# WITAS: An Intelligent Autonomous Aircraft Using Active Vision

Gösta Granlund, Klas Nordberg, Johan Wiklund,
Patrick Doherty, Erik Skarman, Erik Sandewall
Linköping University, SE-581 83 Linköping, Sweden
http://www.ida.liu.se/ext/witas/

# 1 Introduction

The WITAS[1] Unmanned Aerial Vehicle Project is a long term basic research project located at Linköping University (LIU), Sweden. The project is multi-disciplinary in nature and involves cooperation with different departments at LIU, and a number of other universities in Europe, the USA, and South America. In addition to academic cooperation, the project involves collaboration with a number of private companies supplying products and expertise related to simulation tools and models, and the hardware and sensory platforms used for actual flight experimentation with the UAV. Currently, the project is in its second phase with an intended duration from 2000-2003.

This paper will begin with a brief overview of the project, but will focus primarily on the computer vision related issues associated with interpreting the operational environment which consists of traffic and road networks and vehicular patterns associated with these networks.

## 1.1 UAV Research and the WITAS UAV Project

Ground robotics has been an essential part of artificial intelligence research for some time. The use of UAVs as a testbed for both artificial intelligence and on-board computer vision research is quite recent and there are only a handful of universities around the world currently doing research in this area. The WITAS UAV project distinguishes itself from many of the other projects in terms of breadth of topics covered and focus on both high- and low-level autonomy and their integration with an active vision and a ground control dialogue system.

A generic UAV setup consists of an air vehicle with payload (quite often a still or video camera), a tactical control station (usually stationary) with one or more humans in the loop, and a data-link between the station and air vehicle used for downloading images and data and for uploading navigation and camera control commands. A mission plan is quite often represented as a database of waypoint coordinates with associated loitering and data collection commands. The mission plan is either uplinked via the radio link during the mission or already provided when the UAV begins its mission. Data collection activities generally result in a sequence of still images or analog video downloaded via the radio link or collected upon the UAV's return. The data is generally interpreted by a group of experts manually either during flight or after the UAV's return. Much of the recent research has focused on low-level autonomy and robust flight control issues such as taking-off,

[1]The Wallenberg Laboratory for Information Technology and Autonomous Systems(Pronounced *Vee-Tas*).

landing and getting the UAV to fly robustly from one waypoint to another.

The WITAS UAV project focuses not only on low-level autonomy, but on intermediate and high-level autonomy coupled with an active vision system consisting of digital video and IR cameras as the main sensory components. The intention is that mission goals are provided in a declarative form and the deliberative/reactive system generates the database of waypoints automatically which include loitering and sensory payload commands. The plans are executed and monitored in real-time, sometimes resulting in modification to all or part of the original plan. The on-board active vision system interprets the scene or focus of attention below in cooperation with the reactive and deliberative parts of the overall architecture to interpret the ongoing events below. We assume the use of an on-board geographic information system containing a rich amount of information about the road systems and geographic terrain.

With current technology, it is unrealistic to assume that aviation authorities will permit autonomous aerial vehicles to fly unattended over populated areas without some form of control from the ground station which may include line-of-sight constraints. Consequently, the ground operator is and will remain a vital part of an integrated UAV system in the near future. In order to guarantee clear and concise communication between the UAV and ground operator, multi-modal interfaces which enhance such communication play a fundamental role in the overall design of such systems. Part of the research in this project involves just such a system where the ground operator can communicate with the UAV at various levels of abstraction using speech, pointing devices and video viewing. In addition, the actual communication devices may range from standard laptops to smaller PDP like devices. This opens up an interesting set of issues related to the bandwidth of the interfaces and their dynamic management.

In summary, although the full spectrum of issues ranging from low-level control and signal processing to intermediate and high-level autonomy are an essential part of the project, major focus is being placed on the development of deliberative/reactive system software architectures, integration and development of active vision systems and dialogue systems, and knowledge representation issues such as planning, execution monitoring, chronicle recognition, and temporal and spatial reasoning.

## 1.2 A Typical Scenario

A typical mission goal for our UAV might involve finding, identifying, tracking and trying to discern various patterns associated with a vehicle. Such a mission can be described in terms of achieving the following tasks:

- Locate a vehicle of a particular signature in a designated region. The signature might be given in terms of color and geometry in addition to other distinguishing characteristics.

- Assuming it is correctly identified as the target vehicle, begin tracking the vehicle in a designated region using the on-board GIS to help deal with vehicle occlusion such as going into a tunnel or under a bridge.

- Communicate with ground control for help and advice.

- Attempt to identify certain patterns of behavior such as overtaking, erratic driving, or traversing intersections.

- Return to home base after a designated amount of time.

For this scenario, it may be assumed that the UAV receives as input a vehicle signature, the time (metric) and location coordinates where it was last observed, the designated area of interest, the patterns of interest, and additional time constraints as to the duration of the mission.

This particular scenario is extremely complex and involves robust navigation, high-level decision making, generation of plans, temporal reasoning, dealing with uncertainty both with sensory and qualitative data, chronicle recognition, use of geographic information, anchoring, registration, and signal to symbol transformation of sensory data. Each of these functionalities is a research issue in itself and the integration of these functionalities is probably the most difficult research issue involved.

