII S0004-3702(98)00004-6

surveillance system also provides a less complex problem than a fully interacting vision system since the perceiver just views the scene of interest. We are still left with the problem of understanding the various unfolding dynamic behaviours of the actors/objects in the scene, and also with the many problems associated with using machine-based processing to extract the visual evidence which is subject to noise, occlusion, and the general ill-posed nature of inferring the scene from image data.

Understanding what we perceive is often a surprisingly difficult process and we do not address the full visual understanding problem, instead we make a number of assumptions and consider a restricted case of the surveillance of wide-area dynamic scenes. Our restricted *surveillance problem* has the following simplifications that make visual understanding more tractable: we use a fixed camera that observes the activity of rigid objects in a structured domain. Examples include: a road-traffic scene where the main interest is the road vehicles, and airport holding areas where we are interested in the activities of the various special vehicles that unload and service the passenger aeroplanes.

To perform surveillance we need to reason about the activities of the objects that are perceived. The process by which this visual perception is performed is complex and will not be fully addressed here. However, since perception is an important part of surveillance we need to sketch its relationship to this work. We can separate vision into three stages: low-level (or early), intermediate-level and high-level (or late). Low-level vision is the best understood (Horn [51], Marr [72]), and concerns visual receptors, be they artificial, from a television video camera or biological sensors, with low-level processing that acts on the results from the visual receptors, to provide image features such as edges, corners and flow vectors. Intermediate-level vision is less well understood, and concerns the recognition of objects (e.g., model matching and tracking). Marr [72] describes a framework for 3D interpretation, that begins to address intermediate-level vision. High-level vision is the least well understood and concerns the interpretation of the evolving information that is provided by intermediate-level vision as well as directing what intermediate-level visual processing (or even low-level visual processing) should be performed. In progressing up these levels we see that image oriented information is at the lower levels and the more abstract, symbolic descriptions are at the higher levels. It is the development of high-level visual processing that allows the results from intermediate-level visual processing to be used for reasoning over longer time scales and we use this to obtain a greater understanding of what is going on in the field-of-view.

Until recently it has been rare to find issues of high-level control connected with computer vision (but see, for example, Rimey and Brown [88] and Howarth [55]). The focus tends to be on lower-level techniques that extract information from images rather than on identifying visual behaviours appropriate for visual tasks and how these operate. In this paper we concentrate on the role of high-level vision, with emphasis placed on how what we know about an environment affects the interpretation of the observed object behaviour. And, by using a single fixed camera we ignore difficult issues associated with active cameras and multisensor fusion, while providing ample visual data to address surveillance tasks. Understanding the activity of moving objects starts by tracking objects in an image sequence, but this is just the beginning. The objective of this work is to go further and form conceptual descriptions that capture the

dynamic interactions of objects in a meaningful way. To discuss this we describe some of the advantages obtained by reinterpreting a pipelined, passive vision system under a more active vision, situated approach.[2]

To address the *surveillance problem* we introduce the concept of the "official-observer" which acts to coordinate the tasks from the particular point-of-view of our single fixed camera. In the *surveillance problem* here the official-observer does not participate in the environment and is not a partner in any interaction that takes place in the scene. Although the official-observer visually attends to them, the observed people/objects[3] are not necessarily aware of this. Things that are relevant to the official-observer are likely to differ greatly from those of the parties involved in the observed interaction. From the official-observer's camera input we wish to obtain a description of the activity taking place in the dynamic wide-area scene it perceives and from this obtain an understanding of the dynamic and improvised interactions of the scene objects. There are constraints on the official-observer's interpretation of the objects in the scene: we only see the objects that are in the camera's field-of-view; we do not know each participant's goal (typically something like "go to place X"); and what we see is mostly reactive behaviour (rather than deeply planned).

This paper begins by describing the background and motivation for developing a computer program that uses task-based control in order to address the surveillance problem. In Section 2 the initial project framework and the constraints this framework brought are explained. Then in Section 3 we describe an initial attempt at addressing the *surveillance problem.* This first architecture does not use task-based control, but does serve to illustrate the *surveillance problem.* Section 4 contains a reassessment of this first architecture presenting justification for why task-based control was developed and why it is appropriate for the *surveillance problem.* In Section 5 we present an overview of the architecture that does use task-based control, emphasizing those elements that are used in Section 6, where theoretical details are given. Implementation details are given in Section 7 and in Section 8 examples are presented that illustrate how task-based control operates. The parts of this paper following Section 8 consist of a discussion of related work, conclusions and appendices.

2. VIEWS and HIVIS

In the ESPRIT II project 2152, VIEWS (Visual Inspection and Evaluation of Wide-area Scenes) (for an overview see Corrall et al. [28,29]), the overall flow of information is from images to behavioural or "conceptual" descriptions. This involves a number of steps, which, as shown in Fig. 1, can be coarsely separated into: (1) the Perceptual Component (PC) which performs low- and intermediate-level visual processing, and (2) the Situation Assessment Component (SAC) which performs high-level control and

[2] The ideas behind "active vision" are described at the beginning of Section 5. By using the word "situated" we are recognising the broader issues of "being in the world" some of which are covered in Section 5.3.1.

[3] We use the word "object" to refer to physical entities like cars, trams, and planes, where the person operating the machine may not be visible to the official-observer.

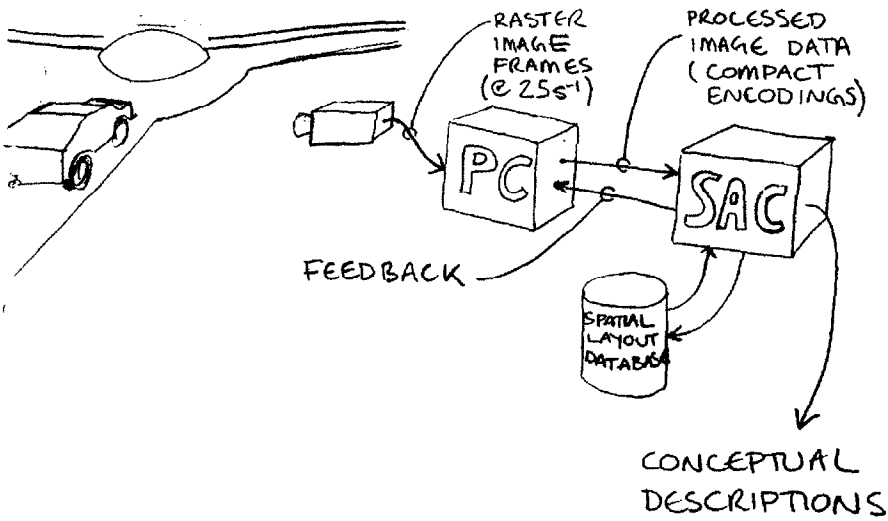


Fig. 1. The perception and situation-assessment components. The PC performs low- and intermediate-level visual processing. The SAC performs high-level processing.

interpretation. On the left of the figure we depict an example scene, a complex junction, in this case a roundabout. What we want to determine is the behaviour of the vehicles that pass through this junction.

This separation of the two components was done to allow simultaneous development of both components but also led to a strong interface between the two. The two HIVIS-systems[4] that we discuss in this paper, HIVIS-MONITOR and HIVIS-WATCHER, are in effect alternative Situation Assessment Components, with HIVIS-MONITOR having a passive architecture based on "event reasoning" which is the identification of behavioural primitives, their selection and composition. In contrast, HIVIS-WATCHER takes a more active-vision approach where the surveillance task of the observer affects the behaviour of the system, let us call this "task-based control".

This interface that falls between intermediate- and high-level processing was initially for the pragmatic purpose of enabling research to be performed on the VIEWS project (see, for example, Howarth and Buxton [56,57] and Buxton and Gong [20]). Drawing the interface between the PC and SAC, as shown in Fig. 1, allows the results from intermediate-level processing to be collected as a stream of "compact encodings"[5] for

[4] HIVIS is an acronym for HIgh-level VISion.

[5] The compact encodings take the form of a sequence of ground-plane outlines of each object, where each outline is called a "posebox". Each posebox represents a result from the model-matcher that determined the "pose position" of the object in the scene for a given image frame. The "shape" of the posebox denotes the extent of the object, and its frame-of-reference in the form of the object's front, back, left and right. More details are given in [54].

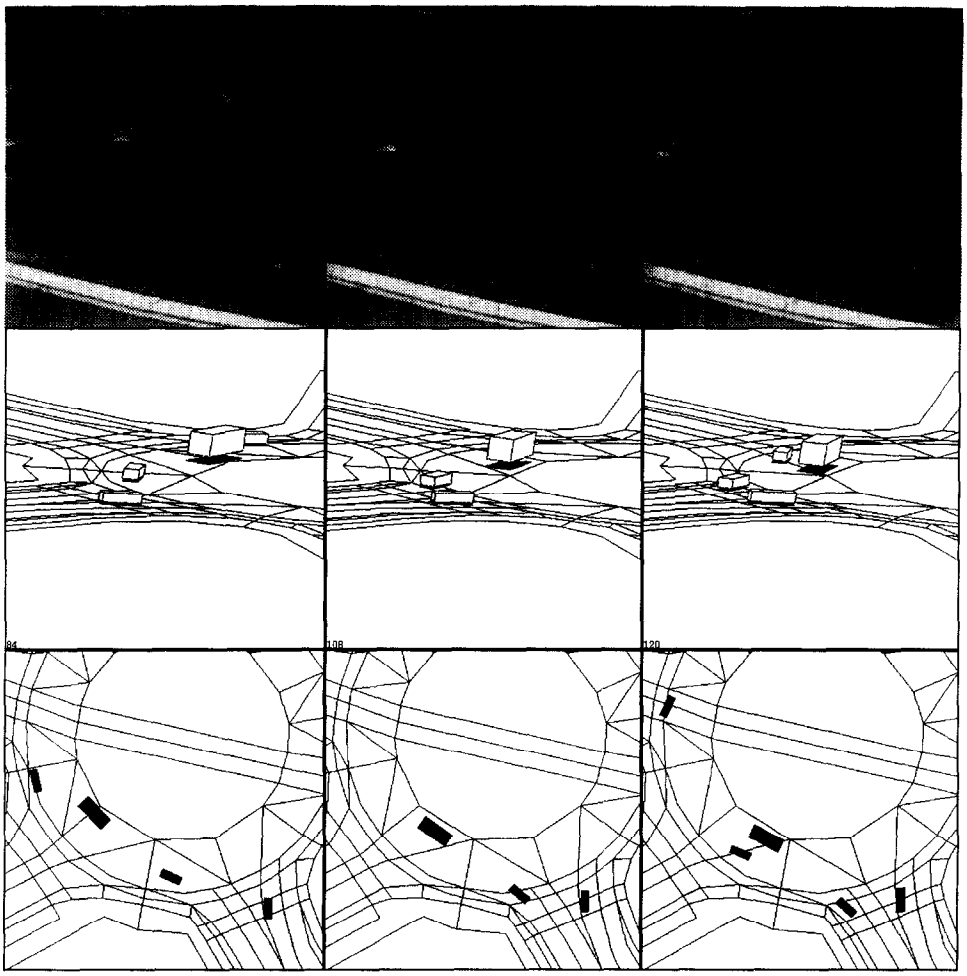


Fig. 2. Three images showing typical vehicle activity on the roundabout and how the 3D pose descriptions can be transformed to a ground-plane view.

subsequent high-level processing. The problems ensuing from this initial decision are highlighted in the research reported here. Unfortunately moving the interface back one stage would require addressing much more material and this more complex problem is the subject of future work.

In the computer program HIVIS-MONITOR event reasoning is used to build temporally evolving episodic descriptions that act as basic elements for conceptual descriptions of all moving objects in a scene. The identification of conceptual descriptions can be pipelined to provide a database of results about the activities of all the moving objects.

In the computer program HIVIS-WATCHER task-based control is used to identify those scene objects likely to be worth attending because they promise to be relevant to

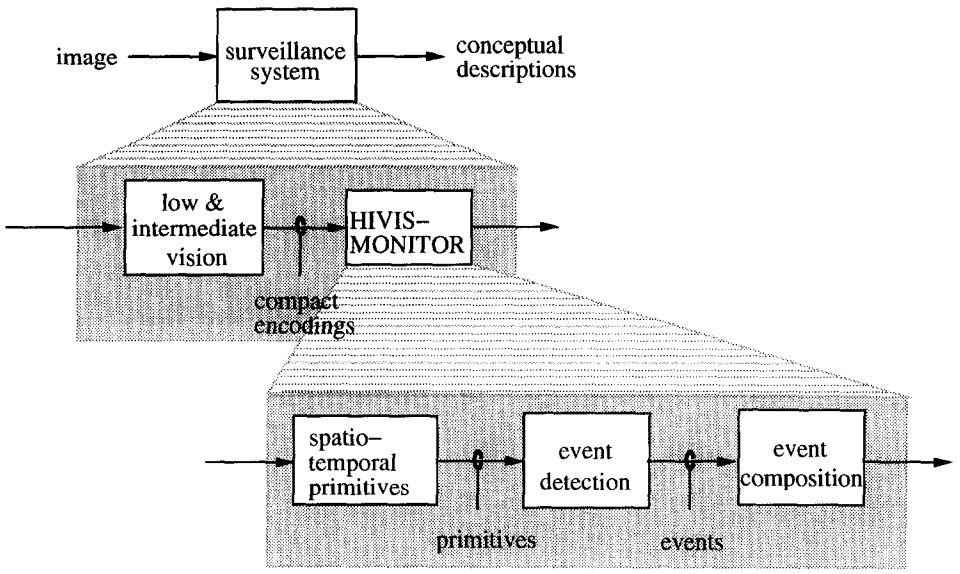


Fig. 3. An outline of HIVIS-MONITOR.

the given surveillance task. In contrast to the monitoring approach, we no longer collect information about all scene objects, instead only data that is potentially task related is processed.

In these HIVIS-based systems we have developed approaches that recognise the importance of the continually changing world. We first briefly describe HIVIS-MONITOR, embodying the initial design, which is mainly to provide background and motivation for the main subject of this paper which is how task-based control operates and how it has been implemented in HIVIS-WATCHER. To demonstrate the different behaviour of the two systems we will use examples drawn from the road-traffic domain. Fig. 2 illustrates this application domain with three image frames selected from a sequence taken at a German roundabout. In this part of the sequence a number of episodic behaviours are unfolding: one vehicle leaves the roundabout; another is in an entry lane to the roundabout; also towards the rear of the image a car begins to overtake a lorry. Below the image frames we provide an illustration of the poseboxes, which are results from a model-matcher (see [33,73,97,110,111] for details), and because we have made the assumption of a known ground-plane, we are able to display the data from an overhead view. Here changing the display viewpoint has not changed the camera position in the real world.

3. HIVIS-MONITOR

In our first system, HIVIS-MONITOR, we adopt a pipelined approach that reflects the general flow of data from images to conceptual descriptions. As shown in Fig. 3

high-level visual processing centres around three components: extracting spatio-temporal primitives from the stream of compact encodings produced by low- and intermediate-level visual processing, detecting events from these primitives, and composing the events to form episodic sequences which are stored in an evolving database. This database is extended as new events continue, begin or end the various episodes under construction. The problem of matching behaviours to a user question is left to the query-based component that interrogates the database.

At first sight, this seems an admirable system design, allowing the parallel, separate development of both the low- and intermediate-level visual processing component and the high-level one. However, as we will see, the passive, data-driven flow of the processing causes problems for high-level vision control.

3.1. Script-based approach

To describe the behaviour of participants in the scene we are using an ontology based upon that presented by Nagel [78] to describe events, which captures the common sense notions of the terms being used. One of the projects that Nagel describes is Neumann's NAOS system [80] which uses an extensive list of motion verbs. The objective in the NAOS system is the formation of off-line natural-language descriptions where as in the HIVIS systems we are generating what could be the input to a natural-language generator.

Our use of events and episodes is similar in some respects to that described by Schank and Abelson [92], who compose events into scripts to describe typical behaviour of a customer at a restaurant such as entering, going to a table, ordering, eating and paying. This provides a hierarchical layering from events to episodes, and then to more complex script-like behaviours. This hierarchical decomposition and relationships between the behavioural elements can be used to define a grammar where events are terminal symbols in the language to be parsed. This approach could use the syntactic methods described by Fu [40], the static semantics of an attributed grammar as described by Frost [39] and Clark [27], the island parsing described by Corrall and Hill [29], or the compositional semantics described by Woods [109]. Howarth and Buxton [56] and Howarth [54] give more details of how the HIVIS-MONITOR computer program is implemented, but basically the approach for maintaining a history of the behaviour that takes place in the scene is to note down the event primitives that have been detected and then use an ongoing interpretation process to see how these events fit together. The input given to the database consists of the events and activities associated with a particular property. In addition to the functions that compute these values there are further functions that update the temporal structure by beginning, extending or ending the continuity of the value/signal for each property. To identify an episode we use a filter that extracts the necessary property values from the available properties.

This has described how HIVIS-MONITOR is data-driven and that it follows the script-based approach, constructing an interpretation of object behaviour in an evolving database that holds entries for the history of each individual object and the interactions between them. This passive *bottom-up* approach reflects the flow of data from image to conceptual descriptions.

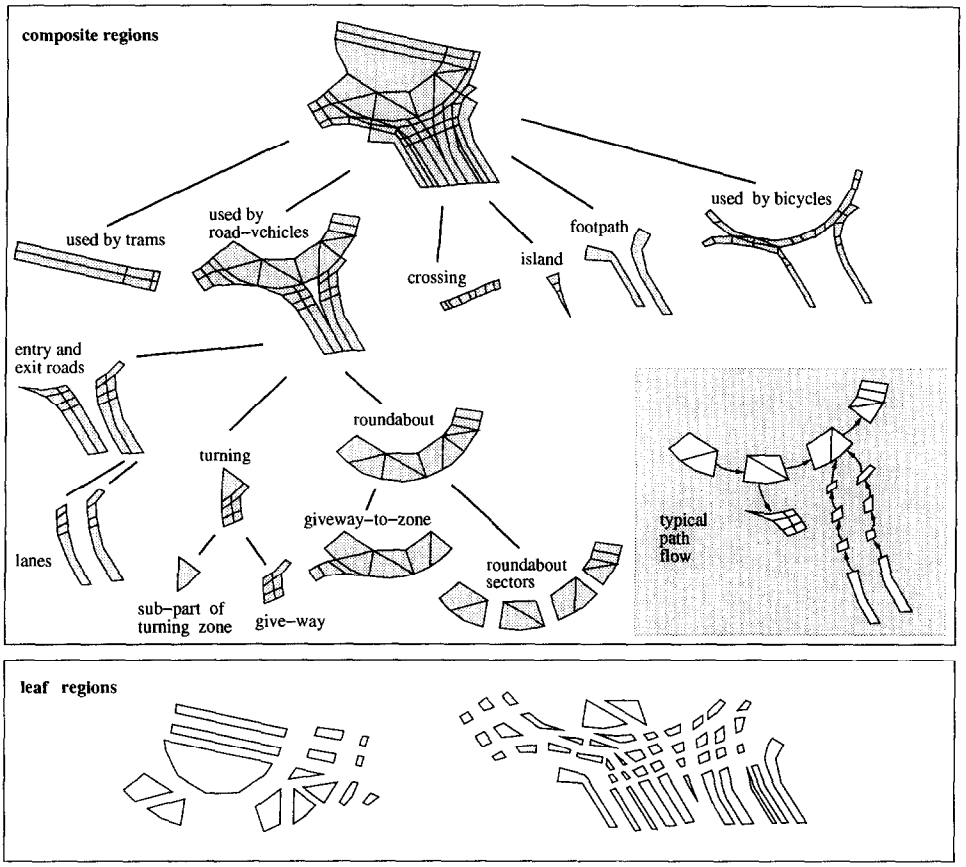


Fig. 4. A spatial decomposition of a fragment from the Bremer–Stern roundabout. This shows the separation between the leaf-regions and composite-regions, and the typical path flow that connects some composite-regions (e.g., the roundabout sectors).

3.2. *Spatio-temporal representation*

The spatial representation used by HIVIS-MONITOR is based on a model of space developed by Fleck [36, 37] for representing digitised spaces and which she has used for both edge detection and stereo matching. In [37] she describes how this representation can be used for qualitative reasoning and for modelling natural language semantics. The spatial and temporal representation Fleck uses and calls "cellular topology" is itself based on a mathematical foundation developed by Whitehead [107] as part of his work on combinatorial topology. Cellular topology uses cells to structure the underlying space (see Fig. 5[6]) which we augment slightly by adding a metric. We can attach

[6] Note that the regular cell pattern illustrated in Fig. 5 is not necessary for a regular cell complex.

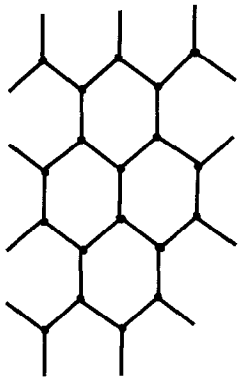


Fig. 5. A regular cell complex.

information about the world to this underlying spatial representation by introducing a level of abstraction to separate implementation issues from the data being represented. For example, if we just consider a 2D spatial representation of a ground plane that contains a number of important edges (e.g., walls) which we use to cut space up into meaningful regions (e.g., rooms) in accordance with the rules that govern what can be represented by cellular topology. These meaningful regions (which may include things like typical paths through a space) may overlap or have subset/superset relationships, giving rise to a hierarchy of regions, the smallest common units of which we call "leaf-regions". These leaf-regions only represent the spatial component, where as the "composite-regions", which are composed from the leaf-region primitives, can have various attributes attached to them. The ground-plane tessellation given in Fig. 2, shows how the leaf-regions cut up the space viewed by the official-observer. This ground-plane depicts the Bremer–Stern roundabout in Germany and contains the various typical paths for trams, road-vehicles, cyclists and pedestrians. Fig. 4 illustrates some of the data held in spatial-layout database used in HIVIS-MONITOR, showing the hierarchy of composite-regions, how type attributes are attached to composite-regions, and how transition information can be used to link composite-regions. For more details and how this database is used for "contextual indexing" see Howarth and Buxton [56] and Howarth [54].

We also use Fleck's cellular topology (see [37]) to model time using a cell complex whose underlying space is the real number line $\mathbb{R}$. This time-line is combined with a four-way classification of situations used in linguistics and originally due to Vendler (van Benthem [105, pp. 133 and 196] calls this "Aristotle–Kenny–Vendler" verb classification). Fig. 6 illustrates the situations showing how they are divided into: "states", like own a house, be a student; "state-changes"[7], like get to work, find a book; "activities", like eating, driving a car; "accomplishments", like travel to work, after an hour's debugging the program compiled. To which we can add another situation called

[7] Also called "achievements".

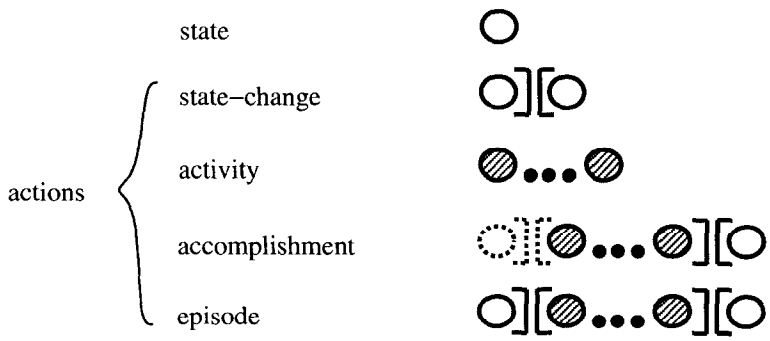


Fig. 6. State and action models of situations using cellular topology. The shaded cells can be referred to using a progressive form. Adapted from Fleck [37, p.21]. Copyright 1996 by Elsevier Science. Adapted by permission.

"episode" to denote accomplishments that have a definite beginning and ending. The term "actions" is used as a cover term for activities, accomplishments, state-changes and episodes. Similar classifications have been used by Ryle [91], Taylor [98], and Allen [3]. From these we identify the key parts of our event ontology which can be defined as follows:

Definition 1. An *event* is a significant change to a property's value.

This definition is similar to that described by Nagel [78] and conforms to the point-of-action definition given by Newtson [81].

Definition 2. An *activity* is a continuous sequence of states that have the same property.

This definition of activity tries to capture the idea of an evolving activity being about one thing, and that it is steady and consistent. These definitions are different to those used by McDermott [74] and Shoham [94] where events are propositions that initiate change in the world. Here an event is the change and the proposition could be, if appropriate, described as an activity or a state.

Definition 3. An *episode* expresses a sequence of related activities bounded by two events.

The objective of HIVIS-MONITOR is to identify when an episode is taking place.

This has now covered the underlying static spatial representation and a model for representing an evolving temporal history, in terms of actions and states. This model of situations is useful for representing discrete observations provided by camera input, allowing us to "fill in the gaps" as illustrated in Fig. 7 and outlined in more detail in [54]. Fleck [37] places this in a linguistic context, which we will not discuss in this paper, however, it does provide one route for the integration of a natural language component into the HIVIS-systems.

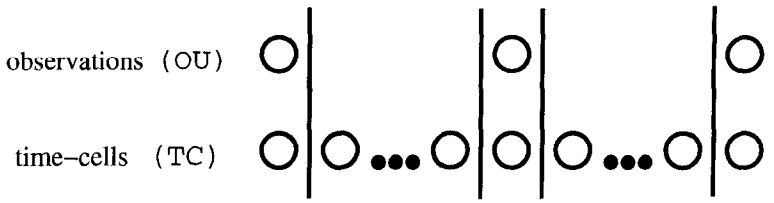


Fig. 7. The time-cells provide a discrete model of the $\mathbb{R}$ time-line, and we receive observations every n time-cells, where n may be fixed and $n \geqslant 1$, $n \in \mathbb{N}$.

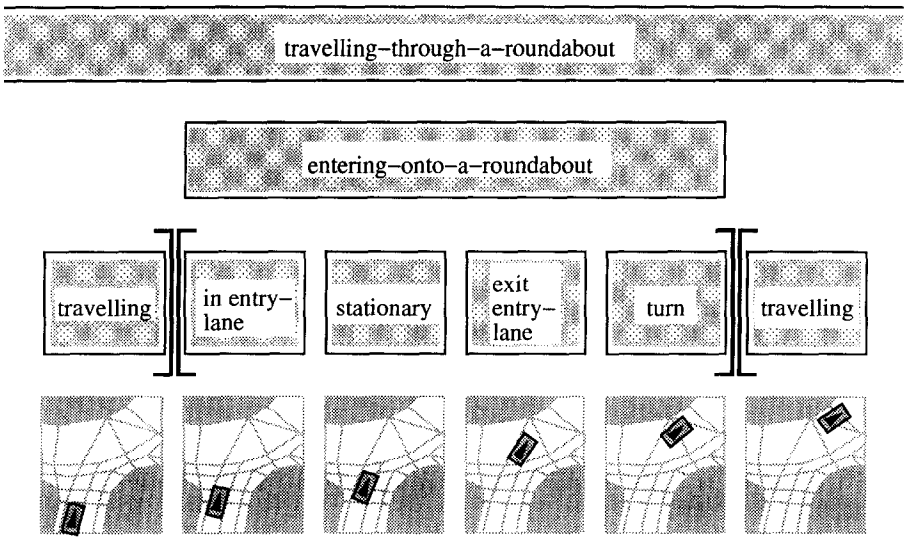


Fig. 8. Entering onto a roundabout.