## 1.3 Research Methodology

Due to the complexity of both the basic research issues involved in addition to the software and hardware integration issues, the research methodology used has been one of iteration on a base prototype architecture developed early in the project. The iterations are driven by scenarios set up in the operational environment and new functionalities necessary to complete mission goals associated with the scenarios. A simulation architecture has been developed to support initial experimentation and debugging of different parts of the architecture. Video sequences gathered using one of our UAV platforms is currently used for off-board computer vision experimentation. Experimental flights combined with simulation experiments will be used throughout the remainder of the project. It is not practical to fly the UAV on a daily or even weekly basis, therefore simulation techniques have an important and major role in the project.

## 2 Project Overview

The long term goal of the WITAS UAV Project is the development of the technologies and functionalities necessary for the successful deployment of a fully autonomous UAV operating over road and traffic networks. While operating over such an operational environment, the UAV should be able to navigate autonomously at different altitudes (including autonomous take-off and landing), plan for mission goals such as locating, identifying, tracking and monitoring different vehicle types, and construct internal representations of its focus of attention for use in achieving its mission goals. Additionally, it should be able to identify complex patterns of behavior such as vehicle overtaking, traversing of intersections, parking lot activities, etc.

The achievement of such an ambitious goal involves dealing with a combination of complex practical and basic research issues together with the integration of research results with existing and newly developed hardware and software technologies used in the project. Successful completion of the project involves (at the very least),

- Development of reliable software and hardware architectures with both deliberative and reactive components for autonomous control of UAV platforms;
- Development of sensory platforms and sensory interpretation techniques with an emphasis on active vision systems to deal with real-time constraints in processing sensory data;
- Development of efficient inferencing and algorithmic techniques to access geographic, spatial and temporal information of both a dynamic and static character associated with the operational environment;
- Development of simulation, specification and verification techniques and modeling tools specific to the complex environments and functionalities associated with the project.

We will touch upon each of these functionalities in the following sections of the paper.

As stated in the introduction, this is a basic research project with a focus on identifying and developing the algorithmic, knowledge representation, software, hardware and sensory functionalities necessary for deploying an autonomous UAV in and over the traffic and road network operational environment. This particular operational environment was chosen because it is sufficiently complex to require both deliberative and reactive behavior on the part of the UAV and a challenging set of issues for the active vision system, but at the same time, it still contains a great deal of structure, such as the road system, to help deal with some of the complexity of the environment.

Though the project does not include development of a commercial product as a major focus, there are a number of practical applications of potential value associated with this and similar operational environments. One can view a UAV with the stated capabilities as an emergency services assistant capable of guiding fire, police or ambulance personnel to, or through, the scene of a catastrophe, monitoring the current situation and relaying real-time video or still photos, perhaps interpreted, to ground personal. The UAV could also be viewed as a mobile sensory platform in a real-time traffic control systems network, moving to areas of congestion, interpreting the reasons for the congestion and relaying video and information to the traffic control center. Finally, the UAV could be used by police and custom services officials for reconnaissance and monitoring.

When choosing experimental UAV platforms, there are essentially two classes of vehicles to choose from, fixed-wing or vertical take-off and landing systems (VTOL). We have chosen to experiment with VTOL systems due to the nature of the operational environment and mission goals. That part of the sensory platform associated with the vision system currently consists of a digital video camera in a gimbaled housing, but will eventually consist of bore calibrated infrared and digital video cameras in a specially designed housing.

## 3 UAV Platform

We are currently collaborating with Scandicraft Systems AB, a university spin-off company that develops autonomous mini-helicopters. The current version in the new series, the Apid Mk III, flew for the first time in October 1999.

The Apid measures 3.63 m from main rotor to tail rotor and is 0.7 m wide. The main and tail rotors measure 2.98 m and 0.62 m, respectively. A 2-cycle, single cylinder modified go-cart motor is used providing 15 HP at 9500 r/min. Fuel usage averages 5 l/h in hovering mode and 2.5 l/h in horizontal and vertical flying modes. The body is manufactured using carbon fiber/kevlar sandwich material. The payload is 20 kg including fuel. The flight control system software is built around the realtime kernel RTkernel and contains an inertial navigation system with gyro, accelerometers and a tilt sensor, which provide the control system with the platforms attitude and velocity.

Other on-board sensors include a radar altimeter, an IR altimeter, barometer, compass and motor RPM sensor. The platform also contains a differential GPS for positioning. A 1 Watt radio link at 439 MHz is used for 2-way communication with the ground station. Information from all sensors can be received from the platform and control commands can be sent to the platform.

The camera system currently being used in experimental flights can contain either a digital video or IR camera. The cameras are contained in a housing with gyro-stabilized pan-tilt gimbals developed by PolyTech/Flir Systems. Panning, tilt and camera zoom can be controlled from the ground via a separate radio link, or on-board using a specially designed interface. A new housing is currently being developed by PolyTech/Flir Systems which will contain bore-calibrated digital video and IR cameras.

## 4 Software Architecture

The Intelligent Vehicle Control Architecture (IVCA) used in the project can be characterized as a multi-layered hybrid deliberative/reactive software architecture with functionality and structure similar in spirit to, but not the same as, the three-layered architectures proposed by Firby [3] and Gat [4]. Conceptually, the architecture can be viewed as consisting of three separate layers, each containing a collection of asynchronous computational processes:

- Deliberative Layer – This layer contains a loose collection of high level services such as planners, trajectory planners, predictors, and chronicle recognition packages. These services are called by the reactive layer when its own packages can not achieve mission goals independently.
- Reactive Layer – The reactive layer contains a library of reactive programs specified in the language CONTAPS (concurrent task achieving procedures) developed in the project. A CONTAP can be viewed as an augmented automaton or a collection of triggered rules which have local state and the ability to open channels of communication to other layers or parts of the architecture including other CONTAPS.
- Process Layer – The process layer is responsible for the concurrent computation of feedback control loops tightly coupling sensing with actuation. It is at this layer that the flight navigation and camera control processing reside.

The architecture contains two main information repositories:

- The Knowledge Structure Repository (KSR) – The KSR contains different types of information associated with high-level deliberative services such as the planner and chronicle recognition packages in addition to the Dynamic Object Repository (DOR) which has the flavor of an object-oriented active database. The DOR is a central and important part of the architecture in that it provides for seamless integration between the signal processing associated with the active vision system and the qualitative processing associated with the reactive and deliberative layers.

- The Geographic Data Repository (GDR) – The GDR is a layered knowledge structure containing spatial data about the operational environment(OE) and access to non-spatial data associated with the spatial structures. The lowest layer contains a repository of digital images over the OE. The next layer contains elevation data correlated with the digital images. The layer above that contains information about the road networks. The layer above that contains landmark and other data.

The different processes associated with the layers in the architecture can access information stored in these structures by opening communication channels to them.

The deliberative and reactive layers of the architecture communicate directly with the vision system via a sensory command language which requests services from the vision system and indirectly via data stored in the DOR as a result of low and intermediate image processing.

For additional system architecture details, see Doherty [2]. In the remainder of the paper, we will focus on computer vision related topics.

## 5 Overview of Vision System

This project includes a number of research problems of in computer vision. Although they are traditional parts of the vision and robotics fields, they are accentuated by the use in autonomous systems, requiring particularly robust functionalities. As the use of autonomous systems will increase in other areas, this research work is essential for future development [6].

### 5.1 A Fast and Flexible FOA Vision Architecture

The particular requirements give the motivation and the opportunity to develop and to test a new sensing, processing and representation architecture. The traditional structure of a video camera followed by a processor does not provide the performance required for most demanding situations in robotics. The project implies the development of a new vision architecture which will very rapidly switch its Focus of Attention (FOA) between different parts and aspects of the image information. It will contain three major units:

- *Actively Controlled Sensing*
- *Fast Processing Architecture*
- *Multi-Resolution Semantic Scene and Object Representation*

Many of the components of this structure have been tested as stand-alone procedures earlier, but it has not been feasible to integrate all of them into one system. The real gain will however appear as all of them are combined into one system, as any one of them does not separately achieve full performance on its own.

The structure of the overall system, with an emphasis on the vision system, is as indicated in Figure 1.

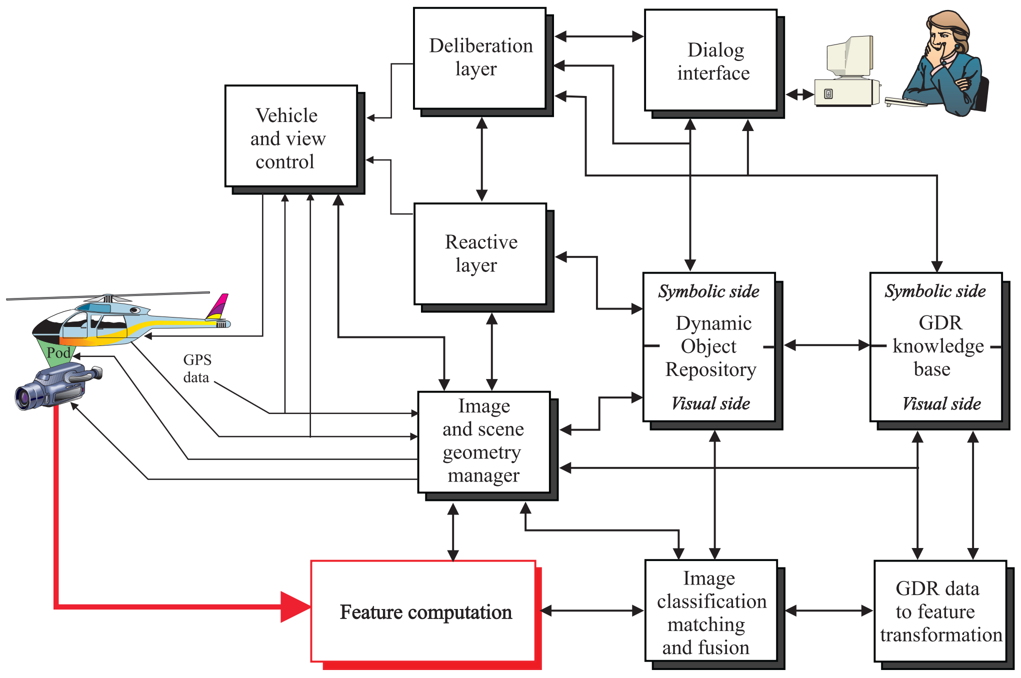

Figure 1: Structure of WITAS control system with vision functions emphasized.