The stream of posebox data supplied by the model-matcher describes the space swept out in time by each object's path to form what we call a "conduit". The conduit is a simplified form of the extrusion described by Cameron [21]. It also corresponds to a simplified form of Hayes' [48] "histories", covering a much more restricted set of physical properties. The conduit is used to provide an approximation of the time at which a region is exited or entered. To do this, we extrapolate the space-time description between updates. The path formed will depend upon the curve or line used to connect the points and whether knowledge about the regions is used (for example, in the road-traffic domain cars usually stay on the road). This method of *small scale* path completion can also be applied to reasoning about missing updates due to error or occlusion. Conduits capture in spatio-temporal form the past history of an object's passage through the scene, providing an unusual representation of continuity of travel, providing a useful framework for illustrating region transition and occupancy. Fig. 8

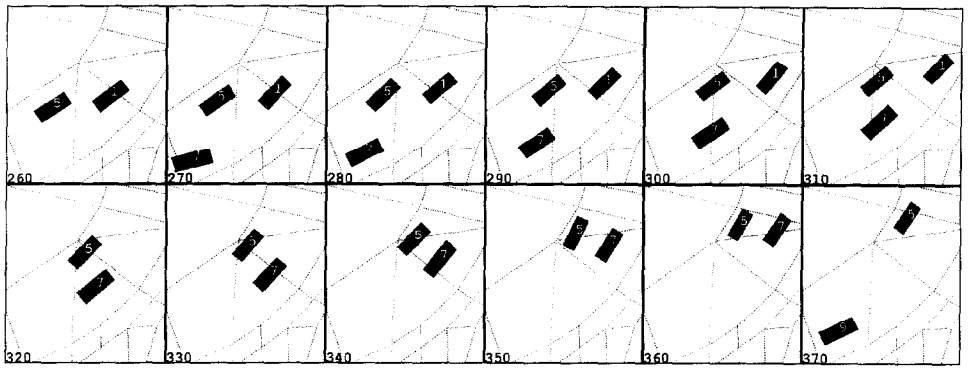


Fig. 9. The frames show four vehicles traversing part of a roundabout.

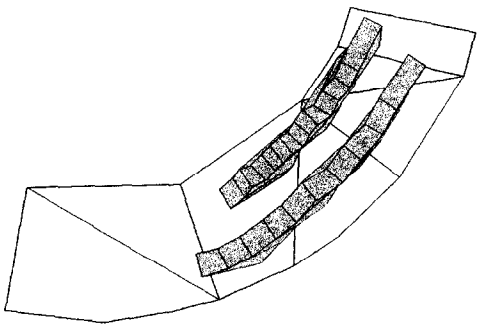


Fig. 10. The conduits from above of vehicles 5 and 7. The other objects have been removed in this example to try and clarify the picture.

provides an illustration of the various elements involved in identifying a simple behaviour like `entering-the-roundabout` by identifying the pattern produced by the activity of the vehicle. This is described in greater detail in [56].

Other behaviours can be similarly identified, such as `following` which is described by Toal and Buxton [100] who use a slightly different spatial representation to that described here, and define `following` as an "overlap in spatial history within some time delay". These analogical approaches do not easily extend to representing behaviours like `overtaking`. For example, consider the sequence shown in Fig. 9 where we have four vehicles traversing the roundabout. Fig. 10 shows two conduits of temporally stacked poseboxes for objects 5 and 7, with the temporal axis displayed more clearly in Fig. 11 where we see the 2D+t conduits from the side.

Once we have generated the conduits, we have the problem of interpreting what they mean. If they intersect then there is a likely collision or near miss, but intersections of conduits is unusual. Other tests can be made possible by removing a pertinent dimension and testing to see if the components of the reduced model overlap, in the test

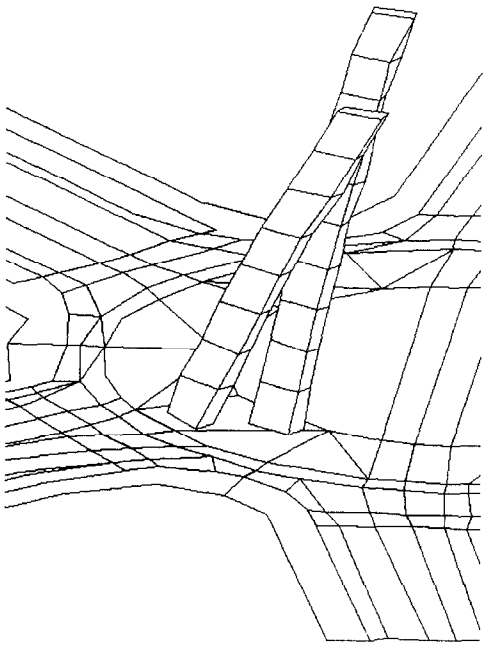


Fig. 11. The conduits from the side.

for `following` behaviour we tested for an overlap with some time delay. Overtaking can be identified by ignoring the spatial dimension parallel to the objects direction of motion however, this spatial dimension should really be the 2D manifold, called the "rectifying developable" (as described by Koenderink [66], for example), that fits the space curve of each object's path. Mapping the conduits into one of these manifolds to perform such as test is difficult, although in principle it should be possible.

3.3. General features

We claim that HIVIS-MONITOR demonstrates typical traits of the class of traditional AI approaches we have called "script-based". In general, all script-based systems will have the following features:

- Maximal detail is derived from the input data. This approach obtains a description of all objects and all interactions, over the whole scene, for all the episodes it has been designed to detect.
- Representation is extracted first and the results are placed in an evolving database that is used to construct more abstract descriptions using hindsight.
- Single object reasoning is performed with ease using this approach.
- Simple implementation can be achieved using standard AI techniques.

It is quite likely that better implementations can be developed that fulfill the script-based approach (see, for example, the description of plan recognition by Allen et al. [4], and the results[8] from the VIEWS project given by Corrall et al. [28,29], King et al. [64] and Toal and Buxton [100]). However, the exemplar developed here should be enough to judge whether to remain with this approach or to investigate an alternative.

3.4. Limitations

HIVIS-MONITOR has the following limitations:

- It is passive in its processing, operating a simple control policy, that is, not affected by changes in the perceived data.
- It is not real-time because the construction of the results database is an off-line process, and does not send feedback to any form of intermediate-level visual processing. This means that there is a problem getting timely recognition of perceived object activity.
- Unbounded storage is required because any pieces of data contained in the results database might be needed later either to compose some more abstract description or to be accessed by the user to answer a query. Since we do not retract what we have seen or the episodes that we have identified, the database structure is monotonically increasing in size.
- Multiple object reasoning is difficult within the global coordinate system used to express pose positions.
- The computation performed by HIVIS-MONITOR is mainly dependent upon the number of objects in the input data, i.e., it is *data-dependent.*
- This model is inflexible because it only deals with known episodes. Within the constraints of the predicates provided (language primitives that describe events and activities), new behavioural models can be added. However, defining new predicates may be difficult.
- The addition of new operators increases the number of tests performed on all the objects in the scene. For a single object operator there is an $O(n)$ increase, for most binary object operators there is an $O(n^2)$ increase, and for most multiple object operators the increase is polynomial with a maximum of $O(n^n)$, where n is the number of objects in the scene.
- The behavioural decomposition does not take into consideration the temporal context in which the events have occurred, which contributes to the process of interpretation. It is possible that the selection of the "correct" episode description is not possible due to only seeing part of an episode.

3.5. Discussion

From these features and limitations we can identify the following key problems: computation is performed to obtain results that may never be required; and as the

[8] Initial achievements of the project are illustrated by the video [19], and Geake [42] provides a brief report on the project.

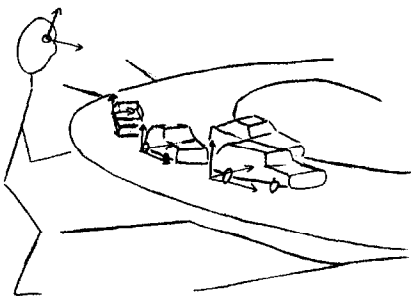


Fig. 12. The different coordinate systems of frame values used by the official-observer and the vehicles it is watching.

database of results increases in size, the performance of the system will degrade. It might be possible to address these by extending the script-based approach however, we will not take this evolutionary route. Instead we will investigate a more situated approach. This new approach differs greatly from the passive, data-driven script-based approach and requires a complete reformulation of the problem to obtain an active, task-driven, situated solution.

4. Reassessment

To begin this reformulation we first consider the use of more local forms of reasoning in terms of the frame-of-reference of the perceived objects, the spatial arrangements of these objects and the use of contextual indexing from knowledge about the environment.

In HIVIS-MONITOR a global extrinsic coordinate system was assumed. By taking a global view we comply with a commonly held Western view of how to represent space in a map-like way as opposed to the egocentric approach described by Hutchins [61] as being used by the Micronesians to perform navigation. The absolute coordinate system also fits well with the concept of the optic-array (see Gibson [43] and Kosslyn et al. [67] for details), if we can consider placing a grid over the ground-plane to be analogical to the optic-array of the perceiver. This representation would allow reasoning to be performed that does not need full object recognition with spatial relationships represented in terms of the optic-array's absolute coordinates (in some respects this is like the video-game world used by Agre and Chapman [2] where the "winner-takes-all" recognition mechanism (see Chapman [25], Koch and Ullman [65], Tsotsos [102] and Tsotsos et al. [103]) allows objects and their positions to be identified by key properties, such as, colour and roundedness).

In contrast to this global viewpoint, when reasoning about the behaviour of each scene object it would be useful if the representation of the properties related to each object could be described in its own relative coordinate system. However, this involves recognising each object to the extent that an intrinsic-front can be identified together with its

spatial extent. This requirement places the need for a more sophisticated understanding of how the image data present in the optic-array relates to how objects exist in the environment. In our *surveillance problem* we can obtain the pose positions of the scene objects via model-matching making local reasoning attractive, although its extra cost in terms of the complexity of intermediate-level vision should be noted. Fig. 12 illustrates how we can just use the official-observers perspective or once we have the pose position of an attended object we can reference other scene objects in relation to this attended object's pose position. Pertinent background details are given by Herskovits [50] and summarised in [55]. We can define these different frames-of-reference as follows:

Definition 4. The *local-form* is representation and reasoning that uses the intrinsic frame-of-reference of a perceived object (exocentric with respect to the observer).

Definition 5. The *global-form* is representation and reasoning that uses the perceiver's frame-of-reference, which operates over the whole field-of-view (egocentric with respect to the observer).

The global-form is not a public world since it, like the local-form, only exists to the perceiver. We are not dealing with representing a shared world in terms of each participant.

The local-form and global-form are not just different frames-of-reference they also include the different reasoning present when using each of these frames-of-reference. The global-form is our familiar, everyday frame-of-reference, such that it provides one overall integrated model of the various spatial objects instead of the different individual frames-of-reference used by each local-form. The local-form, using the frame-of-reference of an attended object to provide a relative spatial framework, seems to require a little more cognitive effort in its use. Both Bühler [18] and Herskovits [50] describe these frame-of-reference as being incompatible. For example, Herskovits describes how a human being learns to construct a frame-of-reference starting from two basic experiences: (1) the experience of looking straight ahead with his or her body standing upright on horizontal ground (we will call this the "canonical position"), and (2) the experience of encountering another human being face-to-face (the "canonical encounter"). These are both illustrated in Fig. 13.

In part (b) of Fig. 13 the perceiver "combines" his or her own point of view with that of the person encountered (i.e., the person called Bob in the figure). As the figure illustrates Bob's front and back axes point in directions opposite to those of the onlooker. However, the right and left axes can be interpreted in two different ways with respect to the observer's right and left axes: either the same, which is called "mirror order", or the opposite, called "basic order". Note that basic order is Bob's own frame-of-reference. We can resolve these two different interpretations by referring to Fig. 12 and identifying that mirror order is using the global-form (Definition 5) and that basic order is using the local-form (Definition 4). By using the frame-of-reference of a perceived object, we can then analyse all the changes involved or transpositions through space of the object in terms of our own field values and experience. This representation seems *natural* because we use it in our everyday interactions with the world.

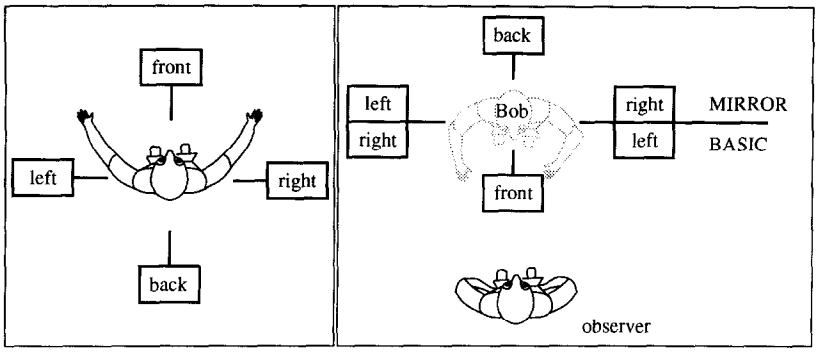


Fig. 13. Frames of reference: (a) an observer in "canonical position", (b) the "canonical encounter". Adapted from Herskovits [50, pp. 158–159]. Copyright 1986 by Cambridge University Press. Adapted by permission.

The suitability of each HIVIS-system is detailed in Table 1, providing an indication of the extent of the reformulation to the *surveillance problem*. HIVIS-MONITOR would be useful for off-line query of behaviour, whereas in HIVIS-WATCHER, by asking the question first, we remove the importance of the results database because we are no longer providing a query-based system. This removes the need to remember everything and solves the problem of the monotonically increasing database because in HIVIS-MONITOR it is difficult to know when something can be forgotten. The development of a more situated approach in HIVIS-WATCHER is part of the adoption of a more local viewpoint that uses a "deictic"[9] representation of space and time. In some applications, using HIVIS-MONITOR and processing all scene objects might be necessary however, in cases where it is not, the HIVIS-MONITOR approach is ungainly. In the context of the *surveillance problem* we are inherently concerned with: the "here-and-now", the evolving contexts of both observer and scene objects, and the formation of a consistent, task relevant interpretation of this observed behaviour. By taking a deictic approach in HIVIS-WATCHER we do not name and describe every object, and we register only information about objects relevant to task.

5. HIVIS-WATCHER overview

In HIVIS-WATCHER we remove the reliance on the pipelined flow of data and instead use feedback to control the behaviour of the system. This use of feedback is common in control theory, but not so common in AI (Nilsson [83] discusses this point). Deictic representation plays an important role in this framework because it supports attentional

[9] Deixis is used in several disciplines such as linguistics (Buhler [18]), the social sciences (Garfinkel [41], Heritage [49]) and spatial representation (Herskovits [50, pp. 156–192]). Deixis is the use or referent of a deictic word (e.g., I, now, this, that, here) and is an aspect of a communication whose interpretation depends on knowledge of the context in which it occurs.

Table 1
This table summarises the comparison between the two HIVIS-based systems, with the illuminates column describing what each row is about.

HIVIS-MONITOR	HIVIS-WATCHER	illuminates
passive	active	control
open-loop/pipelined	closed-loop/feedback	architecture
off-line	on-line	immediacy
structured	purposive	approaches
global	local and global	viewpoint
maximal detail	sufficient detail	investigation
unlimited resources	limited resources	complexity
extract representation first	ask question first	timeliness
answer question from representation data	answer question from scene data	memory cost
data-dependent	task-dependent	propagation

processing with emphasis placed on the behaviour of the perceiver as it interprets the activity of the scene objects rather than just representing the behaviour of the scene objects on their own. This active/animate vision approach that takes account of more active, task-driven requirements is described by Ballard [6], and builds on Ullman's [104] proposed integration of multiple visual routines. Ballard cites two key reasons for this animate vision approach: first, vision is better understood in the context of the visual behaviours engaging the system without requiring detailed internal representation of the scene; and second, it is important to have a system framework that integrates visual processing within the task context. What is new in HIVIS-WATCHER is the adaptation of active vision methods to allow selective processing in advanced visual surveillance.

In Sections 3 and 4 we introduced some of the concepts used in HIVIS-WATCHER such as the notation for events and actions used in HIVIS-MONITOR and the local- and global-forms. As shown in Fig. 14, HIVIS-WATCHER has three separate elements: the "virtual world" which holds data about the world, the "peripheral-system" which can access this data about the world, the "central-system" which controls system behaviour. The peripheral-/central-system architecture used here is similar to that advocated by Agre [1] and Chapman [25] fulfilling both of Ballard's key reasons given above. The peripheral-system is in effect the plant, it contains a set of visual operators. The central-system is the controller that selects which visual operators to run, based on the feedback of the results from the visual operators just run. Because of this the peripheral- and central-systems form a tightly coupled architecture where feedback is an important feature.

Here we briefly describe the three elements of HIVIS-WATCHER, that contain the machinery needed to support the examples given in Section 8, before providing a more detailed description of task-based control.

5.1. Virtual world

The virtual world acts as the interface with the real world, containing a "buffer" and a priori knowledge about the environment being observed which includes the spatial-

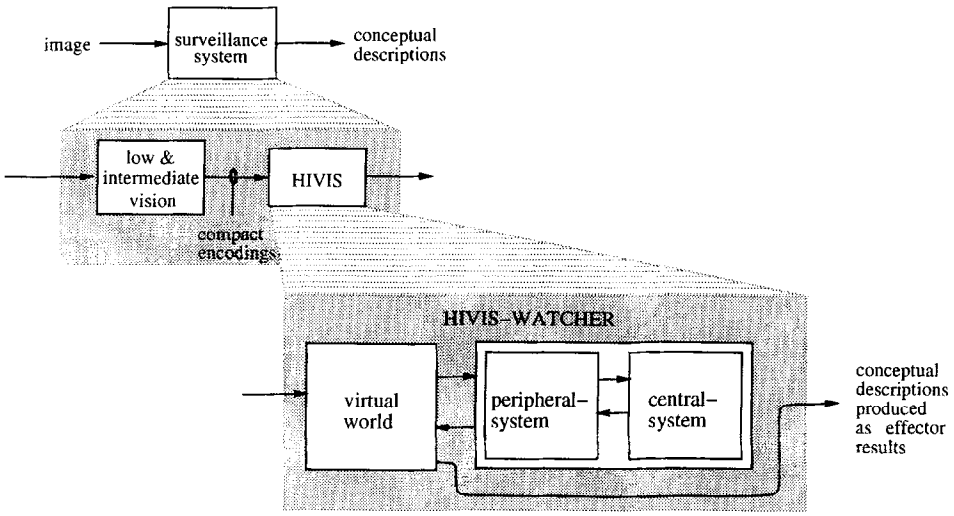


Fig. 14. An outline of HIVIS-WATCHER.

layout database described in Section 3.2. The data held in the virtual world is accessed by the rest of HIVIS-WATCHER via visual routines. The virtual world is not an internal "mental" representation of the world, instead it is a representation of both static a priori knowledge and dynamic data from intermediate-level vision processing. The virtual world represents the signal from a vision component whose global "overhead" viewpoint has been obtained from a fixed camera position (as described in Section 2). The virtual world representation does no runtime reasoning, it is used solely to hold information for access by the peripheral-system. If HIVIS-WATCHER had direct links to the vision system then the visual operators would directly access the available vision system results. Instead we are using a time frame stamped compact encoding from the visual processing as described in Section 2.

The buffer holds the dynamic perceptual data from the intermediate-vision component that arrives as a frame of compact encodings (i.e., 3D pose positions). The dynamic data provides knowledge about the scene objects and is replaced on each system clock tick when a new frame of compact encodings is given. The model-matcher gives each object a unique name and when a new object is identified it is given a "buffer-address" in the virtual world. The buffer-address is used by preattentive processing, and by some attentional markers (which are described in Section 5.2). The model-matcher updates for each object are put in the respective buffer-addresses when this information about each new frame arrives. The virtual world does not maintain any history of past updates, only the ones for the current frame. When a buffer-address is not given an update (i.e., the object it represents has let the scene) it is reused and will be given to a new object when one is identified.

Although all the image data (in the form of conceptual encodings) is present in the virtual world. The rest of HIVIS-WATCHER can only obtain certain preattentive

properties of all objects and attend (i.e., obtain all the results from the model-matcher) when a marker is attached to a buffer-address.

5.2. Peripheral-system

The peripheral-system is based upon Ullman's argument [104] for a "visual routine processor" which integrates multiple visual operators that perform particular sorts of perceptual work such as tracking, representing shape properties and spatial relations. This part of HIVIS-WATCHER follows the approach described by Agre [1] and Chapman [25]. Horswill [52] describes a real-time implementation of such a visual routine processor. Both HIVIS-systems employ event detection operators however, the key difference is that in the HIVIS-WATCHER peripheral-system operators are not run all the time, they are only run when selected by the task-based control system.

5.2.1. Attention and Gestalt primitives

The problem of selecting important figures is reviewed by LaBerge [68], and described by both Rock [89] and Treisman [101] who propose attributing visual processing to two stages called "preattentive" (or peripheral) and "attentive" (or foveal). Pylyshyn and Storm [86] also use this separation in their "FINST" theory by separating multitarget visual tracking into two stages, one a parallel preattentive indexing stage and the other a serial checking stage invoked in selecting a response. At the first stage simple features are preattentively registered, by what we will call "simple operators", which describe global features but not fine detail. Treisman provides examples of texture segregation (a prerequisite for figure ground separation). Murray et al. [77] discuss other cue like processes such as one to detect "looming motion".

At the second stage objects are identified using the candidates set up by the preattentive stage. Mahoney and Ullman [71] describe how the results from demonstrations of the preattentive stage support their use for directly indexing local features as long as the figure of interest is distinguished from irrelevant figures by a single one of these features. In this way preattentive processing can propose figures for use by attentive processing. This ties in with the description of deictic representation given in Section 5.2.2. For example, once the first stage has identified a contiguous blob of space we can mark this location and bring specialised processing to bear on the target. This may involve things like working out what it is, by first attending to the whole object then adjusting downward to align with parts of the object.

In the first stage some of the object interrelationships that we wish to describe are similar to the early discoveries made in Gestalt psychology (see, for example, Gordon [46, pp. 46–75]) concerning grouping properties, such as: proximity, similarity, good configuration, common fate (spatial and/or temporal continuity), closure, and symmetry. The Gestalt primitives act as the basis for the following simple operators used in HIVIS-WATCHER:

- *Proximity*. One of the most elementary spatial primitives in spatial reasoning concerns the relative nearness of one object to another. We use proximity as a preattentive cue because objects generally interact with those that are nearby. The implementation of how the nearest other object is selected, is based upon an approach

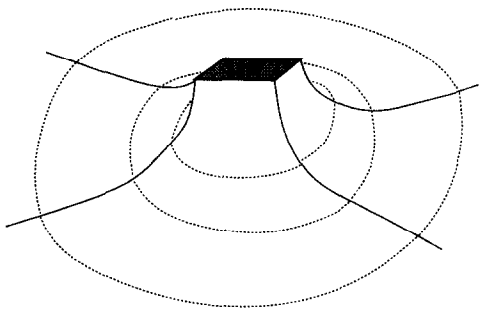


Fig. 15. The nearer to the object the higher the value of "nearness".

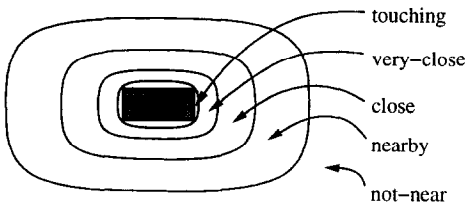


Fig. 16. The dissipation of nearness.

used in robot motion planning (see Latombe [69]). Figs. 15 and 16 show the repulsive field that conceptually surrounds each object. The height of the field reflects the nearness of the proximity relationship. The higher the value the nearer the other object. The nearness measure consists of five qualitative values: `not-near`, `nearby`, `close`, `very-close`, `touching`. While Fig. 15 presents an ideal model rather like the concentric gradient described by Schöne [93, p. 31] for modelling how the intensity of a stimulus (in his work on the spatial control of behaviour Schöne uses the example of a scent gland) decreases gradually as the distance from its source increases. Fig. 16 shows a compromise so that the input can be given to a discrete variable Bayesian network in a qualitative form. Although Pearl [85, pp. 344–357] does describe how continuous variables can be accommodated the more standard discrete model was selected for its relatively greater simplicity.

- *Discontinuity.* In contrast to the Gestalt primitive "common fate" the preattentive cue of discontinuity identifies the distinguishing property of a change in spatio-temporal continuity. This primitive is not part of Gestalt psychology because instead of identifying a group we are identifying when a figure changes between groups. In the scene, an object may become distinguishable for a short time when it changes one of its properties and so changing membership from one group to another (for example, from "figure" to "ground"). In the scene under surveillance the moving objects represent a group and when one object stops moving to become part of the static background its spatio-temporal discontinuity briefly distinguishes it from the other moving objects.

These example Gestalt primitives provide the preattentive cues used by HIVIS-WATCHER in this paper. Neither of these Gestalt primitives needs the detailed positional information provided by the posebox result, a less accurate estimate would probably work just as well.[10] In HIVIS-WATCHER each preattentive cue is used to "pop out" an interesting buffer-address to which a marker can then be attached to investigate the buffers future contents. Once a marker is attached, HIVIS-WATCHER's central-system can then use attentional operators to find out properties about this object (i.e., shape, speed, and so on) and its relationship to its local environment (e.g., the nearest object to it if there is such a proximate object) as described next.

5.2.2. Deictic representation

An advantage of using a deictic representation is that it allows a propositional theory to be developed that is proportional to the number of properties of interest, as opposed to the number of propositional objects in the world. This means that when performing surveillance we do not need to provide a unique name for every propositional object that will ever pass through our field of view. Instead we can define a fixed, smaller number of properties of interest that the official-observer uses to describe the activities of the scene objects. To do this we use the following terms from the work of Agre and Chapman [1,2,25]:

Definition 6. An *entity* is something that is in a particular relationship to the agent.

Definition 7. An *aspect* describes a property of an entity in terms of the agent's purpose.

An aspect provides information about the current state or activity of an object, not its events. For example, *the-car-I-am-driving* is the name of an entity and *the-car-I-am-driving-has-enough-fuel* is the name of an aspect of it. Each time we obtain an aspect (say by looking at the fuel gauge of *the-car-I-am-driving*) its value may be different. We still have the problem of identifying events from the temporal sequence of aspect values (such as the event when *the-car-I-am-driving-has-enough-fuel* becomes false) but this is made more complex because the entity is not always the same (for example, I might not use the same car all the time and each of these different cars is likely to have different levels of fuel). However there is local temporal continuity while using a particular car.