The document will start with a presentation of tasks to be performed by the system, and the different modes in which it will operate. After this, the functionalities required will be presented, and finally the system components necessary will be discussed.

## 5.2 Computation structures

Another important requirement relates to the hardware implementation of the proposed system architecture, both vision processing and higher processing levels. These requirements can be further decomposed into two major fields, where the first one includes all practical aspects of designing a computation platform which fits within the physical limitations imposed by the UAV, and the second one relates to software design of the low level vision processing.

The class of UAV platforms which is considered for the project, described in Section 3, has a typical payload of 20–30 kg and produces approximately 150 W of electrical power. From these figures one has to subtract the weight and power consumption of the camera (or cameras) plus the camera gimbal, and various boxes which are used for the basic control of the UAV and communication to the ground station. Roughly, this leaves 15 kg of payload and 100 W of electrical power to the computer system that is used for both image analysis and all the higher layers of the system architecture. To make a comparison, this amount of electrical power is not sufficient to keep more than a single high-end Pentium III processor in operation.

The only processors which fit within these limitations are either DSPs, which in general have a much higher computational performance per Watt than conventional processors, or low-speed versions of conventional processors (Pentium, PowerPC, etc) for embedded applications. Initial studies indicate that a reasonable solution is a hybrid of DSPs and conventional processors.

The DSPs will work mainly with the preprocessing of the image stream by producing one or several regions of interest, and if so required, bursts of such regions (see Section 8). The sizes of these regions and bursts have to be adapted to the processing power of the subsequent processors of conventional type. These processors implement most of the feature extraction used for defining the various properties and events of the scene. The division of different types of processing into the two classes of processors is based on executing more or less fixed and simple operations in the DSP part, and allowing the processing to be much more flexible and complex in the part made up by conventional processors. The reason for this division is the experience that non-conventional processors (such as DSPs) are much more difficult to operate in a dynamic situation where, e.g., new modes of operations need to be added, or one has to change or upgrade from one processor family to another.

The overall software design of the vision processing is of course related to these considerations. On one hand, it is highly desirable to avoid tedious and laborious implementations of vision processing in special purpose hardware, which in addition usually cannot be reused when going from one processor family to another. This can be done by keeping the software implementation neutral to hardware platforms, e.g., in C language. On the other hand, it is also desirable to obtain a sufficiently high processing power, which typically is done by employing special purpose processors such as DSPs. Furthermore, implementations in, e.g., C language can be made much more flexible and maintainable over time than special purpose implementations.

One research goal of the project is to define a level of description that can be used for image processing operations, and which can be used in both domains; for C implementations on general purpose processors, and dedicated implementations on DPSs. The result is an API which allows the application programmer to describe the vision processing structure independent of the specific hardware platform. The API can then be implemented either in C (or any other general programming language) and, e.g., assembly language for a specific DPS.

## 6 Flight range modes

In the subsequent presentation, the required vision functionalities are discussed for three different modes of operation of the UAV:

- **Landscape Range**
- **Scenario Range**
- **Close Contact Range**

### 6.1 Landscape Range

As the UAV reaches the proper altitude range for landscape navigation, 500 - 2000 m, (the altitude may in practical experiments have to be restricted to a maximum of 300 m for regulatory reasons), it flies according to a path of preprogrammed way points, which may be monitored using vision or GPS coordinates.

#### 6.1.1 GPS Guided Mode

In the GPS guided mode, these way points are defined as GPS coordinates, which determine the progression of the flight from GPS coordinate and altimeter readings. To pre-program the flight path, the GDR system is used interactively to allow the operator define way points, the altitude and the behavior between and around the way points. The initial pre-programmed flight path is stored in a data base, where the properties of paths and way points (nodes) can be separately accessed and edited. In this way, certain parts of a road can be marked for particular attention. This can be visualized on top of the GDR rendering of the region in question. This system can as well be used to visualize the movement of the UAV during operation, using a rendering of the actual GPS coordinate data.

In addition to the preprogrammed flight path, a set of appropriate *mission strategies* is selected or defined in a separate database. This defines how the system shall react in different parts of the path, and under various conditions, commands given and encounters during its mission. This is independent from the actual flight path, which is defined separately.

Vision is in this phase used essentially to confirm the position of the UAV in terms of visual landmarks. That is, to establish correspondence between structures observed on the ground and predicted appearance of structures taken from the GDR data base. This is done in preparation for surveillance and for transition into the *scenario range*, where visual orientation is essential.

The mission strategies define the exploratory procedures to undertake. Normally as the UAV moves forward, it will navigate as well as perform a scanning for predetermined objects. These objects are defined as a hierarchy, where objects contain sub-objects. An object may at some time be a road, while a sub-object is a car on this road.