Temporal continuity can be maintained by using a marker to point at the scene object (i.e., the entity). Markers are described by Ullman [104] and Agre and Chapman. A similar form of indexical reference, called FINST, is described by Pylyshyn and

[10] We could calculate these preattentive cues without using the results from the model-matcher. However, HIVIS-WATCHER does not have access the the original image sequence only the stream of compact encodings. Access to the source images would provide the option of using a simple approach to obtain object positions in the image plane (for example, by background subtraction to provide an object's silhouette [7, 10, 14] or by clustering flow vectors [45]). Then modified forms of the Gestalt primitives could be performed by mapping the nearness model to the image plane and by tracking objects (using, for example, the sequential probability ratio test [13, p. 152]). Perhaps mapping the position results into the ground-plane is an option since we are using a structured road-traffic environment which might make performing the mutual-proximity test easier.

Table 2
Simplifications using *the-refobj* and *the-other*.

	long form	abbreviation
entity	*the-reference-object-I-have-selected*	*the-refobj*
aspect	*the-reference-object-I-have-selected-is-moving*	*the-refobj-is-moving*
entity	*the-secondary-object-to-the-refobj*	*the-other*
aspect	*the-secondary-object-to-the-refobj-is-moving*	*the-other-is-moving*

Storm [86], who present psychophysical evidence that there is a numerical limit of four or five on the number of objects that can be tracked at one time.[11] The ability to track four or five objects is pertinent to the *surveillance problem* as it can be used to place an upper bound on system complexity, since a human observer can perform surveillance with such limitations. There is the issue of whether humans, our benchmark here, are able to perform all surveillance tasks sufficiently well in traffic scenes, thus this upper limit on system complexity (i.e., no more than four or five objects) may be inappropriate for tasks of greater complexity than we consider in this paper.

When reasoning about an object in the scene it is useful to use the object's local-form (see Definition 4) and obtain aspects about the other objects in relation and relative to the attended object. To do this we extend the deictic approach by identifying a primary reference object that we will call *the-refobj* and which is short for *the-reference-object-I-have-selected* where *I* refers to the official-observer. We also use the abbreviation *the-other* as described in Table 2 to refer to these secondary objects of interest. In relation to these deictic references the following primitives are used in HIVIS-WATCHER:

- *Positional.* The positional locations formed by the field values of an object are used to cut space into qualitative regions.[12] This qualitative coordinate system is used to obtain values for the aspects like *the-other-is-on-the-right*, etc., as illustrated in Fig. 17. These aspects can be composed so that we get things like *the-other-is-behind-and-on-the-right* to describe all eight relative qualitative positions. In Fig. 17 each dark oval represents the position of *the-other* object. An oval is used to show that we can reason about where *the-other* object is without needing to know what it is.[13] Fig. 18 demonstrates that there need not be a clear inverse relationship between two objects. For example, if we use notation like (*P position Q*) to describe *P*'s position via *Q*'s frame-of-reference, i.e., *Q* is *the-refobj* and *P* is *the-other*, then we can say that `(X infront Y)` is not necessarily the same as `(Y behind X)`, because, in Fig. 18, `(c infront b)` is `true`, and `(b behind c)` is `false`. Also in Fig. 18 we do not show the half spaces extending to infinity in order to make the illustration clearer, and because this positional test is typically only used to find the relative qualitative position of the nearest object, extending the lines to infinity is not strictly necessary.

[11] Miller [75] describes similar limits on the capacity of processing information from other stimuli.
[12] In HIVIS-WATCHER we have implemented the "basic order" frame-of-reference introduced in Fig. 13.
[13] As Koenderink says [66, p. 60] "On a sufficiently low level of resolution *any* shape appears as an ovoid".

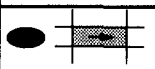
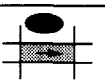
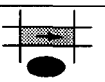
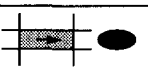
POSITIONAL ASPECTS		ILLUSTRATION
the-other-is-behind		
the-other-is-beside	*the-other-is-on-the-left*	
	the-other-is-on-the-right	
the-other-is-infront		

Fig. 17. The positional aspects of *the-refobj*.

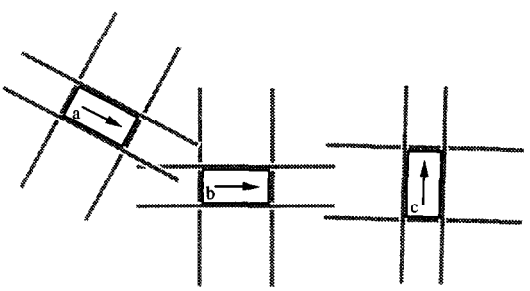


Fig. 18. The half spaces of three cars.

- *Heading difference.* The relative values for the heading of the other objects in relation to *the-refobj* are used to provide the aspect values for *the-other-is-parallel*, *the-other-is-at-right-angles*, or *the-other-is-head-on*. We use either the heading property of the two objects, or the orientation property if the heading is not available (e.g., because one or both the objects is stationary or has just entered the scene). Heading is preferred since it uses the direction of motion and can take account of unusual situations such as reversing or travelling backwards.
- *Speed difference.* To determine the relative motion of *the-other* object in relation to *the-refobj* we use the values for object speed. The results provide values for the aspects *the-other-has-same-speed*, *the-other-is-faster*, and *the-other-is-slower*.

These are the relative qualitative relationships used in HIVIS-WATCHER (for position, direction of motion, and speed) from the reference object to an identified nearest other object, where the nearest other object has been identified by the preattentive cue for proximity (see Section 5.2.1). The problem with this approach is that if we want to obtain all binary relationships between each pair of objects that *the-refobj* and *the-other* can point to in the scene, then we have an $O(n^2)$ number of tests to perform for each

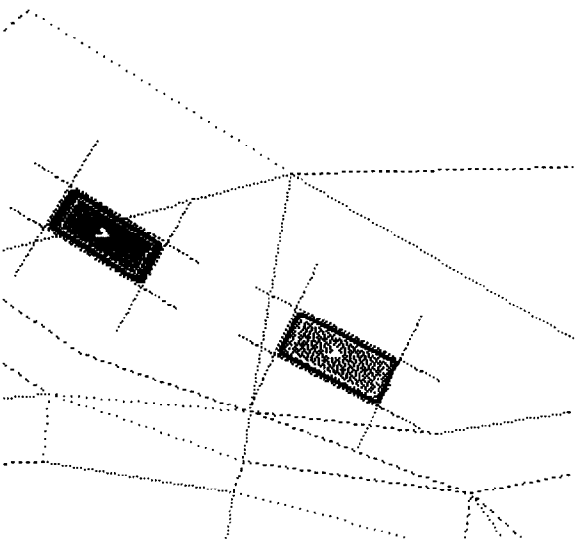


Fig. 19. The local-form view of object properties via *the-refobj*. This facilitates the deictic representation described in Section 5.2.2.

particular relationship. This problem is partially addressed by using the typical-object-model, which we discuss in Section 5.3, however, we still need to select what to attend, which is the subject of Section 6. Next we look at the framework within which these attentional references are made.

5.2.3. Perceived environment

To coordinate perceiver references in the scene we use the frame-of-reference provided by the global-form. This global viewpoint of the scene conforms to the intuitive idea of using general and abstract properties. Great spatial resolution is not needed as the global-form does not involve representing the physical shape of scene objects and ground-plane regions. More detailed object oriented properties (e.g., the local-form) can be obtained by attending the object concerned. In the global-form we only require a coarse representation of position that denotes the location of the fovea/marker (in HIVIS-WATCHER these markers are identified pictorially by shapes like ⬟, ◆ and ✚, following the approach used by Agre and Chapman) and which can be used to access the local-form information should this be required. This global coordinate system uses $\mathbb{Z}^2$ providing a "grid" over the ground-plane that has a qualitative description of object position as discussed in Section 4. A comparison between local and global representation is given in Figs. 19 and 20 with Fig. 21 showing that this grid is part of the peripheral-system and not part of the real or virtual world.

The grid is part of the implementation for describing marker positions and also the size of all the markers, i.e., each marker is of size "grid square". To some extent the granularity of this grid is not important, which perhaps sounds strange. As Fig. 20 shows

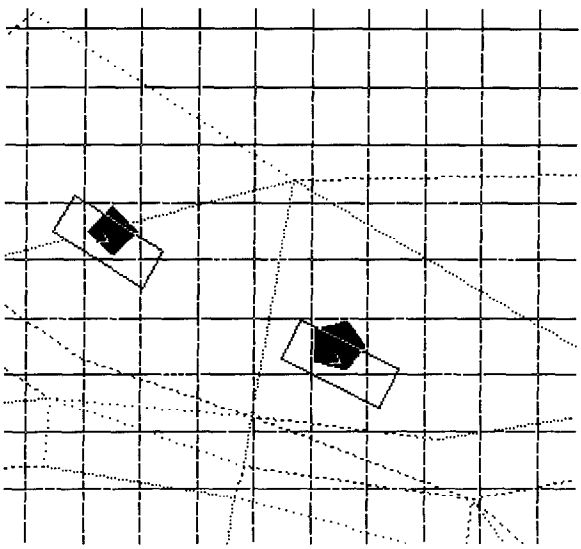


Fig. 20. The global-form view of objects as being indexable positions in a grid, i.e., the displayed marker shapes.

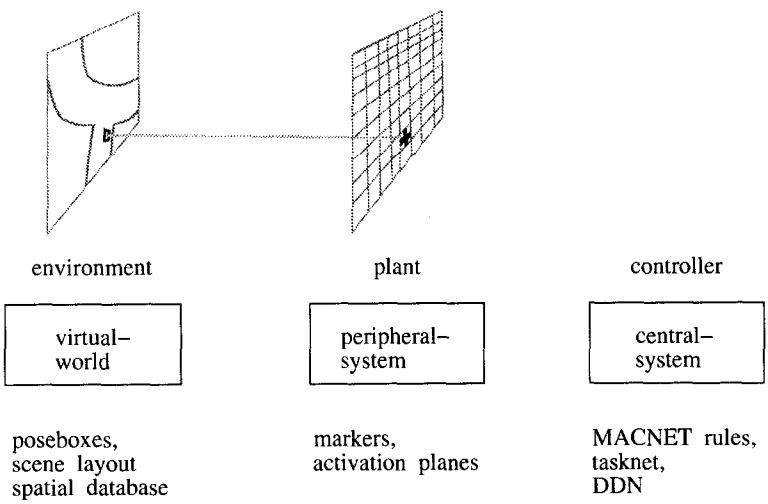


Fig. 21. Environment, plant, controller.

in HIVIS-WATCHER the marker size is about a vehicle width. As the marker tracks an object its position in the grid is calculated from the centroid of the object it is tracking so that its size plays no role in this process. The size of the markers is purely for display purposes. Although not used in HIVIS-WATCHER, the size of each marker would be important if it represented the extent of the zone of attention.

Chapman's program "BLOCKHEAD" [24] (which copies stacks of blocks in the "blocks world" domain) provides an illustration of how the real world might be modelled to comply with this analogy between the optic-array and storage representations that mimic this structure.

5.3. Central-system

Fodor [38] describes the traditional separation made in cognitive science between the peripheral- (or input-) and central-systems. On the input side we have a collection of perceptual and motor processes, each of which are to a large part innate, localised to specific brain areas, and task- and domain-independent. Each element of this collection is a module of the input-system. Fodor argues that the central side is different, saying it is not modular, being instead a single homogeneous central-system. The justification for this is that anything you know can potentially be used in any cognitive task. This view is not held by all researchers, for example, Brooks [16, 17] provides a different view that uses an orthogonal separation based on task-achieving behaviours.

Agre and Chapman say that the central-system should contain no detailed internal representation of the environment and do no explicit planning (as would be done in a classical STRIPS-like planner with world models (see, for example, Chapman [23])). This central-system architecture is an exemplar of Agre and Chapman's argument that our everyday interaction with the world consists of continual improvisation where we are deciding what to do now based on how the world is now. The central-system contains a set of loosely connected "rules" written in a language called MACNET [25] (also see Howarth [53, 54]) that combine to form patterns of activity called "routines" (and which should not be confused with plans). The rules that make up these routines are not rigidly tied to particular objects in the environment (as they would be in a classical STRIPS-like planner) but instead allow a wide variety of system-environment interaction. This is achieved by the use of deictic (context-dependent) reference. One prominent use of deictic reference is in how attentional markers are used to reference attended objects rather than using explicit object names. This means that we represent and reason about the number of properties of interest (e.g., the number of attentional markers) as opposed to the number of objects in the world (see Ballard et al. [8]).

In HIVIS-WATCHER the rules in the central-system select when an operator is to be used and what arguments are to be supplied to its activation. Crafting the rule-operator pairs into sequences (constructing routines) is done by making the result produced by one operator fulfill the input requirement of the next rule. However, this is not the only way a particular rule can be fulfilled, thus allowing the mechanism to react to similar situations that arise via a different route. A routine provides an abstraction for a common pattern of interaction between HIVIS-WATCHER's central-system and peripheral-system. This reduces "planning what to do next" to a matter of deciding what to do "now" based upon how the world is "now". Only the operators have access to the virtual world data structures and the central-system only receives the results of the operators. This allows the central-system to use a simplified description of the world, and only needs to have the information necessary for making its action-selection. This restricted state ensures that the system can only reason about the current situation.

5.3.1. Situatedness and normal behaviour in the central-system

Situated activity (see Norman [84] for an introduction) provides a link between social science and AI, which is important to how HIVIS-WATCHER interprets the world. Winograd and Flores [108, pp. 27–37] describe how the interpretation of the perceiver is not neutral, it is fundamentally social. The activities of agents are not planned out in detail, instead they are in a state of what Heidegger calls "thrownness" (for details see [108]). When interacting it is not possible to step back, reflect and plan. This has been investigated by Suchman [96] who identifies plans as something that can evolve out of situated activity, so that previous experience can be used to structure future activity. This distinction between thrownness and planning is similar to the difference between deictic temporal representation (e.g., previous, now, next) and McDermott's [74] useful observation that in most AI models of temporal reasoning we reason from the side, taking a step back and representing the past and the future, which allows us to give names to time points, such as dates, and measure temporal durations.

As described in [55] we can take inspiration from the work of Garfinkel [41] and Heritage [49]. For example, to begin to understand the behaviour of other actors in the scene the official-observer needs some "model" of the various behaviours it is likely to encounter during its observation of the scene. To give some idea of the complexity of this problem Heritage [49, p. 116] provides an illustrative example, in his discussion of Garfinkel's work [41]. This example is from the office-worker domain where two people walking from opposite ends of a corridor are about to pass such that one, let us call him Oscar here, gives a greeting to his fellow colleague which is ignored. This gives rise to a number of options from which Oscar can choose to explain the behaviour of his colleague. In this process of explaining why his greeting was ignored, Oscar is using what Garfinkel calls "reflexive accountability" where Oscar uses his own model of "normal" behaviour (called the "norm") or "maxims of conduct" to determine the kinds of reasons that might give rise to this unacknowledged greeting, We only really become aware of our "model" of other peoples behaviour when something goes wrong (i.e., not as expected or becomes "broken"—Winograd and Flores describe Heidegger's usage of this term [108, pp. 36–37]). In our example with Oscar, the norm becomes apparent (i.e., he tried to work out what might cause his colleague not to greet him) when it was breached (i.e., when the greeting was not returned). This element of identifying when observed behaviour deviates from the expected norm is not implemented in HIVIS-WATCHER, although it would be a useful extension. The description given here is more to illustrate that there is research that demonstrates that these models do exist.

Another example is provided by Gibson's "valence" approach [43,44] of how people/animals adaptively steer their locomotion through the environment. Gibson and Crooks [44] describe what they call a "field of safe travel" which is a representation of the region of psychological forces or vectors that move with the agent through the environment. It represents the awareness of what behaviour is possible, rather like a complex form of path prediction. In the *surveillance problem* we are interested in how the official-observer can employ its knowledge from driving (i.e., how it uses the field of safe travel) to interpret its perception of the participants in the observed scene.

The agent model used here is an approximation of a human observer's model of a scene object's normal conduct, which we call the "typical-object-model". So that we do not have to go into to much detail here let us just use the following definition.

Definition 8. A *typical-object-model* is a collection of rules (or rather elements of routine behaviour expressed using the MACNET language) that describe the official-observer's knowledge about how objects behave.

Basically the typical-object-model is used to reason about the various aspects relating to the attended object it is run upon. These aspects include those given in Section 5.2.2 as well as aspects relating to properties of the attended object itself such as speed, direction and location attributes (such as the region types, shown in Fig. 4, used for contextual indexing). The typical-object-model is used to select which operators to run next based on the current aspect values. Further details about the typical-object-model are given in [54].

Although all the MACNET rules in HIVIS-WATCHER could have been implemented as one homogeneous collection by repeating the typical-object-model set of rules, with one set for each object we might want to reason about at the same time. To simplify things in HIVIS-WATCHER there is just one typical-object-model that is run for each attended object. The effect of either approach is the same. The first simplifies some data management, the second reduces the number of rules.

5.3.2. Elements for task-based control in the central-system

In HIVIS-WATCHER the central-system also includes other elements in addition to MACNET rules. It contains a small amount of memory in the form of the Bayesian networks used for task-based control, together with additional inference rules (described in Section 7.2.1) that enable local object-centred information to be combined into the more global viewpoint of the global-form. We describe how these operate in Sections 6 and 7.

5.4. Summary

This section has described the three main elements of HIVIS-WATCHER shown in Fig. 14. We have covered the preattentive and attentive operators, the typical-object-model and other forms of reasoning used in the central-system. Next we describe how task-based control ties these disparate elements together enabling the individual results from attending to different objects to be integrated and how useful, task relevant objects are selected in the first place.

6. A computational theory for task-based control in HIVIS-WATCHER

The above overview sets out the framework within which task-based control has been implemented. We first provide a more generalised description of the main elements of this approach and then, in Section 7, describe how it has been implemented in HIVIS-WATCHER.

6.1. Guiding computation

A fundamental step in the approach taken here is to separate the simple and complex operations that act upon the input data, and use the results from the simple operations to guide the application of appropriate complex functions. The goal here is to reduce the irrelevant application of the more computationally expensive complex operations by ensuring the reasoning performed is relevant to the task at hand. In the context of understanding what a scene object is doing, simple operations include the Gestalt primitives described in Section 5.2.1, while complex operations include attentional, agent based computations described in Section 5.2.2. The connection with perception is when we consider the simple operation to be a peripheral one (e.g., motion detection) and the complex one to be a foveal one (e.g., recognition). We can summarise this by considering a *general case*:

Definition 9. Given two operators `OP1` and `OP2` where

(1) `OP1` is much less complex and runs faster than `OP2`,
(2) `OP1` does not return an answer that is as useful as `OP2`,
(3) `OP1` is true in all situations where `OP2` would provide a useful result,
(4) `OP1` is false in all situations where the preconditions of `OP2` would not be met or where we do not want to apply `OP2`.

If these conditions are fulfilled, `OP1` can act as a guide for the application of `OP2`. We call `OP1` the *simple* operator and `OP2` the *complex* operator.

`OP1` is part of the preattentive stage providing a preattentive cue, and `OP2` is part of the attentive stage. We next describe how the appropriate `OP1`s and `OP2`s are chosen for a particular task, and then look at how the distinction between the behaviour of the observed objects and those engaging the perceiver, affect task-based control.

6.1.1. Policy

In HIVIS-WATCHER the user inputs its query before any processing is done. This question takes the form of the tuple, called a "policy" or surveillance task.

Definition 10. A *policy* $\langle cue, attend, ignore \rangle$ specifies the simple operator `OP1` that acts as a preattentive cue, a set of behaviours to look for, and a set of behaviours to ignore.

The policy defines those features that are interesting. Each policy has a preattentive "cue" which may be the same for more than one policy and may apply to policies other than those selected. The policies considered in this paper concern the identification of all occurrences of some specified behaviour such as "tanker refuling aircraft". This causes HIVIS-WATCHER to be blinkered to any observed behaviour that is not related to the current policy.

6.1.2. Agency and kernel

As shown in Fig. 22 there is a horizontal separation into two sections called "agency" and "kernel". The agency represents HIVIS-WATCHER's model of the behaviour of

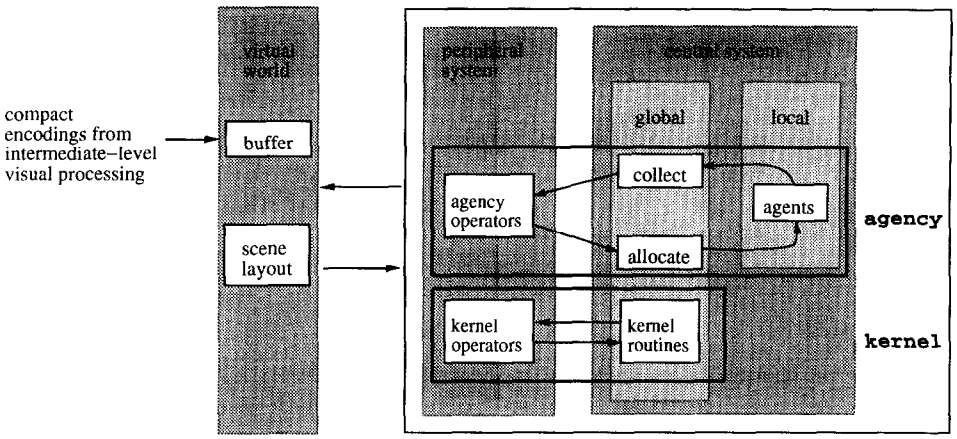


Fig. 22. The elements of HIVIS-WATCHER.

other actors/objects (i.e., those it observes in the scene). In contrast the kernel contains a model of the observer's more explicit visual behaviours. The agency reasons about the actual object shapes and relationships in the scene. The kernel takes a more abstract approach reasoning about the markers in the global grid representation. The kernel follows more closely the approach used by Agre and Chapman [2], where as the agency is an extension to their work, taking account of the environment local to the attended object. Also, in the agency, the global-form contains those elements that control which objects HIVIS-WATCHER attends. These various elements assign, provide input, run and receive the results. In practice the general case is more complicated because the selection of which OP2s to run is dependent upon factors other than the OP1 results alone.

6.1.3. Markers

HIVIS-WATCHER uses two types of attentional marker called "agent" and "kernel" that correspond to the two elements agency and kernel. The main difference between the two is that all agent markers run the same set of rules (called the typical-object-model) and that each kernel marker runs a different set of rules. The two marker types also access different classes of operator in the peripheral-system because they perform different visual actions. For example, agent markers can only be attached to buffer-addresses (called tracking) and once attached can obtain attentional property values associated with the attended object. Kernel markers, on the other hand, have their own operators which provide more extensive functionality such as being placed at any grid address, noting the position or tracking an agent marker, accessing properties about the environment (grid position) being marked, etc.

As described in Section 5.2.3 we denote the position of a marker by a small geometric shape (see Table 3) that is not part of the original image data. The position of a marker is determined by its address in the grid, and for display this is mapped (scaled, etc.) to

Table 3
The various attentional marker types used in Section 8.

display shape		global variable	type	entity
square	■	`*agent0*`	agent	*the-vehicle*
diamond	◆	`*agent1*`	agent	*the-vehicle*
pentagon	⬟	`*agent2*`	agent	*the-vehicle*
down triangle	▼	`*tail-marker*`	kernel	*the-trailing-vehicle*
up triangle	▲	`*head-marker*`	kernel	*the-leading-vehicle*
cross	✚	`*stationary-marker*`	kernel	*the-stationary-vehicle*

provide an indication of its place on the ground-plane or scene object to which it has been allocated (as shown in Fig. 20). Although the results in Section 8 do not show the grid, its presence is apparent by the location of each marker.

This discussion of the agency, kernel and markers sets out the background used in the following description of how the results from the `OP1s` are used to guide or initiate the use of the attentional `OP2s`.

6.2. *The agency*

Task-based control is used to reduce the computation performed when processing a stream of frame updates by resolving three key problems concerning:

(1) the *computational load* of determining what all the objects are doing;
(2) the *amount of evidence* from the temporally evolving mass of input data, an issue even with the relatively compact representation provided by the set of pose positions;
(3) the *viewpoint integration* of the aspects from different attended objects (a more complex form of the *spatial arrangements problem* introduced in Section 7.2.1).

The official-observer is looking for "interesting" task relevant behaviour such as some particular object behaviour, or interactions between two or three objects that fulfill some measure of similarity to an instance from the perceiver's set of known behaviours. These interactions may take a number of image frames, and the interaction might complete or be initiated out of shot. The typical-object-model (introduced in Definition 8) does not maintain much contextual information about individual objects, in accordance to the deictic approach, enabling attention to be switched between different objects and the typical-object-model swiftly applied, something that would be difficult if an extensive historical context had to be maintained. However, this means that the local-form is too reactive and dumb to provide useful results directly and is at the wrong computational level to act as an attentional controller for the official-observer.

To enhance these features so that the central-system can do event reasoning, the global-form has been added. The global-form is used to collect all the relevant agent results together, to produce a single consistent story capturing those features of the scene's happenings that are deemed interesting to the official-observer.

6.2.1. Allocate

The objective of the ALLOCATE component is to direct attention. The allocation process assigns an attentional agent marker to a scene object so that the typical-object-model can both generate deictic state descriptions and select pertinent operators appropriate for this object's current behaviour. Fig. 22 shows that the AGENTS component is part of the local-form, this is because when HIVIS-WATCHER attends to each object it can only obtain information about the object and the object's local environment. In contrast ALLOCATE, part of the global-form, has the official-observer's global view of the objects in the scene. ALLOCATE consists of two parts: a task-dependent "cue" theory of how relevant preattentive data is processed; and an "allocation" theory of how the result from this processing is used to assign an attentional marker to an object. As shown in Table 3 these attentional markers have names like `*agent0*` so we call each attended object an "agent". The cue theory provides a framework for representing the Gestalt primitive results and reasoning about them to obtain an ordered list of interesting figures. The allocation theory consists of three main functions that guide the reasoning of the runtime system:

(1) *focus-of-attention*, which only updates the selected set of target hypotheses and their associated functions;
(2) *selective-attention*, which continues to dynamically select the most interesting hypothesis to watch;
(3) *terminate-attention*, which stops all activities associated with a target hypothesis once it has been confirmed or denied to an acceptable degree of confidence.

By identifying those objects that the policy deems interesting, these provide a solution to the *computational load problem* introduced at the beginning of Section 6.2. Terminate-attention is determined from data given by COLLECT, which is used to decide what to do when a situation is identified (a task goal achieved). Fast identification of uninteresting behaviour allows more time to be spent looking for relevant features. The ALLOCATE component is used to decide whether HIVIS-WATCHER should ignore the objects involved in the exhibited behaviour or continue to attend to them. This allocation process is task driven not data driven, with the processing performed also made context-dependent via the typical-object-model.