Single images are taken with the video camera. These are subjected to interpretation using procedures described elsewhere. They provide descriptions of roads, traffic structures and other significant structures of the landscape.

It is assumed that at this altitude the landscape can be viewed as a 2-dimensional surface, unless the landscape is very rough and altitude data is readily available. The perspective transformations consequently become simple. The actual state parameters of the UAV (altitude, orientation, camera parameters, etc) are used to transform a view from data in the GDR data base. This transformation modifies geometry and structural features in terms of the descriptors used for the matching to corresponding descriptors from the image.

### 6.1.2 Vision Guided Mode

This is a complementary mode to the preceding one, to evaluate the functionality of a self-contained UAV, without external aids for navigation control. The preprogramming of the flight path is done in a similar interactive way with reference to the GDR data base. In this case, way points are ensured to correspond to visually identifiable structures such as road crossings, bridges, etc, rather then just points.

The marked way points are analyzed with respect to their approximate position using the geometrical and scale data of the GDR data base. In addition, the marked way point structures are interpreted immediately in terms of the structural representation used, and stored in the data base as a record of the way point. For practical purposes of mission control safety and display, etc, GPS coordinates will obviously as well be derived and stored in a separate data base to be available for display and other purposes as above.

#### 6.1.3 Preparation for Change of Mode

As the UAV has approached the region for its attention, it will prepare for a transition to the scenario range. It will make sure that it has the geographical landmarks under attention, after which it can start to descend. In the transition to the scenario range, the UAV will depart from GPS guidance, and rely upon visual clues. The GPS data will still be used for checks on the behavior of the UAV.

## 6.2 Scenario Range

As the UAV starts to descend towards the Scenario Range, it will start capture and processing of image data at a higher resolution both spatially and in time. The image capture of time sequences is in this case done as short sequences of 5 to 9 images, or *bursts*, interspersed by still images at a rate of, say, two per second. The short, full time resolution sequence makes it possible to identify moving objects as opposed to stationary ones. On the other hand, the processing power required for such a short sequence is limited. The tracking and update of the moving vehicles is done using the intermittent sequence of still images.

## 6.3 Close contact range

In the *close contact range*, the UAV is in the air, but at relative low altitude and close to the objects of interest, near and around obstacles. It may be in the process of takeoff or landing, or it may be maneuvering in order to obtain a good view of a ground object. In this mode, the UAV will be able to model the three-dimensional structure of a scene to the order of 10–30 m in size, and specific objects in the order of 2–10 meters. Path planning in the presence of obstacles (with many similarities to conventional robotic path planning) is also important.

#### 6.3.1 Scene and shape modeling

Since the size of the UAV limits the sensor configuration to only one or possibly two camera sensors in a single gimbal, scene modeling has to be made using a moving single camera, e.g., by means of motion stereo.

Two modes of 3D modeling can be distinguished. The first mode is modeling of static objects, e.g., buildings, roads, etc, which are stationary and generate an apparent image motion proportional to their corresponding distance as soon as the UAV moves. The second mode is modeling of dynamic objects, e.g., cars, and which can be done only if estimates of their motion relative to the UAV can be computed. The two modeling modes may share a large subset of functionalities but need to be considered as separate, and the scene has to be segmented into parts or objects which are either stationary or dynamic and modeled using the corresponding mode.

## 7 Functionalities required

A number of system level functionalities have been identified.

### 7.1 Estimation of camera position and orientation

The UAV will have components which provide the position and orientation of the camera, e.g. by means of GPS, inertia sensors, and angle gauges. However, doing geo-referencing (establishing correspondence between image and GDR data) requires a higher accuracy than these components can provide. A solution is to use visual feedback to enhance the estimated position and orientation of the camera.

An enhancement procedure of the camera parameters based on visual feedback can be implemented in different ways. One way is to use a reference in terms of a orthographic image, transform it according to the first order estimates of the camera parameters, and then compare the result with what is seen in the camera. This step can generate corrections to the camera parameters, which then are updated, and by iterating this operation a few times, a sufficiently accurately estimate of the camera parameters is obtained.

From a practical point of view, the above procedure presents at least two issues which has to be dealt with. First, it assumes that these exists *one* reference image for any specific area that can be brought to correspondence with the camera image of that area. This may not necessary be true if the ground, as seen from above, have different characteristic depending on time of day, time of year, etc. This can be solved by storing a set of images for each area which cover the relevant characteristics (i.e. different time of day, different time of year), but this raises the question of how to manage the selection of different images in a robust way. Second, it makes no interpretation of local features other than, e.g. edges, which means that the two images must be compared or correlated in their entirety. Only dedicated hardware implementations can handle such operations in reasonable time.

An alternative solution to the geo-referencing functionality, that also solves the problems which are indicated above, is to first extract local features of sufficiently high complexity, in both the camera and reference data, and then to compare or correlate these feature descriptors. Example of such features is rotational symmetry, which typically detects crossings of roads, and high curvature which detects sharp bends in roads. The interesting property of such features is that they detect points which normally are sparsely distributed over the image while at the same time they are sufficiently many to allow an correlation operation between camera data and reference data to produce useful results. Consequently, it is much faster to correlate these feature points than entire images. Furthermore, these features are much more stable over variations in the camera image due to different times of day or different times of the year compared to the images themselves.