6.2.2. Collect

The COLLECT component collects together the results from each agent and provides feedback of identified behaviour to ALLOCATE. Fig. 22 shows that like ALLOCATE, COLLECT is also part of the global-form. Each agent provides a sequences of activities which COLLECT combines into an episode. This episode takes into account everything (or at least the important features) that have been detected from the recent frame updates. The COLLECT component measures the typical-object-model results from each agent against the other "happenings" in the scene to determine whether the observed action is legal (typical, untypical) or illegal. The global-form operates with a longer time frame than the agents in the local-form and is able to combine the various deictic states detected by each agent into a continuous story of what is happening in the scene. This contextual knowledge is used to enable HIVIS-WATCHER's global-form to make sound predictions about what is happening in the scene.

In Section 7.2 we provide an implementation of the COLLECT component that shows how it can solve the *viewpoint integration problem* caused by the use of deictic representation. This problem is simplified by the fact that the official-observer originally selected which objects to attend and thus the execution of an agent's typical-object-model is like the official-observer considering the possible next actions of an object it is attending. The COLLECT component also reduces the *amount of evidence* that needs to be retained by identifying the task relevant values.

6.2.3. Task-based control of agency

Fig. 22 shows how ALLOCATE and COLLECT are linked together, being two parts of the task-based control theory that first identifies what to look at and that then integrates the results from the attentional process into the larger picture. Next we look at how task-based control operates in the kernel.

6.3. The kernel

Fig. 22 shows how the kernel and agency co-exist in HIVIS-WATCHER. There are two problems with the local deictic reasoning outlined in Section 6.2: (1) although useful for describing single or spatially related binary objects, it is not suitable for coordinating non-local relationships, and (2) the expression of such non-local object behaviour as an identifiable episode may take longer than can be easily modelled using the local-form of representation. The first problem is another version of *viewpoint integration* and the second arises from the use of local spatio-temporal reasoning. While COLLECT addresses this second problem in the agency, we need a different approach to cope with non-local object behaviour. The typical-object-model does not retain a temporal history of past events and, because of this we are not able to identify accomplishments (this depends on being able to recognise that the situation after the state change is different from the situation before the state change). We call this the *temporal history limitation problem.*

To address these we introduce the kernel, a more global component, that can (1) operate over longer episode-length time spans, (2) provide a spatial representation to coordinate perceiver references to scene objects and other parts of the environment, (3) reason about these perceiver references. The basic architecture used in HIVIS-WATCHER, from the work of Agre and Chapman, is able to meet these requirements. Illustrations of this architectures suitability for different application domains are given by the programs PENGI [1,2], BLOCKHEAD [24] and SONJA [25]. Agre and Chapman also provide arguments for why this approach is appropriate for reasoning about evolving, dynamic visual data from the environment. In PENGI and SONJA the central-system takes the role of the person playing a video game. In HIVIS-WATCHER the kernel part of the central-system takes the role of the official-observer, providing a model of the observer's own behaviour as it interprets the evolving data unfolding in the scene under observation. With the kernel routines used to determine if a particular policy behaviour has taken place. Each stage in the routine corresponds to an accomplishment of observed object behaviour, and the transition through each stage of the routine in the correct sequence describes an episode. The peripheral-system contains appropriate visual operators that are manipulated by these routines of behaviour.

While ALLOCATE and COLLECT form an integrated couple, ALLOCATE is also used to identify task relevant objects for the kernel. This is done via the allocated agent marker's typical-object-model producing an effector result that moves (or "warps" as Agre and Chapman would say) a kernel marker on top of the agent marker "telling" the kernel an interesting object has been found. While communication between agency and kernel could have been done in the central-system, using attentional markers provides an elegant solution.

6.4. Summary

The separation between agency and kernel is given credence by the observation, described in Section 4, that egocentric and exocentric viewpoints are incompatible. The agency central-system contains the AGENT component that performs local, exocentric reasoning about each attended object, with the ALLOCATE and COLLECT components being used to convert the separate exocentric results into one overall egocentric, global-form picture. In contrast the kernel central-system only does egocentric, global-form reasoning about the objects and other scene features this part of HIVIS-WATCHER chooses to attend.

This has covered the theory of task-based control in the agency and kernel. Next we describe how the various elements in the agency (principally ALLOCATE and COLLECT) are implemented and provide an example set of routines that are used in the kernel, which makes clear the kind of roles the kernel markers in Table 3 play.

The complex operator that HIVIS-WATCHER is guiding is the model-matcher (HIVIS-WATCHER's attentional recognition process) together with the associated attentional reasoning that uses the model-matcher results. While the stream of compact encodings hides the cost of performing model-matching we assume that it is worth minimising. Both the agency and kernel minimise the number of attentional marker applications to scene objects, while still fulfilling the surveillance task policy (HIVIS-WATCHER's main objective).

7. An implementation of task-based control in HIVIS-WATCHER

We have separated the operators in the peripheral-system into: preattentive ones that are global, simple, and of low-cost; and attentive ones which are applied to a single object and are more complex. The preattentive operators are used to guide application of attentive ones. Example preattentive operators include mutual-proximity and gross-change-in-motion which are described below. The motivation behind the preattentive cues chosen here, was their potential usefulness on low-level data such as the identification of possible objects from clustering flow-vectors (for example, Gong and Buxton [45] describe how knowledge about a known ground-plane is used to develop expectations of likely object motion). Once we have these coarse descriptions, and if they comply with the preattentive cue, then they become candidates for further attentional processing such as model-matching (or some other form of object-recognition) to obtain aspects about

Table 4
The DDN node types.

Preattentive operator	Node	States and function
mutual-proximity	$N_i(t)$	{not-near, nearby, close, very-close, touching} The proximity of object i to it's nearest neighbour at time t.
	$R_{(i,j)}(t)$	{ignore, watch} The result value saying how interesting the mutual-proximity relationship between the pair of objects i and j is.
gross-change-in-motion	$PM_i(t)$	{stat, slow, move} For the motion-prior value is moving (i.e., prior moving). stat is short for "stationary".
	$PS_i(t)$	{stat, slow, move} For the motion-prior value is stationary (i.e., prior stationary).
	$R_i(t)$	{ignore, watch} The result value saying how interesting the motion-prior change of object i is.

the object. Basically, once we have found where something interesting is, we then try and workout what it is.

7.1. Allocate

Here we describe the implementation of the two different preattentive cues introduced in Section 5.2.1 and used in the examples in Section 8. Each of these preattentive cues is a simple operator (OP1) that acts as a guide for its more complex operator (OP2). The preattentive cue, OP1, is used by ALLOCATE (see Section 6.2.1) to determine whether an agent marker should be allocated to a scene object. Once such an agent marker is allocated, the more complex process of obtaining visual aspects of the agent by attending to the scene object can be performed. To calculate this preattentive cue we have used a Bayesian network to model ALLOCATE's cue theory. We use Bayesian networks (background details are given by Pearl [85]) because of the uncertainty present in the results provided by the Gestalt primitives that we are using as preattentive cues. The Bayesian network approach can both represent and reason about this uncertainty, and are attractive because of their mathematical foundation. Other approaches like Fuzzy Logic might provide a viable alternative, but these are not considered here.

We do not extend the computational theory of Bayesian networks, although we do present a novel use of them, called "Dynamic Decision Networks" (DDN). This uses Bayesian networks in a dynamic context, with the graph structure updated in discrete steps (or clock ticks) to reflect the contents of the scene. The graph used by the DDN evolves over time to represent the evolving spatio-temporal relationships being modelled. These simple graph structures form singly connected tree structures that execute quickly. Details of how these are constructed is given in Appendix A.

Table 5
DDN conditional probabilities for the proximity mechanism. Note that (1) $M_{R|N}$ is not used directly, and is present to define $M_{R|NN}$; (2) CJP is a function that takes a list of 2D conditional probability matrices and returns a single higher-order matrix by performing the equation $P(a_i \mid b_k, d_l) = \alpha P(a_i \mid b_k)P(a_i \mid d_l)$ described by Kim and Pearl [63]; and (3) illustrated bottom right are the subgraphs for which child node conditional probability matrices are provided.

$M_{R|N}$ =

	ignore	watch
not-near	1.0	0.0
nearby	0.4	0.6
close	0.3	0.7
very-close	0.2	0.8
touching	0.0	1.0

$M_{R|R}$ =

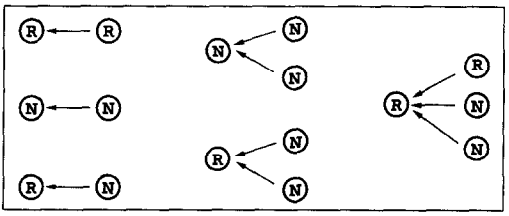
	ignore	watch
ignore	0.9	0.1
watch	0.1	0.9

$M_{N|N}$ =

	not-near	nearby	close	very-close	touching
not-near	0.9	0.1	0.0	0.0	0.0
nearby	0.1	0.8	0.1	0.0	0.0
close	0.0	0.1	0.8	0.1	0.0
very-close	0.0	0.0	0.1	0.8	0.1
touching	0.0	0.0	0.0	0.1	0.9

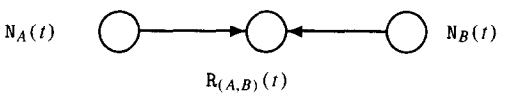
Combining joint probabilities

$M_{N|NN} = \text{CJP}([M_{N|N}, M_{N|N}])$
$M_{R|NN} = \text{CJP}([M_{R|N}, M_{R|N}])$
$M_{R|RNN} = \text{CJP}([M_{R|R}, M_{R|N}, M_{R|N}])$



7.1.1. Mutual proximity

The mutual-proximity test is used to identify when two proximate objects are travelling in the same direction. The spatio-temporal proximity function used for OP1 is the proximity test described in Section 5.2.1 which provides the qualitative values used by $N_i(t)$ in Table 4. In the DDN used here the basic graph primitive is depicted as:



This graph describes a mutual-proximity relationship between objects A and B, capturing the primitive notion of object A being near B and B being near A. The relationship node R denotes this mutual-proximity and holds a belief value that reflects how interesting this is. The states and purpose of the two node types are given in Table 4. The directed edges hold the fixed conditional probabilities given in Table 5. These matrices were derived from careful consideration of the possible input values. The spatial relationship is specified by the matrix $M_{R|N}$, capturing the belief that a proximity relationship becomes increasingly more interesting as two objects are identified as being nearer to each other.

Fig. 23(a) shows that when we evolve the network over time we make use of maintained official-observer object references to the image data. If, at the current time point, *A* and *B* are *still* near to each other we can form temporal links from the previous node to the current ones as shown in Fig. 23(b). From the relative temporal position of the previous time point the temporal links are to the next time point. These directed temporal edges each hold one of the matrices representing temporal continuity. Fig. 23(c) shows how we extract a singly connected network from the graph in part (b). This is done for the parsimonious reason that a tree is easier to evaluate than a multiply connected network. This graph simplification preserves the results obtained from the previous value propagation allowing them to contribute effectively to the formation of the new belief about the relationship that now holds between object *A* and *B*. If there is more than one relationship present then a separate tree is built for each one, enabling, once all propagations are complete, the pairs to be ordered in terms of their "interestingness" relevant to the current task. The network changes structurally over time to reflect the relationships between the objects of interest. For example, Fig. 24 shows what happens if $\mathtt{N}_B$ is missing for two time points (t_3 and t_4) due, say, to occlusion and the nearest neighbour to $\mathtt{N}_A$ now becomes $\mathtt{N}_C$. When $\mathtt{N}_B$ is detected again it is identified as being nearest and the new links and nodes in the network are created to reflect this. The mutual-proximity preattentive cue is used in Section 7.2 and implementation details are given in Appendix B.

7.1.2. Gross change in motion

The preattentive cue "gross-change-in-motion" is used to enable ALLOCATE to assign an agent marker to any scene object that changes from "moving to stationary" or "stationary to moving" and is based on the spatio-temporal discontinuity operator we described in Section 5.2.1. This preattentive cue would raise an attention-calling response in situations where there is a change of steady velocity, such as, when a vehicle stops at a junction to assess the road conditions, or when people get up after sitting in the same place for some time.

To denote the current typical motion of an object we use a prior value called motion-prior such that `motion-prior` $\in$ {`moving`, `stationary`}. When the prior value is `moving`, then stopping is unusual; when the prior value is `stationary`, moving is unusual. We assume that normal behaviour of an object for motion-prior is `moving`. If the observed motion of an object is different from the motion-prior then abnormal behaviour is taking place (see Section 5.3.1). If this abnormal behaviour persists it becomes the norm and the motion-prior is changed to reflect this.

In Table 6 we describe various conditional and joint probability matrices, with the "Node type allocation" table describing the patterns of graph node to which these matrices are assigned. Fig. 25 gives an example of the simple graph structure produced, which is like half of the graph used for the mutual-proximity preattentive cue. Although the graph structure may stay the same the node types change to reflect the motion prior, i.e., the P* nodes in Fig. 25 are either of type PM or PS from from Table 4. In Table 4 the qualitative value `slow` is included to take account of vehicles that hedge forward at a junction. The gross-change-in-motion preattentive cue is used in Section 7.3 with implementation details given in Appendix C.

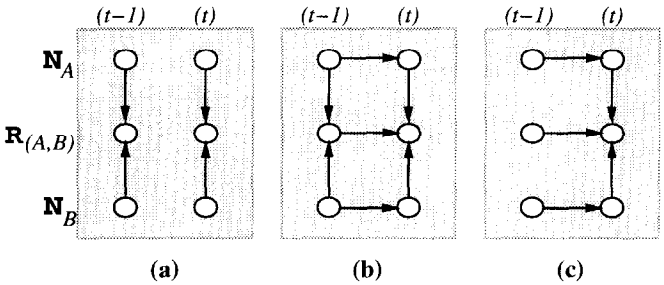


Fig. 23. Graph linking structure.

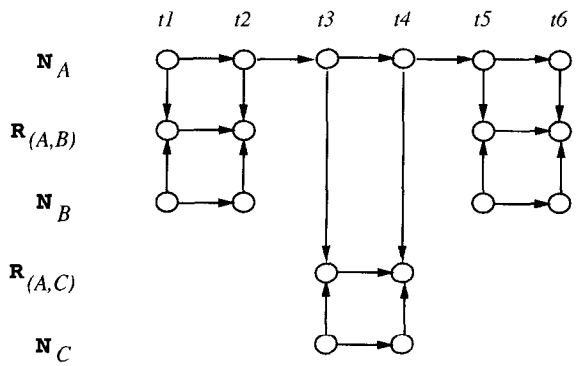


Fig. 24. Change over time.

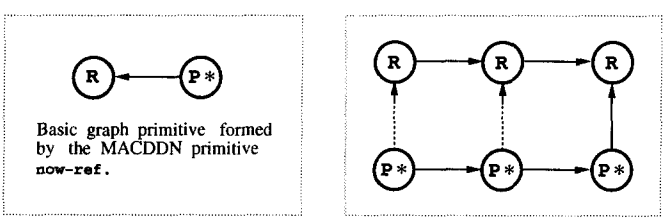


Fig. 25. Basic graph primitive.

7.1.3. *After allocate*

Once ALLOCATE has identified the potentially interesting object(s) and HIVIS-WATCHER has then worked out what each interesting object is and then done appropriate task relevant reasoning relating to this object, the next process is to form some understanding of what the object(s) of interest are doing. HIVIS-WATCHER contains two different approaches to this. The first is the COLLECT component, and the second is provided by the kernel routines. We look at each of these in the next two subsections.

Table 6
DDN conditional probabilities for gross-change-in-motion. Note that MO denotes the general description of motion and is used by the subgraphs described in the "Node type allocation" table.

$M_{R|PM}$ =

	ignore	watch
stat	0.0	1.0
slow	0.4	0.6
move	1.0	0.0

$M_{R|PS}$ =

	ignore	watch
stat	1.0	0.0
slow	0.6	0.4
move	0.0	1.0

$M_{MO|MO}$ =

	stat	slow	move
stat	0.9	0.1	0.0
slow	0.1	0.8	0.1
move	0.0	1.0	0.9

Combining joint probabilities

$M_{MO|MOMO} = CJP([M_{MO|MO}, M_{MO|MO}])$
$M_{R|RPM} = CJP([M_{R|R}, M_{R|PM}])$
$M_{R|RPS} = CJP([M_{R|R}, M_{R|PS}])$
$M_{R|PMPM} = CJP([M_{R|PM}, M_{R|PM}])$
$M_{R|PSPS} = CJP([M_{R|PS}, M_{R|PS}])$

Node type allocation

child	parents	matrix
PM	PM	$M_{MO\|MO}$
PM	PS	$M_{MO\|MO}$
PM	PM PM	$M_{MO\|MOMO}$
PM	PM PS	$M_{MO\|MOMO}$
R	PM	$M_{R\|PM}$
R	R PM	$M_{R\|RPM}$
R	PM PM	$M_{R\|PMPM}$
PS	PS	$M_{MO\|MO}$
PS	PM	$M_{MO\|MO}$
PS	PM PS	$M_{MO\|MOMO}$
PS	PS PS	$M_{MO\|MOMO}$
R	PS	$M_{R\|PS}$
R	R PS	$M_{R\|RPS}$
R	PS PS	$M_{R\|PSPS}$

7.2. *Collect*

The purpose of the COLLECT component, introduced in Section 6.2.2, is to pull together related results from different agent markers. To do this COLLECT takes the deictic viewpoints of two mutually-proximate objects, such that each deictic viewpoint is from an attentional agent providing deictic state descriptions of the other object. As described in Definition 9 we are using two types of operator OP1 and OP2, where the simple OP1 operator is used to guide the applicability of the more complex attentional ones. The complex operators, OP2, used here are the positional, heading and speed tests given in Section 5.2.2. All three of which are guided by the same OP1—the preattentive cue for mutual-proximity described in Section 5.2.1. Section 7.1.1 described how this is implemented using the DDN to identify relevant pairs of objects, i.e., *the-other* relative to *the-refobj*. (Note that conditions (1)–(4) of Definition 9 are fulfilled—the complex operator only works when given the result saying which, if any, is the nearest object.) The result is a deictic reference, such as behind-me, describing the positional aspect between *the-other* and *the-refobj*. Often both objects in the mutually-proximate pair are attended resulting in a description from both objects of its twins relationship.

COLLECT combines the two disparate deictic viewpoints, to say whether they indicate the presence of likely overtaking, following, queueing or some other, unknown but similar, behaviour. These three identifiable behaviours seem the most probable for mutually-proximate objects with queueing being more likely if the scene objects are both stationary. While the input agent marker results could be integrated using a finite state machine approach, here a Bayesian network is used. Although the Bayesian approach does not represent legal transition information it does hold an evolving model of the evidence collected so far. Rather than input the deictic states directly to the Bayesian network, a preprocessing stage is used to reduce the state-space by unifying the respective deictic properties to produce a single qualitative value for each property. Inputting the information directly to a Bayesian network is possible, it would just require higher dimension conditional probability matrices to express the relationships contained in the qualitative unification stage. First we discuss this unification stage and then look at how the Bayesian network, called TASKNET, is implemented.

7.2.1. Unifying twined deictic states

To describe how the results from two attended objects are integrated let us first consider the positional deictic states. Spatial representation plays an important part in the *surveillance problem*, such as the representation of binary spatial arrangements of the observed scene objects. These spatial arrangements are difficult to define in an absolute coordinate system (as described in Section 3.4) and to overcome this *spatial arrangements problem* we use the deictic representation introduced in Section 5.2.2. However, this introduces the problem of composing the deictic descriptions of the objects involved into a common representation that can be used by the official-observer. This problem is illustrated and resolved by the composition rules for relative position described in Table 7. These rules describe a matrix that contains symmetric values so only half of the matrix needs to be specified.[14] The approach used here is based on the local-form and once each object's intrinsic front is identified, it provides a natural framework within which to describe the spatial arrangements of objects in the scene.

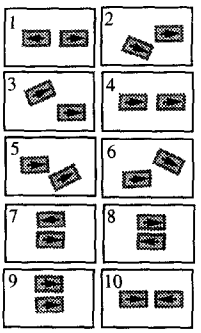
We can use the same framework to combine the values for relative heading and speed.[15] Table 7 shows three tables, each containing a set of statements of the form "$a\ b \rightarrow c$" where on each line the two viewpoints a and b are composed to form the composite c. This composite can be thought of as an iconic model representing the fusion of both viewpoints given by a and b. Some illustrations of the icons for relative position are given that correspond to the numbered rules. Each small illustration shows a prototypical position relationship of a pair of poseboxes. Where each posebox (representing results

[14] This is the format used in the LISP implementation and which is given to a LISP macro that generates the full matrix used at runtime.

[15] Unlike position, the heading and speed pairwise relationships do have a valid inverse, e.g., if `(a faster-than b)` is `true` then `(b slower-than a)` is `true`. This means that as both objects are attended, combining the results for the relative differences of heading and speed is similar to combining the values for heading and speed of the two attended *the-refobjs*. For example, as described in [60], an object's own heading can be described using eight qualitative values with the combination of such heading values from two paired objects producing similar composite results to those given for relative heading in Table 7. In this paper I have chosen to use relative values so that the same (extendible) framework is used for each property.

Table 7
The composition rules for relative position, heading and speed.

	Positional viewpoints		Composite
1	behind-me	behind-me	→ back-to-back
2	behind-me	leftside-of-me	→ trans-overtaking-back
3	behind-me	rightside-of-me	→ trans-overtaking-back
4	behind-me	infront-of-me	→ following
5	leftside-of-me	infront-of-me	→ trans-overtaking-front
6	rightside-of-me	infront-of-me	→ trans-overtaking-front
7	leftside-of-me	leftside-of-me	→ same-side-adjacent
8	rightside-of-me	rightside-of-me	→ same-side-adjacent
9	leftside-of-me	rightside-of-me	→ overtaking-passing
10	infront-of-me	infront-of-me	→ head-on



Heading difference viewpoints		Composite
parallel-to-me	parallel-to-me	→ alongside-consecutive
similar-heading-to-me	parallel-to-me	→ alongside-consecutive
similar-heading-to-me	similar-heading-to-me	→ alongside-consecutive
similar-heading-to-me	perpendicular-to-me	→ converge-diverge
perpendicular-to-me	perpendicular-to-me	→ converge-diverge
opposite-to-me	opposite-to-me	→ opposing
parallel-to-me	perpendicular-to-me	→ unknown
parallel-to-me	opposite-to-me	→ unknown
opposite-to-me	perpendicular-to-me	→ unknown
similar-heading-to-me	opposite-to-me	→ unknown

	Speed difference viewpoints		Composite
⋆	slower-than-me	faster-than-me	→ disparate
	similar-speed-to-me	faster-than-me	→ disparate
⋆	similar-speed-to-me	similar-speed-to-me	→ flowing
	slower-than-me	slower-than-me	→ flowing
	slower-than-me	similar-speed-to-me	→ flowing
	moving-slowly-like-me	faster-than-me	→ flowing
	faster-than-me	faster-than-me	→ flowing
⋆	moving-slowly-like-me	moving-slowly-like-me	→ sluggish
	slower-than-me	moving-slowly-like-me	→ sluggish
	stationary-like-me	similar-speed-to-me	→ sluggish
	stationary-like-me	faster-than-me	→ sluggish
	moving-slowly-like-me	similar-speed-to-me	→ sluggish
⋆	stationary-like-me	stationary-like-me	→ congested
	slower-than-me	stationary-like-me	→ congested
	stationary-like-me	moving-slowly-like-me	→ congested

Table 8
The TASKNET node types.

Node	States and function
$LE_{(i,j)}$	{overtaking, following, queueing, unknown} The likely-episode of the marker pair (i, j).
$UP_{(i,j)}$	{back-to-back, trans-overtaking-back, following, trans-overtaking-front, overtaking-passing, same-sides-adjacent, head-on} The unified positional state of the marker pair (i, j).
$UH_{(i,j)}$	{alongside-consecutive, converge-diverge, opposing, unknown} The unified heading difference of the marker pair (i, j).
$US_{(i,j)}$	{disparate, flowing, sluggish, congested} The unified speed difference of the marker pair (i, j).

from the model-matcher described in Section 2) contains an arrow denoting its front. In the table for speed difference the starred ($\star$) lines are the main entries with the others included for completeness. For example, the value `disparate` denotes different speeds and so is best described by one vehicle going faster than another. The deictic viewpoints in the tables are given in an abbreviated form, for example, `leftside-of-me` is short for *the-refobj* saying that `the-other-is-on-the-leftside-of-me`. These composition rules resolve the deictic viewpoints of the selected pairs of agents to provide the official-observer's interpretation of the relationship. This acts as input to the pair's dedicated TASKNET which, as described next, builds a coherent interpretation of the temporally evolving object relationship.

7.2.2. *The TASKNETs*

The TASKNETs in HIVIS-WATCHER are implemented using a static Bayesian network to guide the computation performed according to the selected surveillance task. This is similar to Rimey and Brown's [87] use of Bayesian networks to actively direct the camera using geometric relationships. Here, however, our application requires scene surveillance based on spatio-temporal reasoning relative to a static camera. Each TASKNET has a temporally fixed tree structure with conditional probabilities that are defined before runtime. A TASKNET is allocated to each pair of indexes[16] to provide a description of how observed properties relate to known behaviour. The input nodes represent key features relevant to the task that the TASKNET has been constructed to identify. The output root node represents the overall belief, based on the evidence collected so far, in a set of candidates consisting of the wanted task, related but unwanted tasks, and the default unknown task.

In Section 8 we use TASKNETs to distinguish between likely overtaking and following behaviour based on the unified values for position, speed and heading. In this implementation the TASKNETs use the nodes described in Table 8 and have the simple structure shown in Fig. 26(a),[17] with $UP_{(i,j)}$, $US_{(i,j)}$ and $UH_{(i,j)}$ being the root

[16] Each "index" is a buffer-address with the pairing created by the preattentive DDN for mutual-proximity.
[17] LE stands for Likely-Episode, UP stands for Unified Positional state, UH stands for Unified Heading difference, US stands for Unified Speed difference, with the $*$-PRO forms representing the internal nodes of the Bayesian network that are used for PROcessing.

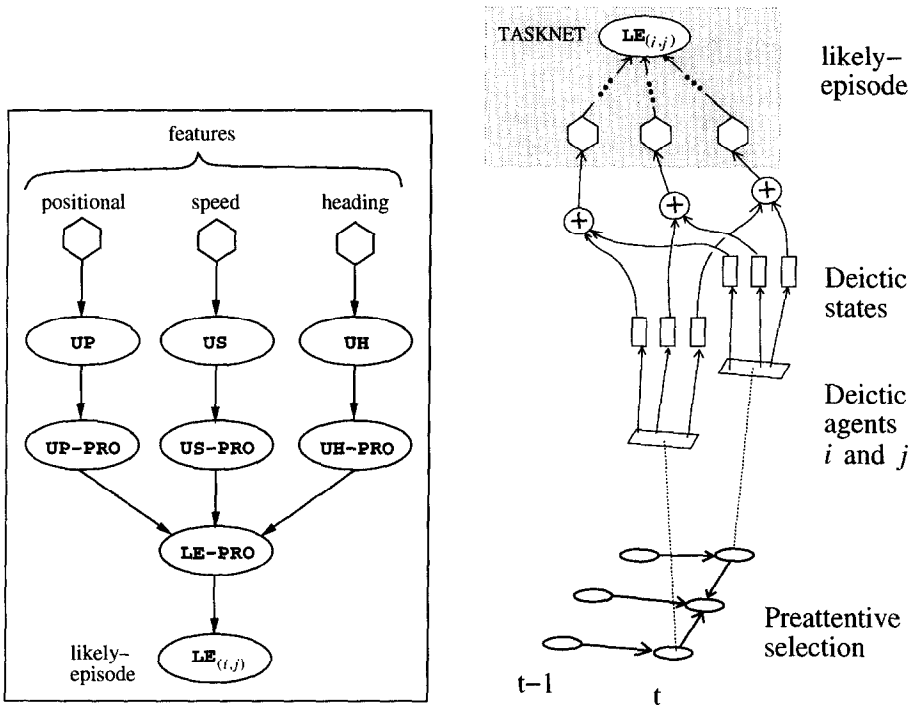


Fig. 26. The TASKNET graph. (a) The TASKNET network used in the proximity example. (b) A summary of the task-based control mechanism.

nodes, and $LE_{(i,j)}$ being the output node. These nodes are related according to the conditional probabilities given in Table 9. Each TASKNET's relationship to the agency part of HIVIS-WATCHER is illustrated by the pictorial summary in Fig. 26(b). At the bottom the DDN graph provides preattentive selection of the two deictic agents. The results from the mutual-proximity nodes in the DDN (index pairs of the form (i, j)) are used to select which agents to run on which indexes. We can combine the results from the correct agents running on indexes i and j, because i and j are in the scope of the global-form. Next, up from the selected agents, are the deictic states generated by the typical-object-model run on each agent. The composition of these agent results produces a composite feature value, which is integrated together with other evidence to produce an estimate of the likely episode that the two agents are engaged in. For example, the positional feature obtained from the agent results has two components: the index-reference of the nearest object; and the deictic-event-primitive $\in$ {`behind-me`, `rightside-of-me`, `leftside-of-me`, `infront-of-me`}. The input given to $TASKNET_{(i,j)}$ is the result from an appropriate composition matrix $\oplus$ (such as those described in Section 7.2.1). The result from running the Bayesian inference algorithm on each TASKNET is used to guide agent allocation, as described in Section 6.2.1,

Table 9
TASKNET conditional probabilities for the road-traffic example.

$M_{\text{LE|UP}}$ =

	overtk	follow	queue	unk
back-to-back	0.0	0.0	0.0	1.0
trans-overtaking-back	0.6	0.0	0.4	0.0
following	0.0	0.6	0.4	0.0
trans-overtaking-front	0.6	0.0	0.4	0.0
overtaking-passing	0.6	0.0	0.4	0.0
same-sides-adjacent	0.0	0.0	0.0	1.0
head-on	0.0	0.0	0.0	1.0

$$M_{\text{UP|UP}} = \forall o, i \in \text{UP},\ \text{P}(o \mid i) = \begin{cases} 0.70 & \text{if } o = i, \\ 0.05 & \text{if } o \neq i. \end{cases}$$

$M_{\text{LE|US}}$ =

	overtk	follow	queue	unk
disparate	0.7	0.1	0.1	0.1
flowing	0.5	0.5	0.0	0.0
sluggish	0.15	0.15	0.5	0.2
congested	0.0	0.0	0.7	0.3

$$M_{\text{US|US}} = \forall o, i \in \text{US},\ \text{P}(o \mid i) = \begin{cases} 0.79 & \text{if } o = i, \\ 0.07 & \text{if } o \neq i. \end{cases}$$

$M_{\text{LE|UH}}$ =

	overtk	follow	queue	unk
alongside-consecutive	0.4	0.4	0.2	0.0
converge-diverge	0.45	0.35	0.2	0.0
opposing	0.0	0.0	0.1	0.9
unknown	0.0	0.0	0.0	1.0

$$M_{\text{UH|UH}} = \forall o, i \in \text{UH},\ \text{P}(o \mid i) = \begin{cases} 0.79 & \text{if } o = i, \\ 0.07 & \text{if } o \neq i. \end{cases}$$

Combining joint probabilities
$M_{\text{LE|UPUSUH}} = \text{CJP}([M_{\text{LE|US}}, M_{\text{LE|UH}}, M_{\text{LE|UP}}])$

by identifying the likelihood that the relationship is either `overtaking`, `following`, `queueing` or `unknown`, and then, as described in Appendix D, using ALLOCATE's allocation theory to determine if this identified likely behaviour should continue to be attended or not.

7.3. Kernel routines

As outlined in Section 6.3, some scene object behaviours that we wish to recognise evolve in an episodic way over time. This type of surveillance task is not suitable for the TASKNET described in Section 7.2, because it requires identification of a number of distinct stages that involve different scene participants. To illustrate how kernel routines can be used to detect such an episode we will describe the routines associated with the official-observer looking for the presence of one vehicle giving way to another. The first kernel routine is initiated by the gross-change-in-motion preattentive cue, as described in Sections 6.3 and 7.1.2, which is used to detect when a vehicle stops at a junction.

The perceptual task of the official-observer involves three important entities: the first two correspond to the two roles in the giveway episode and are denoted by $\mathcal{S}$ for

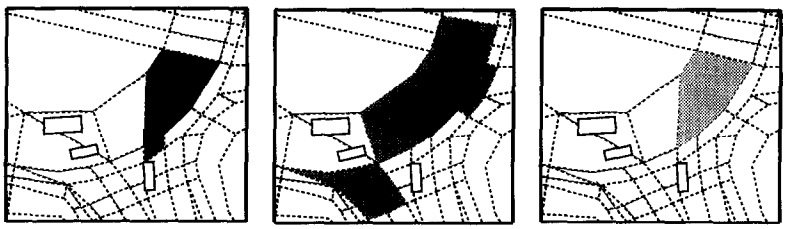
frame 192			
global variable	`*activation-plane1*`	`*activation-plane2*`	`*conflict-plane*`
entity	*the-stationary-vehicle's-path-prediction*	*the-blocking-path-prediction*	*the-conflict-area*
display	black	dark grey	light grey

Fig. 27. A table describing the regions used by the three activation planes used in the example developed here.

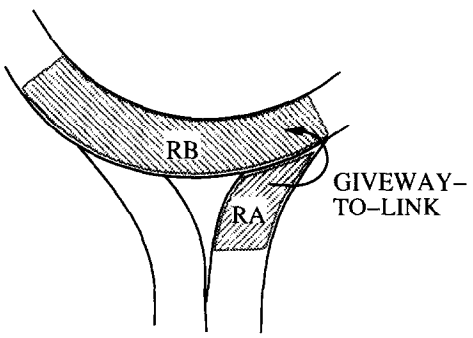


Fig. 28. RA is the giveway region linked to RB, its giveway-to-zone. See also Fig. 4.

the-stationary-vehicle, and $\mathcal{PB}$ for *the-vehicle-that-*$\mathcal{S}$*-is-giving-way-too*; and the third is denoted $\mathcal{CA}$ for *the-conflict-area* (a special region). When the two roles of $\mathcal{S}$ and $\mathcal{PB}$ have been found, an area of mutual conflict, $\mathcal{CA}$ can be identified (the space in front of $\mathcal{S}$ and through which $\mathcal{PB}$ will pass). This area links $\mathcal{S}$ to its cause. All that remains is to determine that $\mathcal{S}$ is giving way to approaching traffic, and exhibits no other plausible behaviour (e.g., broken down, parked).

We separate the giveway episode into five routines that use region-based-prediction (see Figs. 4 and 27 with more details given in [54]), and perceiver level coordination. These routines are:

- Notice-stopping-object, which on completion generates `event-gw1`. The gross-change-in-motion from moving to stationary allocates an agent, called $\mathcal{S}$, and prompts the question "why is vehicle $\mathcal{S}$ stationary?" There are a number of possible answers however, if $\mathcal{S}$ is in a giveway zone of an entry lane to a roundabout, the most likely answer is that $\mathcal{S}$ is giving way to something on the roundabout.

- Look-for-path-blocker, which on completion generates `state-change-gw2`. This stage identifies $\mathcal{PB}$. For $\mathcal{PB}$ to be blocking $\mathcal{S}$ it does not need to be physically in the way, it can also block by having "right-of-way" such that its path *will* block $\mathcal{S}$. If $\mathcal{PB}$ exists it will typically be in the giveway-to-zone corresponding to the giveway-zone that $\mathcal{S}$ is occupying. This is illustrated in Fig. 28, and using this contextual-indexing we find $\mathcal{PB}$. Having proposed that $\mathcal{PB}$ is blocking $\mathcal{S}$'s path causing $\mathcal{S}$ to avoid a collision by stopping, the next two routines are to prove that this is true.
- Work-out-conflict-area, which on completion generates `state-change-gw3`. Having predicted the paths of $\mathcal{S}$ and $\mathcal{PB}$, we intersect them to find the mutually shared conflict area, $\mathcal{CA}$. The presence of $\mathcal{CA}$ supports the proposal made in stage look-for-path-blocker, and binds both $\mathcal{S}$ and $\mathcal{PB}$ together. $\mathcal{CA}$ should be invariant during the giveway episode.
- Watch-for-enter-conflict-area, which on completion generates `state-change-gw4`. In order to determine whether $\mathcal{S}$ gives way to $\mathcal{PB}$, we wait until $\mathcal{PB}$ has passes through $\mathcal{CA}$. To determine that $\mathcal{S}$ is giving way, we only need to check that at least one object passes through $\mathcal{CA}$.
- Notice-starts-to-move, which on completion generates `event-gw5`. We then observe if $\mathcal{S}$ moves. The gross-change-in-motion from stationary to moving reallocates an agent to $\mathcal{S}$.

These five routines given above order and, as a continuous sequence, describe a temporal sequence of perceiver activity that identifies a giveway episode. These routines for identifying a giveway episode are intended to be an illustrative subset of routines that also check for things like hesitation (where there is a space that the waiting car could have pulled out into) which would contribute towards explaining away giveway behaviour and support the belief that the car has broken down or been parked.

7.4. Top-level loop

Fig. 29 shows the top-level loop which describes the order of execution of the various elements in Fig. 22. This top-level loop shows that HIVIS-WATCHER has a tightly coupled architecture where feedback is an important feature. As described in Section 5.3 the traditional separation made in cognitive science between input and central systems provides a description of the two tightly coupled components in HIVIS-WATCHER. The input system holds the functionality that obtains object aspects, while the central system guides which objects should be attended so that their aspects can be obtained to fulfill the given surveillance task. The separation of preattentive and attentive processing, and the use of a task directed central mechanism provides what is needed for identifying those task related features that the official-observer is looking for. Using this architecture HIVIS-WATCHER provides timely surveillance information about what is happening in the scene.

The whole of HIVIS-WATCHER constitutes the official-observer (or rather some implemented subset of the ideal official-observer) with, as described in Section 6, a distinction made between the two sections important to task-based control that we have called agency and kernel. The agency part is more concerned with reasoning about the

(1)	Update buffer with new frame of compact encodings.		
(2)	Run peripheral-system and then present inputs to the central-system.	(2.1)	Run agency peripheral-system.
		(2.2)	Run kernel peripheral-system.
(3)	Run the agency central-system and then run the kernel central-system.	(3.1.1)	Allocate agents.
		(3.1.2)	Run each allocated agent's typical-object-model.
		(3.1.3)	Collect results.
		(3.2)	Run kernel routines.
(4)	Run the effector-system.		

Fig. 29. Top-level loop of HIVIS-WATCHER.

observed behaviour of the actors/objects in the scene. Where as the kernel is concerned with the range of visual behaviours that the official-observer uses when it is engaged in some surveillance task. These are both instances of the selective-attention machinery used for task-based control. It is just that the actions of the official-observer that they control are different (as described in Section 6) and in some cases, such as that discussed in Section 8.4, it is useful to combine their operation.

7.5. Implementation

HIVIS-WATCHER has been implemented in LISP (Franz Inc.'s Allegro CL). The basic Bayesian network program follows the algorithms given by Pearl [85] and Neapolitan [79] with extensions as described in Appendix A. The MACNET language that is used to formulate the rules in the central-system has been reimplemented following the description given by Chapman [25] where possible, with slight syntactic changes as described in [53,54]. This implementation of MACNET also draws on Agre's [1] description and the underlying execution mechanism follows that used by Terman [99]. To test this implementation of MACNET it has been successfully used as part of a reimplementation of Chapman's program BLOCKHEAD [24]. The implementation of the spatial-layout database (see Section 3.2) and associated programs are described in [54,56].

The results shown in the Tables 11–15 were all written by the HIVIS-WATCHER program which output the results directly to file in the LaTeX[18] table format.

[18] LaTeX is a document formatting language.

Table 10
The policies.

cue	*attend*	*ignore*	see Table	position in Figs. 30 and 31	see Figs.
mutual proximity	overtaking	following	11	nearside	30, 31
mutual proximity	following	overtaking	12	middle	30, 31
gross change in motion	giveway		13	farside	30, 31
mutual proximity and gross change in motion	overtaking, following and giveway		15		32, 33

8. Examples of how task-based control operates in HIVIS-WATCHER

The examples used here are drawn from the road-traffic domain. Fig. 2 illustrates this with three image frames selected from a sequence taken at a German roundabout. In this part of the sequence a number of episodic behaviours are unfolding: one vehicle leaves the roundabout; another is in an entry lane to the roundabout; also towards the rear of the image a car begins to overtake a lorry.

Here we compare the effect of using four different tasks: "look for likely overtaking behaviour", "look for likely following behaviour", "look for likely giveway behaviour" and the combination of these behaviours. Table 10 lists the policies we will be using and their corresponding results are displayed in Figs. 30–33. To explain the format of these results from HIVIS-WATCHER we will use Figs. 30–31 and Tables 11–13. In the figures the three columns, each showing a sequence of results from the same raster image frame sequence (numbers 96–228), show the effect of different task-based control policies given in Table 10. The background to each result frame is the spatial-layout of the ground-plane as described in Section 3.2. The rectangular poseboxes depict the results from the model-matcher where it has identified a scene object/vehicle however this information is only accessed when a marker (i.e., one of the six shapes ■, ◆, ●, ▼, ▲, ✚) is allocated to an object. The numbers near the centroid of each posebox gives the object's buffer-address (see Section 5.1) and the chevron (or arrow head) indicates the object's front (as determined by the model-matcher).

In frame 192 the shaded regions show the path predictions and identified conflict-area as discussed in Section 7.3 (in particular see Fig. 27). Frames 204–228 show the "activation plane"[19] for the conflict-area represented in the marker's grid (see Section 5.2.3) used during the routine "watch-for-enter-conflict-area" as described in Section 7.3. The activity of the various markers is described in the accompanying Tables 11–13. In these tables two different formats are used although all three tables

[19] Activation planes are used to keep track of interesting regions in an image, as in Ullman's [104] routine for computing containment. See also Fig. 21 and Chapman [25]. The calculation of the conflict-area uses three activation planes, one for each path-prediction and one to hold the result of their intersection.

Table 11

The results from looking for likely overtaking behaviour. The numbers in the agent columns are buffer-address indexes, and are also displayed in the figure. The "refobj deictic states" column lists the marker names for self and other, together with the deictic state found using the *the-refobj*'s field values that relates self and other.

time	Selection		Deictic representation						Task-based control					
	pairs	watch	agents			refobj deictic states			likelihood values				likely episode	ignore
			■	◆	⬟	s	o	deictic state	overtk	follow	queue	unk		
96					2									
108														
120	(1.0)	0.599												
	(3.2)	0.909												
132	(3.2)	0.977	1	2	3	◆	⬟	faster-than-me	0.579	0.027	0.232	0.162	overtaking	
						◆	⬟	rightside-of-me						
						⬟	◆	leftside-of-me						
144	(3.2)	0.987		2	3	◆	⬟	parallel-to-me	0.901	0.038	0.053	0.007	overtaking	
						◆	⬟	faster-than-me						
						◆	⬟	rightside-of-me						
						⬟	◆	similar-heading-to-me						
						⬟	◆	slower-than-me						
						⬟	◆	leftside-of-me						
156	(3.2)	0.981		2	3	◆	⬟	parallel-to-me	0.879	0.052	0.060	0.008	overtaking	
						◆	⬟	similar-speed-to-me						
						◆	⬟	rightside-of-me						
						⬟	◆	parallel-to-me						
						⬟	◆	similar-speed-to-me						
						⬟	◆	leftside-of-me						
168	(3.2)	0.977		2	3	◆	⬟	parallel-to-me	0.879	0.052	0.060	0.008	overtaking	
						◆	⬟	similar-speed-to-me						
						◆	⬟	rightside-of-me						
						⬟	◆	parallel-to-me						
						⬟	◆	similar-speed-to-me						
						⬟	◆	leftside-of-me						
						⬟	◆	behind-me						

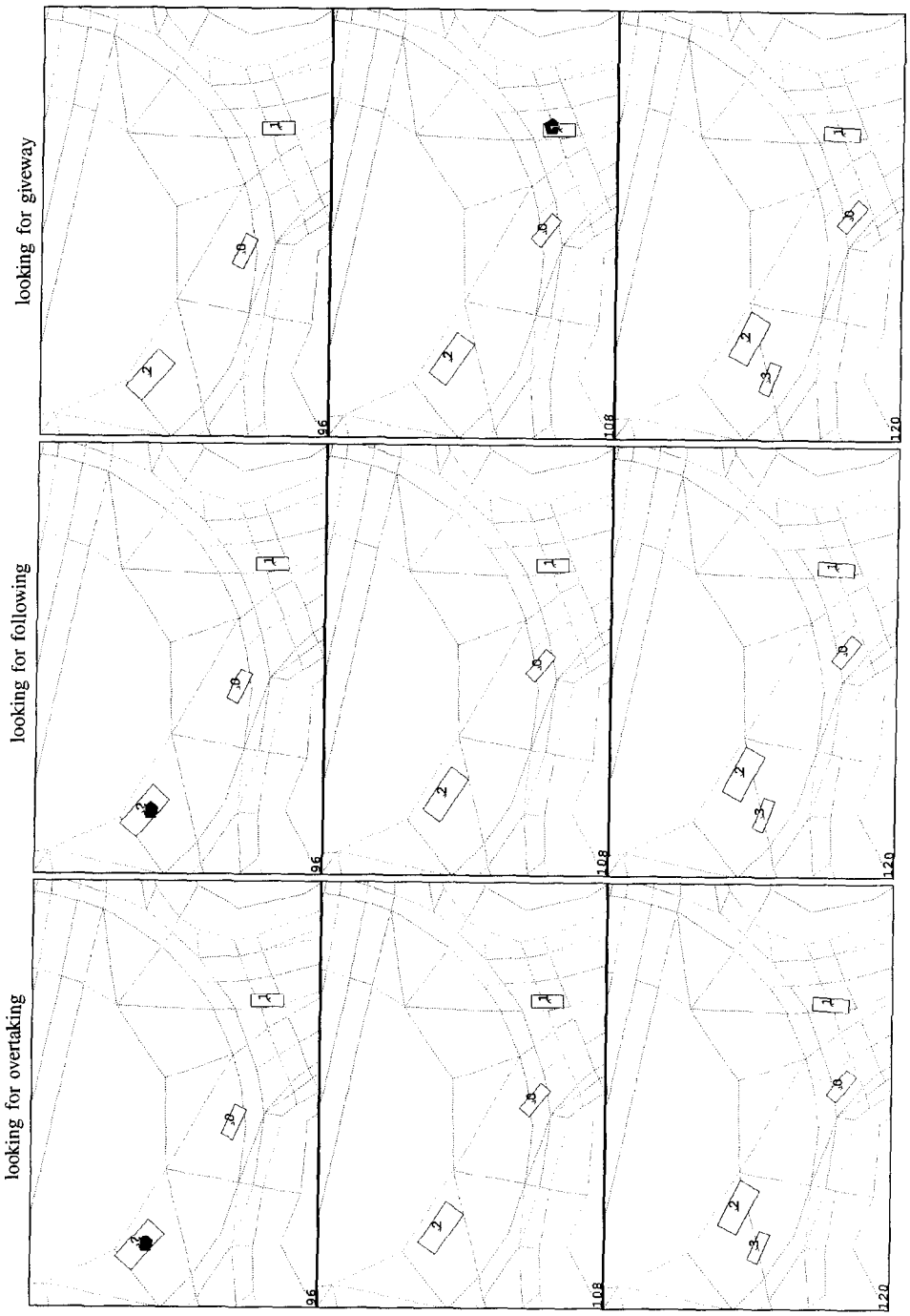
180	(3.2)	0.977	2	3	◆	⬢	parallel-to-me	0.606	0.326	0.060	0.008	overtaking
					◆	⬢	faster-than-me					
					◆	⬢	infront-of-me					
					⬢	◆	parallel-to-me					
					⬢	◆	slower-than-me					
					⬢	◆	leftside-of-me					
					⬢	◆	behind-me					
192	(3.2)	0.977	2	3	◆	⬢	parallel-to-me	0.532	0.399	0.061	0.008	overtaking
					◆	⬢	similar-speed-to-me					
					◆	⬢	infront-of-me					
					⬢	◆	parallel-to-me					
					⬢	◆	similar-speed-to-me					
					⬢	◆	leftside-of-me					
					⬢	◆	behind-me					
204	(3.2)	0.977	2	3	◆	⬢	parallel-to-me	0.606	0.326	0.060	0.008	overtaking
					◆	⬢	faster-than-me					
					◆	⬢	infront-of-me					
					⬢	◆	parallel-to-me					
					⬢	◆	slower-than-me					
					⬢	◆	leftside-of-me					
					⬢	◆	behind-me					
216	(3.2)	0.967	2	3	◆	⬢	parallel-to-me	0.606	0.326	0.060	0.008	overtaking
					◆	⬢	faster-than-me					
					◆	⬢	infront-of-me					
					⬢	◆	parallel-to-me					
					⬢	◆	slower-than-me					
					⬢	◆	leftside-of-me					
					⬢	◆	behind-me					
228	(3.2)	0.961	2	3	◆	⬢	similar-heading-to-me	0.606	0.326	0.060	0.008	overtaking
					◆	⬢	faster-than-me					
					◆	⬢	infront-of-me					
					⬢	◆	similar-heading-to-me					
					⬢	◆	slower-than-me					
					⬢	◆	leftside-of-me					
					⬢	◆	behind-me					

Table 12
The results from looking for likely following behaviour.

time	Selection		Deictic representation						Task-based control					
	pairs	watch	agents			refobj deictic states			likelihood values				likely episode	ignore
			■	◆	⬟	s	o	deictic state	overtk	follow	queue	unk		
96					2									
108														
120	(1.0)	0.599												
	(3.2)	0.909												
132	(3.2)	0.977	1	2	3	◆	⬟	rightside-of-me	0.579	0.027	0.232	0.162	overtaking	t
						⬟	◆	leftside-of-me						
144	(3.2)	0.785												
156	(3.2)	0.960		3	2	⬟	◆	parallel-to-me	0.879	0.052	0.060	0.008	overtaking	t
						⬟	◆	similar-speed-to-me						
						⬟	◆	rightside-of-me						
						◆	⬟	parallel-to-me						
						◆	⬟	similar-speed-to-me						
						◆	⬟	leftside-of-me						
168	(3.2)	0.676												
180	(3.2)	0.938		2	3	◆	⬟	parallel-to-me	0.606	0.326	0.060	0.008	overtaking	t
						◆	⬟	faster-than-me						
						◆	⬟	infront-of-me						
						⬟	◆	parallel-to-me						
						⬟	◆	slower-than-me						
						⬟	◆	leftside-of-me						
						⬟	◆	behind-me						