## 7.2 Detection of objects

An important functionality of the vision system is to indicate the presence of certain objects in a region. From the system's point of view, this operation may be divided into two modes depending on the expected result:

- Detection of one or several objects of a particular object class. This implies that the system wants to find *all* objects which fit a certain description, e.g. all cars of a particular type or which travel along a certain road.
- Detection of a *single* particular object which has a known description or signature.

The distinction between the two modes is motivated by different modes of operation in the higher layers of the system. However, the two modes should be implemented by means of a single and general object detection functionality for the following reasons. First, the two modes expect a description of the object/objects to be found, and this description has to be "unsharp" or "fuzzy" in both cases. In the first case, the "fuzzyness" relates to the fact that we want to describe a larger set of objects in terms of a wide-range signature, and in the second mode it is because, in practice, we have to allow the object's current signature to deviate from a previously estimated signature. Note that, in general, the "fuzzyness" is larger in the first mode than in the second mode.

Second, the object detection will in both modes result in a set of zero, one or many matches relative to the given signature. In the first mode this is natural, since we expect to find a larger set of objects which fit a wide-range signature. In the second mode, we expect the situation to be such that exactly one match occurs relative to a narrower signature. However, if the wanted object is not present or there are similar objects in the same region, zero or multiple matches have to be reported.

Consequently, a single function which detects a set of objects given a signature should be implemented. This function can then be used both for detecting a larger set of objects given a wide-range signature, or possibly a single object given a narrower signature. In both case, all objects which fit the given signature is reported.

## 7.3 Measurement of object properties

Given a specific object, typically a vehicle, the vision module must be able to measure a range of different properties, constituting a signature, of that object. Examples are

- Geometrical properties, such as length, width, area, shape. These should be view-based, i.e. described from the view of the camera.
- Spectral properties such as color and IR-signature.
- Velocity on the ground.

All properties carry uncertainties which must be appropriately represented, e.g. by means of fuzzy sets.

## 7.4 Search for object

Search for an object is initiated by the reactive layer which sends a command to the *Dynamic Object Repository* (DOR), where it evokes the object in question. An object has a contextual frame which is partly general and partly instantial. A general contextual property of a car is that it is located on what is recognized as a road. An instantial property of a car can be that it was observed at a particular position 3 seconds ago, and that it is red.

The degree of specificity of the command will determine the size of the contextual frame. “Search for a car” will involve a wide frame, location of roads upon which cars can be located, etc. “Search for red car R”, evokes a smaller and more specific contextual frame. It’s estimated position is used by the *Image and Scene Geometry Manager* to compute orientation and zoom setting of the camera given the state of the vehicle.

## 7.5 Backtracking in the object hierarchy

If the system is unable to locate the car as demanded, or some form of ambiguity arises such that the confidence measures are not accepted, it is necessary to backtrack. This is done according to the structure of the object hierarchy. The attention goes to the next level encompassing frame, which is **road section**. The road section associated with the assumed position of the missing car is activated. This causes the *Image and Scene Geometry Manager* to compute orientation and zoom setting of the camera given the information of the new frame.

Every interval of the object hierarchy is associated with a set of procedures.

## 7.6 Tracking

Given an object which is identified on the ground, i.e. its current position is known, the object can be tracked by centering either the camera or a region of interest (ROI) on that object over a period of time. This functionality may be implemented with different levels of complexity.

- **Low-level tracking**. Needs only the detection of a point in the image, e.g. by detecting motion or some other property which is easily measured in the image. By means of a Kalman filter, the image position of the point can be used to control the camera or a ROI in order to center on the corresponding object.
- **Medium-level tracking**. Based on a low-level tracking, the system can simultaneously measure a signature of the tracked object. The tracking supervisor ensures that the signature is stable over time, e.g. that one and the same vehicle is tracked. If the signature changes radically over time, this is reported to higher levels of the system.
- **High-level tracking**. This operation includes high-level support in terms of detection and recovery of short-term occlusion. A priori information, typically

from a GDR source, is used to predict visibility of the tracked object, as well as where and when the tracking can resume if occlusion occurs.

## 7.7 Anchoring

Given a position in an image, e.g. of a visible object, the system needs to anchor the image position to geographical coordinates or to information related to the geographical coordinates, e.g. static objects such as roads or buildings.

There are a number of ways this function can be implemented. One way is to assume that the ground can be approximated as a flat surface, and that the lens distortion is either negligible or corrected for. As a consequence, the mapping from 2D ground coordinates to 2D image coordinates (and vice versa) can be described in terms of an eight parameter bilinear function. This function depends on the camera's position and orientation, and these vary over time, which implies that the eight parameters have to be estimated whenever the mapping is needed. The estimation can be done by tracking a number of landmarks on the ground, which are clearly visible and have known world coordinates. Theoretically, four landmarks suffice, but at least twice or more are required to get reliable estimates, given that one or more landmarks may be invisible for a number of reasons, and that their positions in the image are not been accurate. Given that the eight parameters have been estimated, any image coordinates can be mapped to the corresponding ground coordinate.