Table 12 — continued

time	Selection		Deictic representation						Task-based control					
	pairs	watch	agents			refobj deictic states			likelihood values				likely episode	ignore
			■	◆	⬟	s	o	deictic state	overtk	follow	queue	unk		
192	(3.2)	0.676												
204	(3.2)	0.938		3	2	⬟	◆	parallel-to-me	0.606	0.326	0.060	0.008	overtaking	t
						⬟	◆	faster-than-me						
						⬟	◆	infront-of-me						
						◆	⬟	parallel-to-me						
						◆	⬟	slower-than-me						
						◆	⬟	leftside-of-me						
						◆	⬟	behind-me						
216	(3.2)	0.585												
228	(3.2)	0.871		2	3	◆	⬟	parallel-to-me	0.606	0.326	0.060	0.008	overtaking	t
						◆	⬟	faster-than-me						
						◆	⬟	infront-of-me						
						⬟	◆	parallel-to-me						
						⬟	◆	slower-than-me						
						⬟	◆	leftside-of-me						
						⬟	◆	behind-me						


looking for giveway
looking for following
looking for overtaking
96
108
120

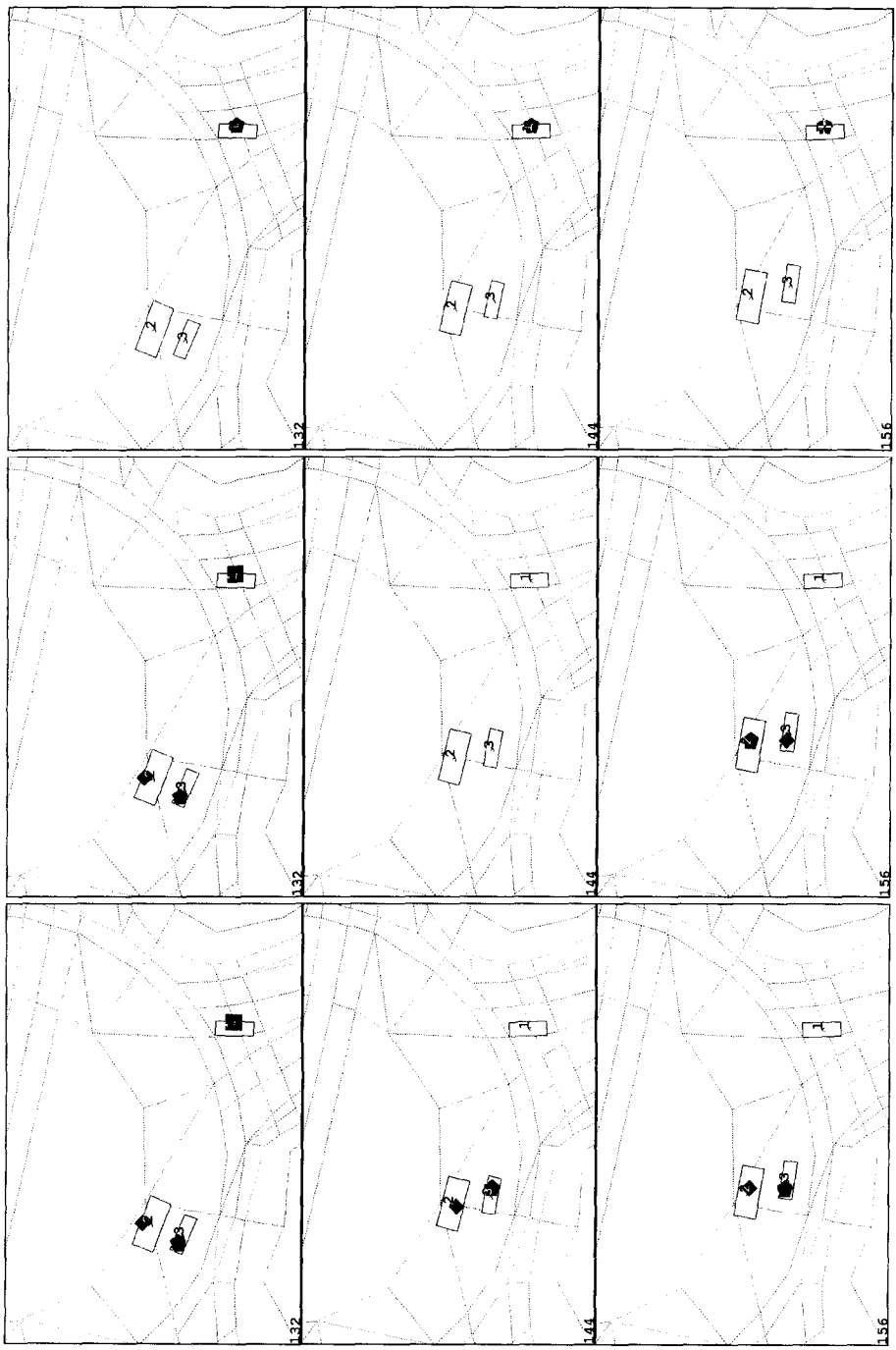
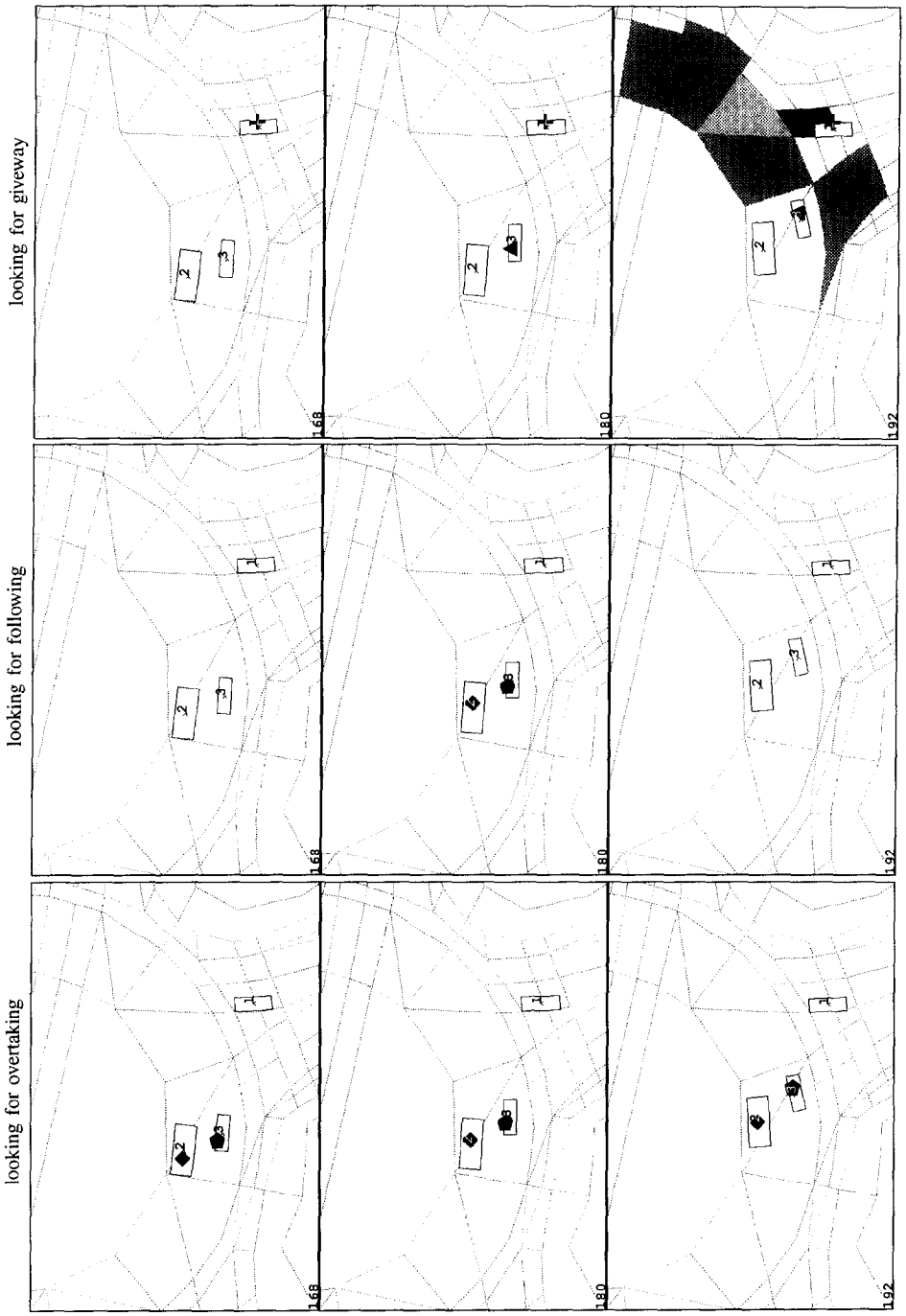


Fig. 30. Part A1.


looking for overtaking
looking for following
looking for giveway
168
180
192

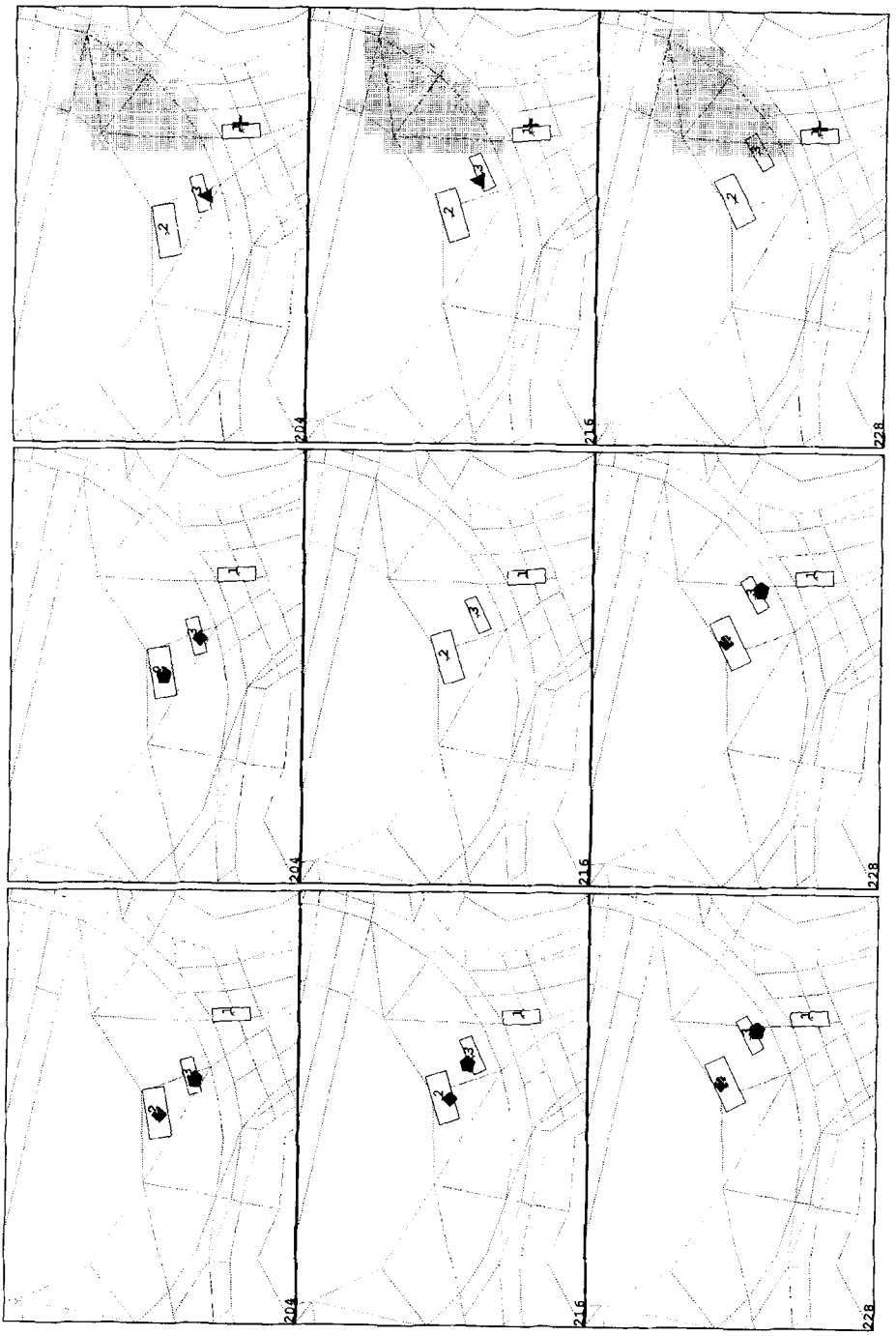


Fig. 31. Part A2.

Table 13
The kernel results from looking for the giveway episode.

time	agents			kernel			events and state changes
	■	◆	⬢	▼	▲	✚	
96							
108			1				
120							
132			1				
144			1				
156			1			1	
168						1	(:EVENT-GW1)
180					3	1	(:STATE-CHANGE-GW2)
192					3	1	(:STATE-CHANGE-GW3)
204					3	1	
216					3	1	
228						1	(:STATE-CHANGE-GW4)

cover the same sequence as denoted by the camera footage time count used to number the raster image frames.

The format used in Tables 11 and 12 shows results from the agency. The "pairs" column shows the results from the preattentive cue for mutual-proximity as indicated by each LISP dot paired cons cell. The object numbers are buffer-addresses (see Section 5.1). The "watch" column gives the probability measure of how close the two objects in the pairing are as calculated by the DDN. This uses the result from the mutual-proximity preattentive cue, so the higher the value the closer together the objects are and the more interesting the pairing is. The "agent" column says which attentional markers are allocated to which objects. The three shapes ■, ◆ and ⬢ denote the three agent markers as described in Table 3. The numbers in the column for each agent marker says which object (i.e., buffer-address) the marker is "on" (i.e., allocated to). Once the marker is allocated, HIVIS-WATCHER can then access the results from the model-matcher and obtain the object's pose position. This information enables the deictic states for each *the-refobj* to be calculated and presented under the subheadings "s" (self), "o" (other) and "deictic state". The marker shapes are used under the headings s and o because it is via marker allocation that the positional deictic state has been calculated. The column "likelihood values" shows the distribution of the probability values that the episode being observed is an instance of overtaking, following, queueing or some unknown (to HIVIS-WATCHER) behaviour. The "likely episode" column presents the most likely explanation. The "ignore" column says whether this pairing should be ignored so that HIVIS-WATCHER can check to see if there is anything else that is more interesting to watch in accordance to its current policy.

In Table 13 the time and agents column is as for Tables 11 and 12. The next column, titled "kernel", shows the allocation of the kernel markers ▼, ▲ and ✚ which are as described in Table 3. This part of the table describes the behaviour of the kernel. The numbers again denote which object (buffer-address) the marker is allocated to. The "events and state changes" relate to those described in Section 7.3.

8.1. Overtaking

The purpose of this example is to show that the DDN and TASKNETs can pick out a pair of vehicles that are performing an overtaking episode. To do this we use the TASKNET policy "attend to likely overtaking and ignore likely following". The results from the frame sequence are also shown in Table 11 where the selected objects are denoted by the entries in the agent columns. The missing entries are because one of the vehicles occludes the other from the camera during frames 96 and 108, so no mutually proximate objects are visible to the observer. Fig. 2 shows the camera's field-of-view which affects the contents of the frame updates because we are dependent upon what is visible from the camera position not what is visible from the overhead view. By frame 132 overtaking is positively identified.

8.2. Following

Alongside these overtaking pictures are those that illustrate the difference in agent allocation when we change the TASKNET policy to "attend likely following and ignore likely overtaking". Comparing Table 11 with Table 12 shows the effect changing TASKNET's policy has on the interpretation of the same data. When the ignore boolean, in Table 12, is true, HIVIS-WATCHER terminates-attention (in accordance to the allocation theory given in Section 6.2.1) re-setting the current watch value (for the attended object in the DDN) to 0.0, which causes the marker to be removed.

8.3. Giveway

To illustrate the need for local and global viewpoints we use the policy "look for likely giveway behaviour". The results from the giveway implementation are given in Table 13 and in the sequence next to those for overtaking and following. In the table HIVIS-WATCHER only uses three attentional markers to perform the giveway detection routine, and the events listed in this table correspond to the five routines described in Section 7.3. In Figs. 30–31 the frames 108, 132–156 describe the allocation of `*agent2*` cued by gross-change-in-motion. At frame 120 the vehicle moved again, before the motion-prior was altered from `moving` to `stationary` by the agency operator `change-motion-prior!` which changes the motion-prior of the preattentive cue for the

Table 14
The problems.

Problem	See Section(s)
surveillance	1–10
spatial arrangements	7.2.1, 7.2
computational load	6.2
amount of evidence	6.2
viewpoint integration	6.2, 6.2.2, 6.3, 7.2
temporal history limitation	6.3, 7.3

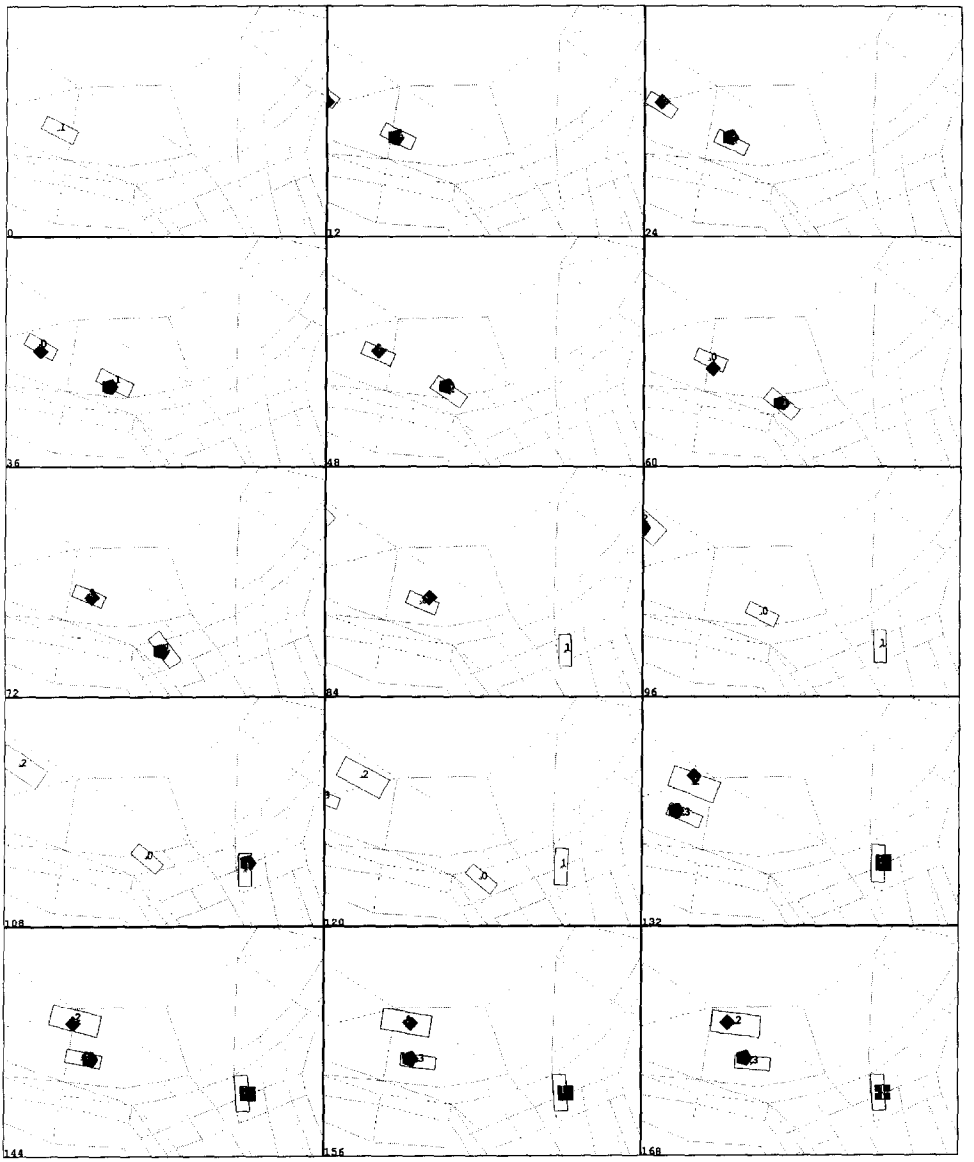


Fig. 32. Part B1.

attended object (see Section 7.1.2). The value of motion-prior is changed by frame 168 because the object ceases to have an interesting motion property (i.e., the agent marker has been removed). Frame 192 shows the results from the region path predictions that generate the contents of the kernel activation planes. Frames 204–258 display the activation plane for $\mathcal{CA}$. Frame 228 shows the removal of `*head-marker*` following a successful intersection with $\mathcal{CA}$.

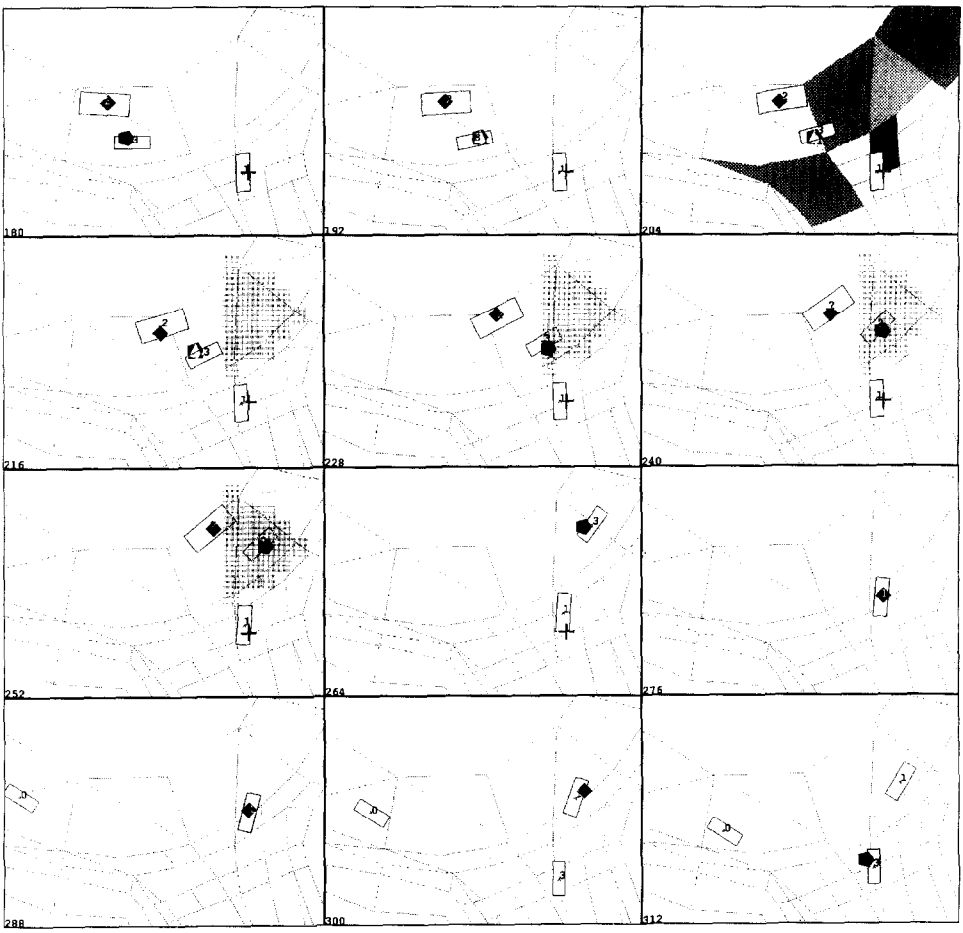


Fig. 33. Part B2.

8.4. Assessment

This short example has illustrated how a given surveillance task can affect interpretation and that uncertainty and ambiguity of interpretation is present even after scene objects have been identified. The task-based control developed here has enabled HIVIS-WATCHER to obtain a viable task-dependent description of what is happening in the scene. Task-based control is an important element of the *surveillance problem* and its related sub-problems, which are listed in Table 14. The solutions developed in this paper have contributed towards the situated approach described here, endowing HIVIS-WATCHER with timely response to each surveillance task.

The first two examples demonstrate the task-based control model described in Section 6.2. Overtaking and following are similar behaviours, and in the implementation

Table 15
The agency and kernel results from looking for likely overtaking, following and giveway behaviour. The □ indicates no currently available agent reference.

time	pairs	watch	agents			refobj deictic states			likelihood values				likely episode	ignore	kernel	events and state changes
			■	◆	⬟	s	o	deictic state	overtk	follow	queue	unk			▼ ▲ ✚	
0	(1,0)	0.599														
12	(1,0)	0.719		0	1											
24	(1,0)	0.779		0	1											
36	(1,0)	0.807		0	1	⬟	◆	behind-me	0.309	0.076	0.155	0.460	unknown			
	(3,2)	0.599														
48	(1,0)	0.820	3	0	1	◆	⬟	infront-of-me	0.124	0.428	0.264	0.184	following			
						⬟	◆	behind-me								
60	(1,0)	0.868		0	1	◆	⬟	infront-of-me	0.220	0.701	0.069	0.009	following			
						⬟	◆	behind-me								
72	(1,0)	0.888		0	1	◆	⬟	infront-of-me	0.220	0.701	0.069	0.009	following			
						⬟	◆	behind-me								
84	(3,2)	0.599		0												
96					2											
108					1											
120	(1,0)	0.599														
	(3,2)	0.909														
132	(3,2)	0.977	1	2	3	◆	⬟	rightside-of-me	0.579	0.027	0.232	0.162	overtaking			
						⬟	◆	leftside-of-me								
144	(3,2)	0.987	1	2	3	◆	⬟	rightside-of-me	0.901	0.038	0.053	0.007	overtaking			
						⬟	◆	leftside-of-me								
156	(3,2)	0.981	1	2	3	◆	⬟	rightside-of-me	0.879	0.052	0.060	0.008	overtaking			
						⬟	◆	leftside-of-me								
168	(3,2)	0.977	1	2	3	◆	⬟	rightside-of-me	0.879	0.052	0.060	0.008	overtaking			
						⬟	◆	leftside-of-me								
						⬟	◆	behind-me								

Table 15 — continued

time	pairs	watch	agents			refobj deictic states			likelihood values				likely episode	ignore	kernel			events and state changes
			■	◆	⬟	s	o	deictic state	overtk	follow	queue	unk			▼	▲	✚	
180	(3.2)	0.977		2	3	◆	⬟	infront-of-me	0.606	0.326	0.060	0.008	overtaking				1	(:EVENT-GW1)
						⬟	◆	leftside-of-me										
						⬟	◆	behind-me										
192	(3.2)	0.977		2	3	◆	⬟	infront-of-me	0.532	0.399	0.061	0.008	overtaking			3	1	(:STATE-CHANGE-GW2)
						⬟	◆	leftside-of-me										
						⬟	◆	behind-me										
204	(3.2)	0.977		2	3	◆	⬟	infront-of-me	0.606	0.326	0.060	0.008	overtaking			3	1	(:STATE-CHANGE-GW3)
						⬟	◆	leftside-of-me										
						⬟	◆	behind-me										
216	(3.2)	0.967		2	3	◆	⬟	infront-of-me	0.606	0.326	0.060	0.008	overtaking			3	1	
						⬟	◆	leftside-of-me										
						⬟	◆	behind-me										
228	(3.2)	0.961		2	3	◆	⬟	infront-of-me	0.606	0.326	0.060	0.008	overtaking				1	(:STATE-CHANGE-GW4)
						⬟	◆	leftside-of-me										
						⬟	◆	behind-me										
240	(3.2)	0.961		2	3	◆	⬟	infront-of-me	0.606	0.326	0.060	0.008	overtaking				1	
						⬟	◆	leftside-of-me										
						⬟	◆	behind-me										
252	(3.2)	0.961		2	3	◆	⬟	infront-of-me	0.532	0.399	0.061	0.008	overtaking					
						⬟	◆	leftside-of-me										
						⬟	◆	behind-me										
264	(3.1)	0.676			3	⬟	□	behind-me										(:EVENT-GW5)
276				1														
288				1														
300	(3.1)	0.599		1														
	(4.2)	0.599																
312	(3.1)	0.719	2	4	3	⬟	□	infront-of-me										
	(4.2)	0.719				◆	■	infront-of-me										

described here, both are identified by the same preattentive cue. In [54] the kernel and agency are further integrated to enable overtaking to be distinguished from similar behaviour such as when a car pulls out to overtake but then drops back. Thus enabling the identification of likely overtaking by the TASKNET to be verified.

The third example shows that HIVIS-WATCHER can identify likely giveway behaviour. We have described how the gross-change-in-motion preattentive operator can be used as the foundation for identifying the first and last routines in the temporal history of a giveway episode, with the three central routines of the giveway episode demonstrating how the kernel is able to coordinate the collection of information about different scene objects. This is in contrast to the local reasoning of the typical-object-model. This example has shown how the local-form and global-form can be combined to provide a task-dependent interpretation of the image sequence. Some refinements to this approach are given in [54].

8.4.1. Doing visual tasks in parallel

It is possible for HIVIS-WATCHER to "look for both giveway and overtaking and/or following behaviour". This is because the preattentive operators for mutual-proximity and gross-change-in-motion are independent as described in Appendices B and C. Figs. 32 and 33 and Table 15 show the results from doing all three visual tasks of looking for following, overtaking and giveway behaviour in parallel. To make the table shorter we just show the positional deictic states. It is interesting to see how these visual tasks interact causing the detection of giveway behaviour to be slightly delayed, demonstrating the flexibility of the kernel routines. The format used in Table 15 is a combination of that used in Tables 11–13.

8.4.2. General applicability of this approach

The preattentive cues used in the examples here are to illustrate that the theory described in Section 6 can be implemented in a working computer program. Other preattentive cues may be appropriate in different applications where different attention-calling responses are needed. It might also be useful to combine preattentive cues, for example mutual-proximity and gross-change-in-motion could be used in an indoor office domain to select likely instances where two people are talking face-to-face, i.e., they stop walking, stand still and have a chat. In [54] another Gestalt grouping primitive, similarity, is used to identify clusters of objects travelling in the same direction. The members of these clusters also form a network of mutual-proximity relationships which might coalesce into a queue if there is any congestion (such as waiting for a tram to pass through).

8.4.3. Performance

One of the benefits of task-based control is that the processing done is not dependent on the number of objects in the sequence, but is instead dependent on the given visual task. This has been demonstrated in the examples shown in Figs. 30–33 where marker application is dependent upon the different surveillance tasks, and the relationships and actions of the scene objects, but not on the number of scene objects. The graph in

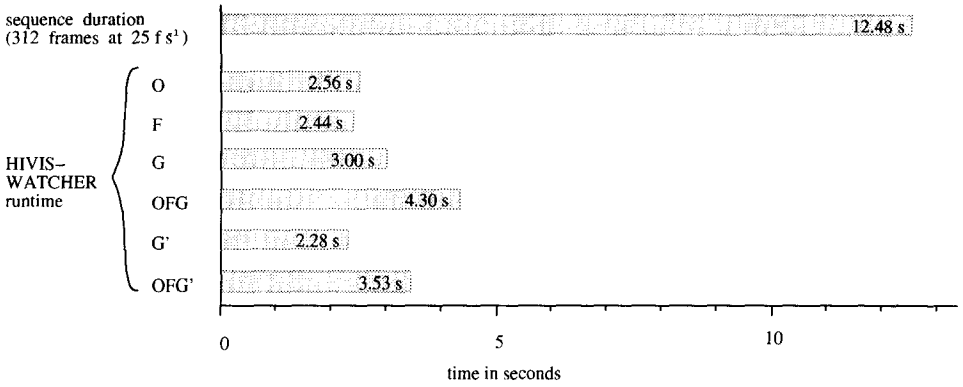


Fig. 34. Performance timings for the different official-observer behaviours, showing that computation is faster than real-time.

Table 16
Which percentage of runtime is spent doing what.

Function	% of total time					
	O	F	G	OFG	G′	OFG′
Top-level-loop	100.0	100.0	100.0	100.0	100.0	100.0
Update buffer	21.1	21.1	24.6	17.8	33.3	19.4
Run agency peripheral-system	34.3	33.0	14.2	14.5	18.0	19.4
Preattentive	16.0	14.4	13.4	6.5	15.3	11.0
Attentive	17.8	18.2	0.8	7.5	2.6	8.4
Run kernel peripheral-system			14.2	10.3	18.5	12.3
Allocate agents	31.0	33.0	42.5	45.0	22.8	32.7
Build DDN	14.1	16.7	26.5	25.1	11.1	17.5
Run DDN	11.7	12.9	9.7	14.5	5.8	12.6
Run each agents typical-object-model	8.0	8.6	<1.0	4.9	2.1	7.8
Collect results	4.2	3.8		3.6		4.2
Run kernel routines			3.4	3.6	4.8	2.9
Run effector-system	0.9	0.5	<1.0	<1.0	<1.0	<1.0

Fig. 34 shows the different durations that HIVIS-WATCHER [20] took to process the given sequence for the visual tasks of individually looking for overtaking (O) or following (F) or giveway (G) behaviour and of looking for all three at once (OFG). The figure

[20] The implementation used here was written in Allegro Common Lisp version 4.2 and running on a Sparc 20. The timings all used the "real-time" value returned from the LISP function `time` and were averaged from seven runs.

Table 17
The number of model-matches. The numbers in this table are from the twenty seven frames used here not the full 312 frame sequence, but they give an idea of the proportions involved.

Input data and Attentional behaviours	number of model-matches
compact encoding stream	106
O	38
F	31
G	20
OFG	57

also shows that the processing performed by HIVIS-WATCHER is less than the real-time duration of the image sequence used here. Although it should be noted that we only process every twelfth frame, i.e., there are twenty-nine frames in this instance, spaced at roughly half second intervals. The results in Fig. 34 and Table 16 show the various costs of the preattentive cue for gross-change-in-motion (GCIM) and mutual-proximity (MP). In the general case the preattentive operator is applied over the whole scene, where a GCIM network is created for each scene object, and a MP network is only created for those object pairs that have a mutual-proximity relationship. The results are shown by the values for O, F, G and OFG. However, by limiting the use of GCIM networks to objects that are in giveway/turning regions (see Fig. 4 the GCIM overhead can be greatly reduced as shown by the values for G′ and OFG′. Illustrating the benefits of using knowledge about the scene to guide preattentive processing (i.e., knowing where to look).

As shown in Table 17 the differences would be more marked if the timings included the cost of model-matching. In the table a count of the number of model-matches made for the frames shown in the 312 frame sequence is given together with those identified as being necessary by the attachment of an attentional marker for the various behaviours used in this paper. The values in this table illustrate how a given surveillance task can be fulfilled with minimal resources.

9. Related work

We have already covered some related work in Section 5, particularly the use of deictic representation by Agre and Chapman and related issues from Ballard's [6] description of active vision. We also provided the background to the peripheral/central split that provides the foundation for the situated approach developed here. Although there is a bias towards Agre and Chapman's description of the central-system here, there are a number of alternative approaches such as Horswill's Prolog-like mechanism [52], Nilsson's Teleo-Reactive condition-action rules [83], Mackworth's constraint-based approach [70], and Rosenschein and Kaelbling's situated automata [90], for example.

This separation of input- and central-system and the feedback provided by the tight coupling (i.e., runtime execution cycles or oscillates) between these two systems allows us to address control without using traditional planning or plan recognition (see Allen et al. [4] for example). Although a common idea in control theory, feedback is not often used in AI systems. Nilsson [83] has pointed out that this may in part be due to the sequential nature of program execution, but it could also be related to how we perform temporal reasoning or planning. When we step back and consider a temporal history, its past, present and future, from the side (as described by McDermott [74]) the idea of feedback seems incongruous, however, when we perform or act upon a plan, our real-world "execution" is in continuous feedback with the world.

The temporal representation we employ uses a deictic approach, uncommon in AI. Temporal representations tend to use a time line and "reason from the side" (see McDermott [74]) because they are reasoning about symbolic time points (e.g., dates) which is useful for predicting events (see Hanks and McDermott [47]) and building time-maps (see Dean and McDermott [32]). We do not use such temporal reasoning here and have instead adopted a simple model of time.

The application of Bayesian networks used in HIVIS-WATCHER to model temporal properties is novel and, although we only use simple graphs they make clear the general approach. Related work includes the dynamic construction of Bayesian networks (see Breese [15], Charniak and Goldman [26]), but the result is a single static network, rather like a single iteration of the NEI algorithm described in Appendix A. Other researchers have investigated domains where the world changes and the focus is reasoning over time. Sometimes a complete model of the previous time state or time-slice might not be needed making it possible to simplify the relevant history to a single node, Sucar and Gillies [95] call this "semi-static recognition". They also describe the use of relationship nodes that sit on the boundary between time-slices holding a value that describes how each property changes between successive time-slices. They call this "dynamic recognition" because the temporal results are collected into a single root node for each recognised object.

A few researchers (see Dagum et al. [30], Dean et al. [31], Kiaerulff [62], Nicholson and Brady [82]) have investigated dynamic networks that have a *repeated* structure, where, as the network grows over time, the state of each domain variable at different times is represented by a series of nodes that has a limit on the history maintained (i.e., a window of time-slices). Like the DDN developed here, they obey the Markov property that the future is conditionally independent of the past given the present.

HIVIS-WATCHER is more an engineering proof that task-based control is feasible than being an implementation based on known psychophysical evidence. Chapman [25] makes both cases for his implementation of SONJA, and most of the experimental data he references can also be used to support the work described here. Other psychophysical experiments are described by Allport [5] and van der Heijden [106] some of which provide general support for task-based control. For instance Allport [5] suggests that selection is based on goal-directed action, providing a task/action-specific form of focusing. And van der Heijden [106] emphasizes that selection is performed in close cooperation and interaction with expectations and intentions, and is controlled by both the subject and the world. Some researchers are more concerned with biologically plausi-

ble approaches and have developed connectionist models. For example, van der Heijden describes "SLAM" his SeLective Attention Model. Also see Baluja and Pomerleau [9] and Mozer [76]. Tsotsos et al. [103, p. 537] suggest that task requirements and some kind of internal spatial working memory of what has been seen may play a role in determining whether to re-attend locations when subsequent eye movement brings previously attended objects back into view. This identified requirement seems similar to the description of task-based control given in Section 6 and the global-form grid described in Sections 4 and 5.2.3. Van der Heijden also identifies that selective-attention provides temporal order or temporal structure in a spatially structured visual world. While this probably has more to do with marker manipulation, there may be a correspondence between the functionality provided by the DDN as part of preattentive selection and van der Heijden's description of the time course of activation for mapping from location to identity which seems to where the Gestalt process is most likely to take place.

10. Conclusion

HIVIS-WATCHER represents an advance over more traditional AI approaches (such as HIVIS-MONITOR) due to its greater expressive power and its ability to limit its inference according to the demands of the task at hand. We have shown that interpretation of a scene has to take into account the dynamic nature of the world and also the official-observer's knowledge about what typically happens in the world. Doing so involves integrating a theory of task-based control (see Section 6) with a theory of representing typical object behaviour (see Section 5.3.1). The main concept behind task-based control is the preattentive/attentive split or sequential processing of attentive reasoning. The preattentive (simple) operators and attentive (complex) operators are illustrative only. In the implementation developed here, we use a dynamic form of Bayesian network which has been used to model the spatio-temporal evolution of preattentive information, and a static level of control in the form of both an evidence gathering Bayesian network and a collection of rules that describe routine visual behaviour of the official-observer. Together these promise a highly effective attentional control mechanism which can be distributed under the agent formalism to deliver real-time performance.

Other benefits of HIVIS-WATCHER over HIVIS-MONITOR are due to its task orientedness reducing runtime representation, reasoning and complexity. In HIVIS-WATCHER: (1) the deictic representation has simplified the computational model of behaviour, (2) the situated approach has taken into account both the evolving context of the dynamic scene objects and also the task-oriented observer's context, (3) the use of selective-attention provides a more viable form of real-time processing.

Other key points of this paper concern:

- The distinction between script-based and more situated approaches.
- The separation and integration of global and local reasoning in the context of a single official-observer, together with the illustration of how both play complementary roles in developing different levels of understanding.
- The propagation of reasoning in the "here-and-now" through to the control mechanism in order to reflect the reactive quality of dynamic object behaviour.

And although the research in this paper has been illustrated by using data from a road-traffic surveillance application, the intention is that the general framework should be applicable to other application domains.

Current work is addressing two important issues. The first concerns removing the strong interface between the PC and SAC/HIVIS, illustrated in Fig. 1, and making them more tightly coupled. The result of this will enable the HIVIS task-based control component to directly influence the visual data-collection process, so that the attentive process of working out what an object is (e.g., model-matching), is only carried out when a marker is applied to what the preattentive cue identifies as a potentially task relevant interesting object. The current version of HIVIS-WATCHER can be considered as a prototype of this more complete attentional computer-vision system.

The second concerns learning the behavioural information, removing the hand coded element of choosing preattentive and attentive cues in HIVIS-WATCHER.

Acknowledgements

I would like to thank Hilary Buxton, Mike Clarke and Dave Saunders for helpful discussions and guidance that contributed towards the work described in this paper. I also thank Mike Brady, Duncan Gillies and the anonymous reviewers.

This work has been funded by SERC under a CASE award with the GEC Marconi Research Centre and by the EPSRC project "Behavioural Analysis for Visual Surveillance using Bayesian networks" (grant GR/K08772).

I thank Annette Herskovits for giving me permission to base Fig. 13 on Figs. 10.2 and 10.3 from pp. 158–159 of her book [50].

I also thank Margaret Fleck for giving me permission to base Fig. 6 on Fig. 11 from p. 83 of [35] and on Fig. 16 from p. 21 of [37], copyright 1996, with kind permission of Elsevier Science, Sara Burgerhartstraat 25, 1055 KV Amsterdam, The Netherlands.

Parts of the work described here have appeared in earlier forms in the conference papers [58] and [59] and the survey paper [55].

The in-line illustration in Section 7.1.1 is reprinted from [58], Fig. 23 is based on Fig. 1 of [58], Fig. 24 is reprinted from Fig. 2 of [58], Table 4 includes the entries from Table 1 of [58], part of Table 5 is from Table 2 of [58], and part of Table 7 is from Table 3 of [58]. These are from the Proceedings of the Thirteenth International Conference on Artificial Intelligence, copyright International Joint Conferences on Artificial Intelligence, Inc., 1993. Reprinted with permission. Morgan Kaufmann (http://www.mkp.com).

Fig. 2 is reprinted from Fig. 1 of [59], Table 1 is based on Table 1 of [59], and some of the result frames used in Figs. 30–31 are reprinted from Figs. 2, 3 and 4 of [59]. These are copyright 1996 by Springer-Verlag and are used with permission from Springer-Verlag, Heidelberg, Germany.

Fig. 12 is reprinted from Fig. 3 of [55], Fig. 22 is based on Fig. 9 of [55], and Fig. 29 is based on Fig. 8 of [55]. These are used with kind permission from Kluwer Academic Publishers.

(1) run `update-graph-structure`
(2) update prior values of root nodes and supply evidence node values
(3) run `inference-algorithm` to update beliefs

Fig. A.1. The Network Expansion and Inference (NEI) algorithm.

Appendix A. A probabilistic dynamic graph representation

The Dynamic Decision Network (DDN) used by ALLOCATE to identify the most prominent preattentive features consists of a dynamic graph structure that is updated at each clock tick to mirror the information contained in the new image frame. To do this we run the Network Expansion and Inference (NEI) algorithm given in Fig. A.1. It cannot model continuous time because the process of graph extension in necessarily discrete. If the graph structure is constant over time a better solution would be the more usual static graph approach (as described by Pearl [85]), because using a DDN incurs the additional overheads of the `update-graph-structure` step. We can distinguish between the different form of Bayesian network in two distinct ways: topological structure (either continuous or changing), and graph structure (one of: causal tree, causal polytree or multiply connected). The Bayesian network described here is a topologically changing causal polytree.

A.1. Connection rules

To express manipulations to the DDN graph structure we develop a language called "MACDDN"[21] which uses construction rules to constrain the set of possible graph structures. These connection rules are very local, use a deictic model of time (i.e., *now*, *next*, *previous*), and are needed to ensure that loops are not formed within the graph which would invalidate the use of a causal polytree inference algorithm. The set of connection rules determine the validity of each edge in the graph and consist of two forms "now-connections" (NOWC) and "one-step-temporal-connections" (OSTC). The NOWCs link nodes that share the current time value, and OSTCs link nodes with adjacent time values.

Each node is defined by a tuple $\langle \mathtt{n}, \mathtt{o}, \mathtt{a}, \mathtt{b} \rangle$ where `n` is a node type symbol, `o` is an owner symbol, `a` is an attribute symbol, and `b` is a belief symbol. Although time is important, it is not expressed explicitly as part of each node, instead it is implicit in the node's position within the graph. Time is modelled as subgraphs, called "buckets", that hold all the nodes for a particular time-cell. Fig. A.2 illustrates this idea, with OSTCs defining the links that can be made between nodes in adjacent buckets.

[21] MACDDN stands for MACro DDN because the rule language has been implemented as a collection of LISP macros (giving a macrology) that specify how to construct a DDN.

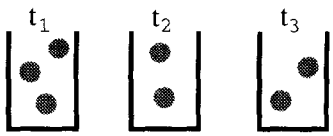


Fig. A.2. The time-cell as a bucket holding a set of nodes.

For graph construction we use a more compact notation that rewrites $\langle \mathtt{n}, \mathtt{o}, \mathtt{a}, \mathtt{b} \rangle$ as $\mathtt{n}_{\mathtt{o}}^{\mathtt{a}}$ in the case where as attribute is defined, and $\mathtt{n}_{\mathtt{o}}^{\clubsuit}$ if not, where ♣ acts as a wild card. This notation hides the belief value because it is not used during graph construction, and makes explicit the type, owner and attribute symbols that the construction rules operate upon. This part of the language is used to express the vertices present in the DDN, using the syntax: NT is the set of node type symbols, OV1 is the set of owner symbols, OV2 is the powerset of OV1, VT is the set of node tuples, and OV1♣ is the set of owner symbols extended by the wild card, i.e., OV1♣ = OV1 ∪ {♣}. In each node tuple $\mathtt{n}_{\mathtt{o}}^{\mathtt{a}} \in \mathtt{VT}$, we have $\mathtt{n} \in \mathtt{NT}$, $\mathtt{o} \in \mathtt{OV2}$, and $\mathtt{a} \in \mathtt{OV1}\clubsuit$.

We use three predicate tests: nowc(p) says whether the node p is present in the bucket for $\mathtt{t}_{\mathtt{now}}$; nowc(p ∧ q) says whether both nodes p and q are present in the bucket for $\mathtt{t}_{\mathtt{now}}$; and ostc(p, q) says whether the nodes p and q are in the consecutive buckets under consideration. Buckets are held in an indexable data structure, and at the end of each system-cycle, the storage is "moved" (by pointer reallocation) so that what was held at $\mathtt{t}_{\mathtt{i}}$ is now held at $\mathtt{t}_{\mathtt{i-1}}$. A more concrete semantics for these predicates is given in [54] however, this has given the functionality that we use to perform tests on the node tuples held in the bucket structure.

A.2. Construction algorithm

The MACDDN language uses the following notation.[22] To add an edge we use the infix operator ➜ called "add directed edge", that is a function (➜) of type VT → VT → DDNG → DDNG, that updates the DDN graph of type DDNG. To make the notation less cluttered we will hide the DDNG parameter so that we only need to specify first two arguments of (➜). For example, we will write p➜q to add an edge between the node tuples p and q where p, q ∈ VT.

In addition to adding edges, the MACDDN language also needs functionality for adding a new node. This is done using two functions: make-node(NODE-TYPE, NAME) which creates a node tuple with a wild card attribute (i.e., make-node is a function of type NT → OV2 → VT); and add-vertex(DDNG, NODE) which adds a node tuple (for now we will hide the DDNG parameter and define it as add-vertex(p), where p ∈ VT).

To describe the constructor macros used in the MACDDN language we will use templates that are written using the notation introduced above, together with additional

[22] An introductory explanation of the type notation used here (including "currying") to describe the MACDDN functionality is given by Bird and Wadler [12, pp. 8–12].

syntax for tests (**if** *antecedent* **do** *consequent* **od**) and value assignment (denoted by the := symbol).

- `now-ref(p)` =
  if `nowc(p)` **do** `o := node-owner(p);`
  `r := make-node(R,o); add-vertex(r); p➜r` **od**
  this adds a result node to the graph so that the output from the belief slot can be translated to a more useful form.
- `now-relationship(p,q)` =
  if `nowc(p ∧ q)` **do**
  `o := {node-owner(p), node-owner(q)};`
  `r := make-node(R,o);`
  `add-vertex(r); p➜r; q➜r` **od**
  this combines the results from two nodes denoting some relationship between them, and is used as part of the solution to the *spatial arrangements problem.*
- `temporal-change(p, q)` = **if** `ostc(p,q)` **do** `p➜q` **od**
  this temporally links nodes that are of different types, but the same owner.
- `temporal-continuity(p)` = **if** `ostc(p,p)` **do** `p➜p` **od**
  this temporally links nodes that are the same in terms of the type, owner and attribute slots.

The construction rules express the essential details but do not describe how the connections are made, for this we first need a description of the DDN graph structure in terms of temporally separated buckets. Each bucket has a set of vertex-indexes V and edges that are denoted by the set of edge-destinations ED and the set of edge-sources ES. ED and ES are subsets of the set of graph indexes GI, such that each graph index is defined so that in addition to referring to the vertex-index they also refer to the relative time, e.g., if $g \in \mathtt{GI}$, $h \in \mathtt{H}$, $v \in \mathtt{V}$ then $g = (h, v)$ where h is a deictic-time-index relative to $\mathtt{t_{now}}$.

Denoting edges with edge-destinations and edge-sources enables us to describe the links between buckets. In the examples in this paper we use only three buckets, however, a general definition is:

Definition 11. A *DDN graph* $\mathtt{DDNG} = (\mathtt{TS}, \{\mathtt{sg}_1, \ldots, \mathtt{sg}_n\})$ where TS is a list of bucket-indexes ordered to reflect their current temporal status. Each bucket $\mathtt{sg}_i = (\mathtt{v}_i, \mathtt{es}_i, \mathtt{ed}_i)$ where $\mathtt{v}_i \subseteq \mathtt{V}$ are $\mathtt{sg}_i$'s vertices, $\mathtt{es}_i \subseteq \mathtt{GI}$ are $\mathtt{sg}_i$'s edge-sources, and $\mathtt{ed}_i \subseteq \mathtt{GI}$ are $\mathtt{sg}_i$'s edge-destinations.

This definition of directed edges decomposes the meaning of the (➜) function into two distinct functions which need to be executed to add an edge. These functions are `add-edge-destination(DDNG, NODE, TO-NODE)` and `add-edge-source(DDNG, NODE, FROM-NODE)`, they both operate on graph-indexes and are of type $\mathtt{DDNG} \rightarrow \mathtt{GI} \rightarrow \mathtt{GI} \rightarrow \mathtt{DDNG}$.

We have now described the macro elements used to specify the construction rules. The macro expansion is done before runtime to provide a function that is able to link nodes that match the given predicate tests. LISP macros were used to simply common functionality and enable the specification of similar DDN functions.

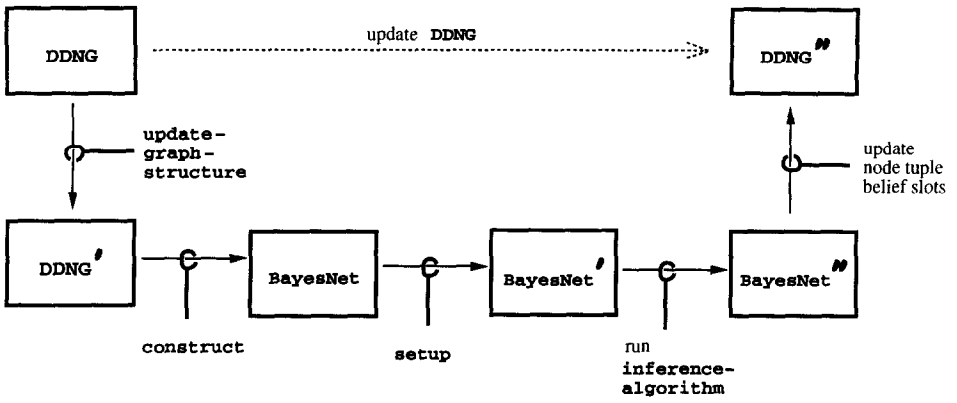


Fig. A.3. A break down of the various forms of DDN graph and Bayesian network used to perform one complete update to the DDN.

A.3. Graph properties

The complexity of the graph is related to the length of temporal history reasoned over. A graph that has no temporal history just reasons about the current time value. Increasing the length of history, also increases the size of the graph. If we assume that the number of nodes, n, at time t_{i-1} is the same as at time t_i then we have a linear increase in the number of nodes as the length of history, h, in increased, i.e., $O(n \times h)$. Neapolitan [79, p. 249] (also see Pearl [85, p. 174]) describes that in the case of an arbitrary singly connected network, the time requirement for the update of all variables is linear with respect to the number of variables in the network. This means that the DDN scales well with respect to the length of temporal history.

All the MACDDN construction macros, except `now-relationship`, generate $O(n)$ algorithms. `now-relationship` generates an $O(n \log n)$ algorithm, this is because we have a Cartesian product of `p` and `q`, and can ignore the symmetric component. This means that the generation of all the links between the nodes in the DDN graph is $\Theta(n \log n)$.

The number of nodes in the DDN at time `t`, is determined by the results from the preattentive operators, which are themselves dependent on the number of scene objects.

We can ensure that no loops are formed in the DDN graph at runtime if (1) we only allow non-temporal connections to be made between nodes that share the current time value, and (2) that no loop is formed by the directed edges added by the OSTCs. The first part is present in the use of NOWCs. The second is a static test that can be performed prior to runtime. These constraints restrict the DDN graphs to tree-like forms.

A.4. Updating algorithm

Updating the `DDNG` takes three stages as described in the NEI algorithm given in Fig. A.1 and this is also shown, in more detail, in Fig. A.3.

Table B.1
Mutual-proximity MACDDN construction rules.

Constructor macro	Pattern
now-connections	
$(\forall x, y \in \texttt{ABS1},\ \texttt{now-relationship}(N^{y}_{\{x\}}, N^{x}_{\{y\}}))$	$\texttt{add-vertex}(R^{\clubsuit}_{\{x,y\}}) \wedge N^{y}_{\{x\}} \rightarrow R^{\clubsuit}_{\{x,y\}} \wedge N^{x}_{\{y\}} \rightarrow R^{\clubsuit}_{\{x,y\}}$
one-step-temporal-connections	
$(\forall x \in \texttt{ABS1},\ \texttt{temporal-continuity}(N^{\clubsuit}_{\{x\}}))$	$N^{\clubsuit}_{\{x\}} \rightarrow N^{\clubsuit}_{\{x\}}$
$(\forall s \in \texttt{ABS2},\ \texttt{temporal-continuity}(R^{\clubsuit}_{s}))$	$R^{\clubsuit}_{s} \rightarrow R^{\clubsuit}_{s}$

- The first stage involves running the connection rules and is called `update-graph-structure`. It consists of two components defined in the MACDDN language called `node-building` and `node-linking` that are used to describe which node tuples are to be created and how they are to be connected.
- The second stage involves constructing an equivalent Bayesian network, and then setting it up by identifying the appropriate conditional probability matrix for each processor node in the Bayesian network. The belief slots from the DDNG node tuples provide the initial values for the root nodes in the Bayesian network.
- In the third stage the inference algorithm is run, with the results from this used to update the belief values of selected result nodes in the DDNG.

The second stage is really redundant and it would be much more efficient if Bayesian inference were applyed directly to the DDNG, so removing the need for creating the Bayesian network afresh for each execution cycle. Chang and Fung [22] describe an approach that can make local changes to Bayesian network topology and which might prove to be a useful technique for updating network structure.

Appendix B. Mutual-proximity graph

The form of the mutual-proximity graph is specified in Table B.1 which uses the node types N and R introduced in Table 4. We will also use the variables x and y to denote any two distinct objects (i.e., $x \neq y$) in the MACDDN constructor `now-relationship` where they are used to define the attribute symbol of the node tuple. In Table B.1, ABS stands for the currently active buffer-addresses, with $\texttt{ABS1} \subseteq \texttt{OV1}$ and ABS2 being the powerset of ABS1 such that $\texttt{ABS2} \subseteq \texttt{OV2}$. From the MACDDN specification in Table B.1 we generate the two functions that perform `node-building` and `node-linking` used in the NEI algorithm (see Section A.4). The `node-building` function generates all the nodes which are then linked into a graph by the `node-linking` function. Part of the `node-linking` function is shown in Fig. B.1 by the example expansion, in pseudo-code, of the NOWC MACDDN rule `mutual-proximity-relationship` shown in Table B.1. For more details see [54].