It should be noted that this implementation assumes a flat ground, which may be a reasonable approximation in the landscape range. However, the closer and the more hilly the ground is, the less valid is the assumption. Furthermore, the implementation relies on the existence of a sufficiently dense grid of landmarks which have ground coordinates determined with sufficiently high accuracy. Finally, even though ground coordinates can be determined with sufficient accuracy for any set of image coordinates, the ground coordinates have to be related to data in the GDR database which is an additional operation not included in this discussion.

Another way to anchor image points to GDR data is to assume that the GDR data have a full 3D representation of all static objects, e.g. roads, buildings, etc. Given the camera's position and orientation, it is then possible to render an image representation of the GDR data, as seen from the camera's point of view. The result, a *virtual image*, contains in each pixel a representation of GDR data, e.g. a reference to a static object. In practice, this implementation can be made more efficient by computing this information on demand, i.e. only for those image coordinates which are of interest, rather than creating a full virtual image. It should be noted that this implementation of the anchoring functionality does not make any assumptions regarding the ground surface, but instead assumes accurate estimates of the camera's position and orientation, which can be accomplished according to the discussion in section 7.1.

# 8 System Strategies

A number of particular system strategies are employed.

## 8.1 Actively Controlled Image Capture

For identification of events of interest, a number of different sensing modalities are used. They include conventional color TV cameras for common day light visual orientation and object identification purposes. They also include extended exposure time arrays for night vision, and infrared light range cameras for detection of heat emission, or generally to give a temperature image to relate to the visual image. Very sensitive night vision sensors may be useful for many situations. It may also be advantageous to use radar information for certain aspects of navigation and identification. The definitive sensor setup will depend upon the tasks expected from the autonomous aircraft. The following discussion of active sensor control will apply more or less to all types of sensors, as it is presumed that the spatial information they provide can brought into a common representation or format which can be handled by the system. It is however anticipated that visual sensing will be the initial default variety, and the ensuing discussion will be oriented towards this.

A traditional approach in robotics has been to use video camera sensing. This is generally suboptimal for most situations, and a more flexible and controllable arrangement is desirable. There are a number of requirements on the sensing process:

- *Wide angle of view*
- *High spatial resolution*
- *High time resolution*
- *High light sensitivity*

In order for the system to obtain sufficient overview for navigation and localization of objects, it is necessary that the sensing is made over a sufficiently wide view angle. To allow robust identification of objects and details, it is necessary that the sensor provides sufficient spatial resolution. Interpretation of dynamics in scenes, such as motion of objects, requires a sufficient time resolution or frame rate.

Fortunately, all these requirements are generally not present simultaneously, a fact which relinquishes not only the demands on the sensing system but furthermore on the subsequent processing system. This illustrates the usefulness of a sensing camera with a number of controllable properties:

- *Controllable integration time to adjust exposure*
- *Controllable frame rate to adjust time resolution*
- *High resolution sensor element*
- *Region of interest (ROI) control of readout from sensor*
- *Variable zoom optical system, or multiple focal length systems*
- *Fast control of camera visual field orientation*

Not all variables are independent, such as the integration time and the frame rate.

A typical sensing procedure may be that the camera moves around mechanically, while images of wide angle resolution are recorded. There is generally no particular reason to have a high time resolution for this case. As the system detects something of interest it will orient the camera to that direction and zoom in on that part. If it is needed to measure motion, the sensor will increase the frame rate accordingly.

What is important for this process to be useful is that it goes very fast. The visual field of the sensor will "jump around" in the scene, switching between different regions, objects and aspects, very much like we do with our head and eye movements. This allows us to obtain a sufficiently high resolution in all important aspects, without flooding the subsequent processing system with redundant information.

Sensor elements and other components having the desired properties are available today, but what is required are the control mechanisms which allow close integration with the processing structure.

### 8.1.1 Capture of landscape range overviev images

For the landscape range mode, only still images will be captured, as no information about motion is assumed to be required here. Images are assumed to have short enough exposure time to avoid blur due to motion of the vehicle, vibrations etc.

Depending upon the quality of the attitude data from the vehicle, it may me necessary to capture images sideways to get estimates of the horizon. This is essential for the subsequent estimation of perspective transformation of captured images.

### 8.1.2 Capture and Stabilization of Image Sequences

For the Scenario Range and the Close Contact Range modes, estimation of motion of objects is essential. Rather than using continuous full video sequences, which would give huge amounts of data clobbering subsequent processing stages, motion is estimated for limited time frames. In this case short sequences are used, containing 5 - 9 images. This corresponds to a time window of 200 - 360 msec. A car moving at 60 km/h will during this time period move 3.3 - 6.0 m. See Figure 2.