The DDN itself does not hold an overall picture of what is happening. For the creation of a fuller picture we use a separate Bayesian network called TASKNET which is described in Sections 7.2 and Appendix D.

```
procedure mutual-proximity-relationship(SG_now, A, B)
  if (node-type(A) = "nearest") and
    (node-type(B) = "nearest") and
    (node-owner(A) = reverse(node-owner(B))) do
      let R be make-node("mutual-proximity", node-owner(A))
      let RGI be make-graph-index(now, R)
      let AGI be make-graph-index(now, A)
      let BGI be make-graph-index(now, B)
      add-vertex(SG_now, R)
      add-edge-source(SG_now, RGI, AGI)
      add-edge-destination(SG_now, AGI, RGI)
      add-edge-source(SG_now, RGI, BGI)
      add-edge-destination(SG_now, BGI, RGI)
      od
```

Fig. B.1. An example expansion of the now-relationship constructor macro for the mutual-proximity-relationship which is part of the node-linking function.

Table C.1
Gross-change-in-motion MACDDN construction rules. Note that ABS stands for the currently active buffer-addresses, with ABS1 $\subseteq$ OV1.

Constructor macro	Pattern
now-connections	
($\forall x \in$ ABS1, now-ref($PS^{\clubsuit}_{\{x\}}$))	$PS^{\clubsuit}_{\{x\}} \rightarrow R^{\clubsuit}_{\{x\}}$
($\forall x \in$ ABS1, now-ref($PM^{\clubsuit}_{\{x\}}$))	$PM^{\clubsuit}_{\{x\}} \rightarrow R^{\clubsuit}_{\{x\}}$
one-step-temporal-connections	
($\forall x \in$ ABS1, temporal-continuity($R^{\clubsuit}_{\{x\}}$))	$R^{\clubsuit}_{\{x\}} \rightarrow R^{\clubsuit}_{\{x\}}$
($\forall x \in$ ABS1, temporal-continuity($PS^{\clubsuit}_{\{x\}}$))	$PS^{\clubsuit}_{\{x\}} \rightarrow PS^{\clubsuit}_{\{x\}}$
($\forall x \in$ ABS1, temporal-change($PS^{\clubsuit}_{\{x\}}$, $PM^{\clubsuit}_{\{x\}}$))	$PS^{\clubsuit}_{\{x\}} \rightarrow PM^{\clubsuit}_{\{x\}}$
($\forall x \in$ ABS1, temporal-change($PM^{\clubsuit}_{\{x\}}$, $PS^{\clubsuit}_{\{x\}}$))	$PM^{\clubsuit}_{\{x\}} \rightarrow PS^{\clubsuit}_{\{x\}}$
($\forall x \in$ ABS1, temporal-continuity($PM^{\clubsuit}_{\{x\}}$))	$PM^{\clubsuit}_{\{x\}} \rightarrow PM^{\clubsuit}_{\{x\}}$

Appendix C. Gross change in motion

To define gross-change-in-motion as a preattentive cue we specify the set of MACDDN construction rules shown in Table C.1. These rules use three node types PS, PM, and R, which stand for motion-prior value is stationary, motion-prior value is moving, and result. The R node is present to hold the belief value for {watch, ignore} that represents the result of running the inference-algorithm. This set of rules is independent of those for mutual-proximity and their use changes the behaviour of the system.

The NOWC rules combine the current and previous values denoting the interestingness of this object and the node type reflects the motion-prior of the object. When an object

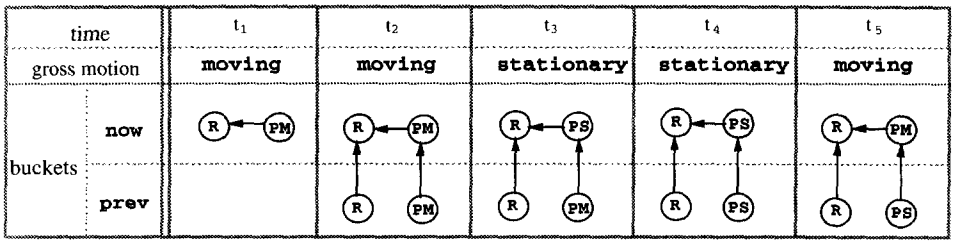


Fig. C.1. A sequence of graph examples to illustrate how gross-change-in-motion affects the allocation of PM and PS nodes.

is moving a PM node is used and when an object is stationary a PS node is used. To illustrate how these rules operate consider the sequence of graphs shown in Fig. C.1 where an object changes from moving to stationary to moving again. Along the top of the figure are the frame time-cells, with the two buckets now holding the contents of the t_{now} time-cell (i.e., the time of the current frame) and prev holding the contents of the t_{now-1} time-cell. In addition to the two buckets for now and prev, the figure also shows the nodes that are in these buckets and the links between and inside each bucket. Notice how the contents of the t_i now bucket become the t_{i-1}'s prev bucket, and that the links are changed to conform to the connection rules.

Appendix D. TASKNET, the DDN and allocate

As described in Section 7.2.2 each identified pair of buffer-addresses (i, j) is given a distinct TASKNET, denoted $\text{TASKNET}_{(i,j)}$. Also the buffer-addresses and their pairings are used in the DDN as owner symbols by the node tuples (see Appendix A). The mechanism, that manages the TASKNETs, reflects the three stages of the ALLOCATE component's allocation theory given in Section 6.2.1.

- The *focus-of-attention* stage ensures that on allocation the TASKNET is initialised and collecting information related to the pair, and then ensures that an agent marker is running on both selected objects.
- The *selective-attention* stage updates the root nodes and runs the inference algorithm.
- When an uninteresting situation is recognised, we first instantiate the belief value held in the pair's current DDN reference node. On the next clock tick when the DDN is updated and the inference algorithm run, if the relationship is still present, a high ignore value is placed in the reference node at what is now t_{now-1}. This will ensure that, unless a potentially interesting indicator is present at time t_{now}, the relationship will be ignored. This high ignore value is temporally propagated, causing both the allocated agents and TASKNET to be terminated for the pair. Thus achieving the *terminate-attention* stage.

Using this method, once a relationship is identified it can be ignored or continue to be watched.

References

[1] P.E. Agre, The dynamic structure of everyday life, Ph.D. Thesis, MIT AI Lab., AI-TR 1085, Cambridge, MA, 1988.

[2] P.E. Agre, D. Chapman, Pengi: an implementation of a theory of activity, in: Proceedings AAAI-87, Seattle, WA, 1987, pp. 268–272.

[3] J.F. Allen, Towards a general theory of action and time, Artificial Intelligence 23 (1984) 123–154.

[4] J.F. Allen, H.A. Kautz, R.N. Pelavin, J.D. Tenenberg, Reasoning about Plans, Morgan Kaufman, Los Altos, CA, 1991.

[5] A. Allport, Visual attention, in: M.I. Posner (Ed.), Foundations of Cognitive Science, MIT Press, Cambridge, MA, 1989, pp. 631–682.

[6] D.H. Ballard, Animate vision, Artificial Intelligence 48 (1991) 57–86.

[7] D.H. Ballard, C.M. Brown, Computer Vision, Prentice-Hall, Englewood Cliffs, NJ, 1982.

[8] D.H. Ballard, M.M. Hayhoe, Feng Li, S.D. Whitehead, Hand-eye coordination during sequential tasks, in: G.A. Horridge, H.B. Barlow, J.P. Frisby, M.A. Jeeves (Eds.), Natural and Artificial Low-level Seeing Systems (Special Issue), Philos. Trans. Roy. Soc. London Ser. B: Biological Sci. 337 (1992) 331–339.

[9] S. Baluja, D.A. Pomerleau, Using the representation in a neural network's hidden layer for task-specific focus of attention, in: Proceedings IJCAI-95, Montréal, Que., 1995, pp. 133–139.

[10] A. Baumberg, D. Hogg, Learning flexible models from image sequences, in: J.-O. Eklundh (Ed.), Proceedings 3rd ECCV Conference, Stockholm, Sweden, Vol. I, Lecture Notes in Computer Science, Vol. 800, Springer, Berlin, 1994, pp. 299–308.

[11] D.C. Beardslee, M. Wertheimer, Readings in Perception, Van Nostrand, New York, 1958.

[12] R. Bird, P. Wadler, Introduction to Functional Programming, Prentice-Hall, Englewood Cliffs, NJ, 1988.

[13] S.S. Blackman, Multiple-Target Tracking with Radar Applications, Artech House, 1986.

[14] R.D. Boyle, R.C. Thomas, Computer Vision: A First Course, Blackwell, Oxford, 1988.

[15] J.S. Breese, Construction of belief and decision networks, Comput. Intell. 8 (4) (1992) 624–647.

[16] R.A. Brooks, A robust layered control system for a mobile robot, IEEE J.. Robotics and Automation 2 (1) (1986) 14–23.

[17] R.A. Brooks, Intelligence without reason, in: Proceedings IJCAI-91, Sydney, Australia, 1991, pp. 569–595.

[18] K. Bühler, The deictic field of language and deictic words, in: R.J. Jarvella, W. Klein (Eds.), Speech, Place, and Action: Studies in Deixis and Related Topics John Wiley and Sons, New York, 1982, pp. 9–30 (abridged translation of: Sprachtheorie, Fischer, Jena, 1934).

[19] H. Buxton, D.R. Corrall, R. Godden, R. Howarth, Z. Hussain, M. Popovich, C.K. Sung, J. Thoméré, A.F. Toal, G. Sullivan, A. Worrall, VIEWS: Visual Inspection and Evaluation of Wide-area Scenes, in: IJCAI-91 Videotape Program, Morgan Kaufmann, Los Altos, CA, 1991 (ISBN 155860-183-X).

[20] H. Buxton, S.G. Gong, Visual surveillance in a dynamic and uncertain world, Artificial Intelligence 78 (1995) 371–405.

[21] S. Cameron, Collision detection by four dimensional intersection testing, IEEE Trans. Robotics and Automation 6 (3) (1990) 291–302.

[22] K.-C. Chang, R. Fung, Refinement and coarsening of Bayesian networks, in: P.P. Bonissone, M. Henrion, L.N. Kanal, J.F. Lemmer (Eds.), Uncertainty in Artificial Intelligence, Vol. 6 North-Holland, Amsterdam, 1991, pp. 435–445.

[23] D. Chapman, Planning for conjunctive goals, Artificial Intelligence 32 (1987) 333–377.

[24] D. Chapman, Penguins can make cake, AI Magazine 10 (4) (1989) 45–50.

[25] D. Chapman, Vision, Instruction and Action, MIT Press, Cambridge, MA, 1991.

[26] E. Charniak, R.P. Goldman, A probabilistic model of plan recognition, in: Proceedings AAAI-91, Anaheim, CA, 1991, pp. 160–165.

[27] A.N. Clark, Pattern recognition of noisy sequences of behavioural events using functional combinators, Comput. J. 37 (1994) 385–398.

[28] D.R. Corrall, A.N. Clark, A.G. Hill, Airside ground movements surveillance, in: NATO AGARD Symposium on Machine Intelligence in Air Traffic Management, Berlin, Germany, 1993, pp. 29:1–29:13.

[29] D.R. Corrall, A.G. Hill, Visual surveillance, GEC Review 8 (1992) 15–27.
[30] P. Dagum, A. Galper, E. Horvitz, Dynamic network models for forecasting, in: Uncertainty in Artificial Intelligence, Vol. 8, Morgan Kaufman, Los Altos, CA, 1992, pp. 41–48.
[31] T. Dean, T. Camus, J. Kirman, Sequential decision making for active perception, in: Proceedings Image Understanding Workshop, Pittsburgh, PA, 1990, pp. 889–894.
[32] T.L. Dean, D.V. McDermott, Temporal data base management, Artificial Intelligence 32 (1987) 1–55.
[33] Li Du, G.D. Sullivan, K.B. Baker, Quantitative analysis of the viewpoint consistency constraint in model-based vision, in: Proceedings 4th International Conference on Computer Vision, Berlin, Germany, 1993, pp. 632–639.
[34] M.A. Fischler, O. Firschein (Eds.), Readings in Computer Vision: Issues, Problems, Principles, and Paradigms, Morgan Kaufman, Los Altos, CA, 1987.
[35] M.M. Fleck, Boundaries and topological algorithms, Ph.D. Thesis, MIT AI Lab., AI-TR 1065, Cambridge, MA, 1988.
[36] M.M. Fleck, Representing space for practical reasoning, Image and Vision Computing 6 (2) (1988) 75–86.
[37] M.M. Fleck, The topology of boundaries, Artificial Intelligence 80 (1996) 1–27.
[38] J.A. Fodor, The Modularity of Mind: An Essay on Faculty Psychology, Press, Cambridge, MA, 1983.
[39] R.A. Frost, Constructing programs as executable attribute grammars, Comput. J. 35 (4) (1992) 376–387.
[40] K.S. Fu, Syntactic Methods in Pattern Recognition, Academic Press, New York, 1974.
[41] H. Garfinkel, Studies in Ethnomethodology, Prentice-Hall, Englewood Cliffs, NJ, 1967.
[42] E. Geake, Camera keeps an eye on airport vehicles, New Scientist 136 (1850) (1992) 21.
[43] J.J. Gibson, The Ecological Approach to Visual Perception, Houghton Mifflin, Boston, MA, 1979.
[44] J.J. Gibson, L.E. Crooks, A theoretical field-analysis of automobile-driving, in: E. Reed, R. Jones (Eds.), Reasons for Realism: Selected Essays of James J. Gibson, Lawrence Erlbaum Associates, London, 1982, pp. 119–136; also in: Amer. J. Psychology 51 (1938) 453–471.
[45] S.G. Gong, H. Buxton, Bayesian nets for mapping contextual knowledge to computational constraints in motion segmentation and tracking, in: Proceedings 4th British Machine Vision Conference, Guildford, Surrey, UK, BMVA Press, 1993, pp. 229–238.
[46] I.E. Gordon, Theories of Visual Perception, John Wiley and Sons, New York, 1989.
[47] S. Hanks, D. McDermott, Modeling a dynamic and uncertain world I: symbolic and probabilistic reasoning about change, Artificial Intelligence 66 (1) (1994) 1–55.
[48] P. Hayes, The second naive physics manifesto, in: J.R. Hobbs, R.C. Moore (Eds.), Formal Theories of the Commonsense World, Ablex, Norwood, NJ, 1985, pp. 1–36.
[49] J. Heritage, Garfinkel and Ethnomethodology, Polity Press, Cambridge, UK, 1984.
[50] A. Herskovits, Language and Spatial Cognition: An Interdisciplinary Study of the Prepositions in English, Cambridge University Press, Cambridge, UK, 1986.
[51] B.K.P. Horn, Robot Vision, MIT Press, Cambridge, MA, 1986.
[52] I. Horswill, Visual routines and visual search: a real-time implementation and an automata-theoretic analysis, in: Proceedings IJCAI-95, Montréal, Que., 1995, pp. 56–62.
[53] R.J. Howarth, MACNET: a language for situated AI systems, Technical Report 684, Queen Mary and Westfield College, London, 1994.
[54] R.J. Howarth, Spatial representation, reasoning and control for a surveillance system, Ph.D. Thesis, Queen Mary and Westfield College, University of London, 1994.
[55] R.J. Howarth, Interpreting a dynamic and uncertain world: high-level vision, Artificial Intelligence Review 9 (1) (1995) 37–63.
[56] R.J. Howarth, H. Buxton, An analogical representation of space and time, Image and Vision Computing 10 (7) (1992) 467–478.
[57] R.J. Howarth, H. Buxton, Analogical representation of spatial events for understanding traffic behaviour, in: B. Neumann (Ed.), Proceedings ECAI-92, Vienna, Austria, John Wiley and Sons, New York, 1992, pp. 785–789.
[58] R.J. Howarth, H. Buxton, Selective attention in dynamic vision, in: Proceedings IJCAI-93, Chambéry, France, 1993, pp. 1579–1584.

[59] R.J. Howarth, H. Buxton, Visual surveillance monitoring and watching, in: B. Buxton, R. Cipolla (Eds.), Proceedings 4th European Conference on Computer Vision, Cambridge, UK, Vol. II, Lecture Notes in Computer Science, Vol. 1065, Springer, Berlin, 1996, pp. 321–334.
[60] R.J. Howarth, H. Buxton, Attentional control for visual surveillance, in: S. Maybank, T. Tan (Eds.), IEEE Workshop on Visual Surveillance, Bombay, India, IEEE Press, New York, 1998, pp. 86–93 (held at ICCV-98).
[61] E. Hutchins, Understanding Micronesian navigation, in: G. Dedre, A.L. Stevens (Eds.), Mental Models, Lawrence Erlbaum Associates, London, 1983, pp. 191–225.
[62] U. Kiaerulff, A computational scheme for reasoning in dynamic probabilistic networks, in: Uncertainty in Artificial Intelligence, Vol. 8, Morgan Kaufman, Los Altos, CA, 1992, pp. 121–129.
[63] J.H. Kim, J. Pearl, A computational model for causal and diagnostic reasoning in inference systems, in: Proceedings IJCAI-83, Karlsruhe, Germany, 1983, pp. 190–193.
[64] S. King, S. Motet, J. Thoméré, F. Arlabosse, A visual surveillance system for incident detection, in: 1993 AAAI Workshop on AI in Intelligent Vehicle Highway Systems, AAAI Press, Menlo Park, CA, 1994, pp. 30–36.
[65] C. Koch, S. Ullman, Shifts in selective visual attention: towards the underlying neural circuity, Human Neurobiology 4 (1985) 219–227.
[66] J.J. Koenderink, Solid Shape, MIT Press, Cambridge, MA, 1990.
[67] S.M. Kosslyn, R.A. Flynn, J.B. Amsterdam, G. Wang, Components of high-level vision: a cognitive neuroscience analysis and accounts of neurological syndromes, Cognition 34 (1990) 203–277.
[68] D.L. LaBerge, Attention, Psychological Sci. 1 (3) (1990) 156–162.
[69] J.-C. Latombe, Robot Motion Planning, Kluwer Academic Publishers, Dordrecht, 1991.
[70] A.K. Mackworth, Quick and clean: constraint-based vision for situated robots, in: Proceedings IEEE International Conference on Image Processing, Lausanne, Switzerland, Vol. II, IEEE Press, New York, 1996, pp. 789–791.
[71] J.V. Mahoney, S. Ullman, Image chunking defining spatial building blocks for scene analysis, in: Z.W. Pylyshyn (Ed.), Computational Processes in Human Vision: An Interdisciplinary Perspective, Ablex, Norwood, NJ, 1988, pp. 169–209.
[72] D. Marr, Vision, W.H. Freeman and Company, New York, 1982.
[73] R.F. Marslin, G.D. Sullivan, K.B. Baker, Kalman filters in constrained model-based tracking, in: P. Mowforth (Ed.), British Machine Vision Conference 1991, Glasgow, UK, Springer, Berlin, 1991, pp. 371–374.
[74] D. McDermott, A temporal logic for reasoning about processes and plans, Cognitive Sci. 6 (1982) 101–155.
[75] G.A. Miller, The magic number seven, plus or minus two: some limits on our capacity for processing information, Psychological Review 63 (1956) 81–97; also in: D.C. Beardslee, M. Wertheimer, Readings in Perception, Van Nostrand, New York, 1958, pp. 90–114.
[76] M.C. Mozer, The Perception of Multiple Objects: A Connectionist Approach, MIT Press, Cambridge, MA, 1991.
[77] D.W. Murray, K.J. Bradshaw, P.F. McLauchlan, I.D. Reid, P.M. Sharkey, Driving saccade to pursuit using image motion, Internat. J. Computer Vision 16 (1995) 205–228.
[78] H.-H. Nagel, From image sequences towards conceptual descriptions, Image and Vision Computing 6 (2) (1988) 59–74.
[79] R.E. Neapolitan, Probabilistic Reasoning in Expert Systems: Theory and Algorithms, John Wiley and Sons, New York, 1990.
[80] B. Neumann, Natural language descriptions of time-varying scenes, in: D.L. Waltz (Ed.), Semantic Structures: Advances in Natural Language Processing, Lawrence Erlbaum Associates, London, 1989, pp. 167–206.
[81] D. Newtson, Foundations of attribution: the perception of ongoing behaviour, in: J.H. Harvey, W.J. Ickes, R.F. Kidd (Eds.), New Directions in Attribution Research: Vol. 1, Lawrence Erlbaum Associates, London, 1976, pp. 223–247.
[82] A.E. Nicholson, J.M. Brady, The data association problem when monitoring robot vehicles using dynamic belief networks, in: B. Neumann (Ed.), Proceedings ECAI-92, Vienna, Austria, John Wiley and Sons, New York, 1992, pp. 689–693.

[83] N.J. Nilsson, Teleo-reactive programs for agent control, J. Artif. Intell. Research 1 (1994) 139–158.
[84] D.A. Norman, Cognition in the head and in the world: an introduction to the special issue on situated action, Cognitive Science 17 (1993) 1–6.
[85] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufman, Los Altos, CA, 1988.
[86] Z.W. Pylyshyn, R.W. Storm, Tracking multiple independent targets: evidence for a parallel tracking mechanism, Spatial Vision 3 (1988) 179–197.
[87] R.D. Rimey, C.M. Brown, Where to look next using a Bayes Net: incorporating geometric relations, in: G. Sandini (Ed.), Proceedings 2nd European Conference on Computer Vision, Santa Margherita Ligure, Italy, Lecture Notes in Computer Science, Vol. 588, Springer, Berlin, 1992, pp. 542–550.
[88] R.D. Rimey, C.M. Brown, Control of selective perception using Bayes nets and decision theory, Internat. J. Computer Vision 12 (2,3) (1994) 173–207.
[89] I. Rock, The Logic of Perception, MIT Press, Cambridge, MA, 1983.
[90] S.J. Rosenschein, L.P. Kaelbling, The synthesis of digital machines with provable epistemic properties, in: J.Y. Halpern (Ed.), Theoretical Aspects of Reasoning about Knowledge, Proceedings 1986 Conference, Timberline, OR, Morgan Kaufman, Los Altos, CA, 1986, pp. 83–98.
[91] G. Ryle, The Concept of Mind, Hutchinson, 1949; Penguin Books, 1990.
[92] R.C. Schank, R.P. Abelson, Scripts, Plans, Goals and Understanding, Lawrence Erlbaum Associates, London, 1977.
[93] H. Schöne, Spatial Orientation: The Spatial Control of Behaviour in Animals and Man, Princeton University Press, Princeton, NJ, 1984 (English translation of: Orientierung im Raum ..., Wissenschaftliche Verlagsgesellschaft, 1980).
[94] Y. Shoham, Reasoning about Change: Time and Causation from the Standpoint of Artificial Intelligence, MIT Press, Cambridge, MA, 1988.
[95] L.E. Sucar, D.F. Gillies, Expressing relational and temporal knowledge in visual probabilistic networks, in: Uncertainty in Artificial Intelligence, Vol. 8, Morgan Kaufman, Los Altos, CA, 1992, pp. 303–309.
[96] L. Suchman, Plans and Situated Actions: The Problems of Human–Machine Communication, Cambridge University Press, Cambridge, UK, 1987.
[97] G.D. Sullivan, Visual interpretation of known objects in constrained scenes, in: G.A. Horridge H.B. Barlow, J.P. Frisby, M.A. Jeeves (Eds.), Natural and Artificial Low-level Seeing Systems (Special Issue), Philos. Trans. Roy. Soc. London Ser. B: Biological Sci. 337 (1992) 361–370.
[98] B. Taylor, Tense and continuity, Linguistics and Philosophy 1 (1977) 199–220.
[99] C. Terman, Simulation tools for digital LSI design, Ph.D. Thesis, MIT Laboratory for Computer Science, MIT/LCS/TR-304, Cambridge, MA, 1983.
[100] A.F. Toal, H. Buxton, Spatio-temporal reasoning within a traffic surveillance system, in: G. Sandini (Ed.), Proceedings 2nd European Conference on Computer Vision, Santa Margherita Ligure, Italy, Lecture Notes in Computer Science, Vol. 588, Springer, Berlin, 1992, pp. 884–892.
[101] A. Treisman, Preattentive processing in vision, in: Z.W. Pylyshyn (Ed.), Computational Processes in Human Vision: An Interdisciplinary Perspective, Ablex, Norwood, NJ, 1988, pp. 341–369; also in: Comput. Vision Graphics Image Process 31 (1985) 156–177.
[102] J.K. Tsotsos, Toward a computational model of attention, in: T. Papathomas, C. Chubb, A. Gorea, E. Kowler (Eds.), Early Vision and Beyond, MIT Press, Cambridge, MA, 1995, pp. 207–218.
[103] J.K. Tsotsos, S. M Culhane, W.Y.K. Wai, Y. Lai, N. Davis, F. Nuflo, Modeling visual attention via selective tuning, Artificial Intelligence 78 (1995) 507–545.
[104] S. Ullman, Visual routines, in: S. Pinker (Ed.), Visual Cognition, MIT Press, Cambridge, MA, 1985, pp. 97–159; also in: M.A. Fischler, O. Firschein (Eds.), Readings in Computer Vision: Issues, Problems, Principles, and Paradigms, Morgan Kaufman, Los Altos, CA, 1987, pp. 298–328; also in: Cognition 18 (1984).
[105] J.F.A.K. van Benthem, The Logic of Time, Reidel, Dordrecht, 1983.
[106] A.H.C. van der Heijden, Selective Attention in Vision, Routledge & Kegan Paul, London, 1992.
[107] J.H.C. Whitehead, Combinatorial homotopy I, Bull. Amer. Math. Soc. 55 (1949) 213–245; also in: I.M. James (Ed.), The Mathematical Works of J.H.C. Whitehead: Vol. III on Homotopy Theory, Pergamon Press, Oxford, 1962, pp. 85–177.

[108] T. Winograd, F. Flores, Understanding Computers and Cognition: A New Foundation for Design, Ablex, Norwood, NJ, 1986.
[109] W.A. Woods, Transition network grammars for natural language analysis, Comm. ACM 13 (10) (1970) 591–606.
[110] A.D. Worrall, R.F. Marslin, G.D. Sullivan, K.B. Baker, Model-based tracking, in: P. Mowforth (Ed.), British Machine Vision Conference 1991, Glasgow, UK, Springer, Berlin, 1991, pp. 310–318.
[111] A.D. Worrall, G.D. Sullivan, K.B. Baker, Advances in model-based traffic vision, in: Proceedings 4th British Machine Vision Conference, Guildford, Surrey, UK, BMVA Press, 1993, pp. 559–568.