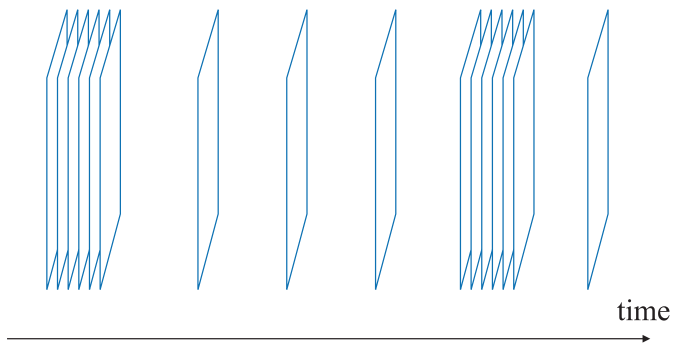

Figure 2: Illustration of burst mode image sequence capture.

These sequences are in addition captured for limited size regions, such as a road, in order to limit the processing power required. This implies that measurements are taken for some time duration, and there is a risk for blur due to vibrations and changes of orientation during the period of capture. It is not clear at this time what the quality of the data from the cameras of the helicopter is going to be, but a number of possible remedies are foreseen if required:

- The camera will be controlled to compensate the motion of the landscape on the array.
- It may turn out necessary to register individual images in a burst sequence to each other. This can be made checking for motion in the region surrounding the detection region, e.g. motion of edges of the road, motion of other surrounding structures.
- In the case of strong motor vibrations transmitted to the camera pod, it may be possible to sense these using a simple accelerometer, and use this information to capture frames in the same phase of the vibration period.

The burst sequences are interspersed with single still images at a rate of, say, two per second. These images are analyzed for objects at the position predicted from earlier measurements. If we assume that one burst and still image period is 3 seconds, this will contain on the average 13 frames. This is a reduction of data to 17%, but higher reductions seem feasible. If we assume that the region of interest is limited to 25 % of the area, the data rate is reduced to about 4 % of the full frame data rate.

Expressed in terms of speed-up, this is a factor 25 compared to full rate video. The fast filters used are estimated to contribute with a factor of 10. Special purpose hardware may contribute with a factor of 10 - 100. The total expected speed-up will then be a factor of 2500 - 25000.

## 8.2 Division into Processing Paths

It has been decided to try to avoid a specific stabilization of the initial image sequence from the camera, to avoid time consuming computations. It now appears possible to utilize gyro stabilized camera heads, which provide sufficient initial stability of image data, to allow use of separate procedures with sufficient robustness, such that the computation of each representation can deal with its own stabilization. It has consequently been decided to divide the computation into two different parts, which have different requirements on the processing strategy:

- **Stationary Scenes** These are overview scenes of the landscape which are necessary for orientation, recognition of landmarks, navigation, recognition of objects, recognition of roads and other traffic structures. These scenes are typically taken as a snapshot with a low or medium resolution over a large area. The scenes will typically contain a large number of pixels. As the scenes are taken as single snapshots, there is no need to stabilize the image to get sufficient sharpness in stationary features on the ground. As new images are taken relatively infrequently, there should be plenty of time to process these snapshots for detailed identification of properties.

- **Dynamic Scenes** Dynamic scenes will in our case consist of moving cars on the road. It will be required to keep track of individual cars and measure their velocity, as well as velocity flows of sets of cars on certain parts of roads. The intention is to employ a model restricted Kalman filter for this purpose. The filter will receive raw, noisy estimates of velocity for each car. The filter will also get data about motion of adjacent structures, in order to compensate for camera movement.

## 8.3 Processing of Stationary Scenes

The purpose of processing of stationary scenes is as stated earlier to generate features necessary for orientation, recognition of landmarks, navigation, recognition of objects, recognition of roads and other traffic structures. These have to build up some knowledge data structure about the environment traversed by the air vehicle. These properties are as well to be compared to references in some form of GIS data structure. It would be very inconvenient to have this information stored in iconic form as images. It will be necessary for several reasons to appear, that the information is represented in some semantic, or partially interpreted form.

Images will be single, still, medium resolution images over a region, containing a large amount of data. They may be in color or in gray-scale, dependent upon the type of sensor. Special types of spatial sensors are possible, such as infrared cameras or some type of radar. The representation used must allow the comparison and fusion between several different modes of imaging.

An image will be subjected to interpretation using the conventional multi-resolution machinery available. The first step is to provide a vector or tensor representation of orientation structures. This can subsequently be processed by a cooperative enhancement procedure, to emphasize lines and to remove noise. Next, the orientation transform will be subjected to analysis for representation of curvature and corners. All these representations are so far in a continuous form defined over conventional arrays. See Figure 3, left side.

As stated earlier it is inconvenient to have this information stored in iconic form as images. It is preferable to have the information represented in some semantic, or partially interpreted form. This gives potential advantages of being more compact, as well as it provides advantages for fusion, search and relation to other references. The three levels of features, including color if used, will be thresholded and discretized, and represented as a local data structure of the particular image under consideration. See Figure 3, right side.

These three levels of features, will be the features to store and use for comparison with earlier derived data.

## 8.4 Integration of image data into the resident data bases

A single snapshot image will give a limited patch view of the environment. Consequently, it will be necessary to take repeated snapshots, as the air vehicle moves over large areas, to keep track of the movement of the vehicle. As a consequence it will be necessary to relate and fuse subsequent representations of this stationary

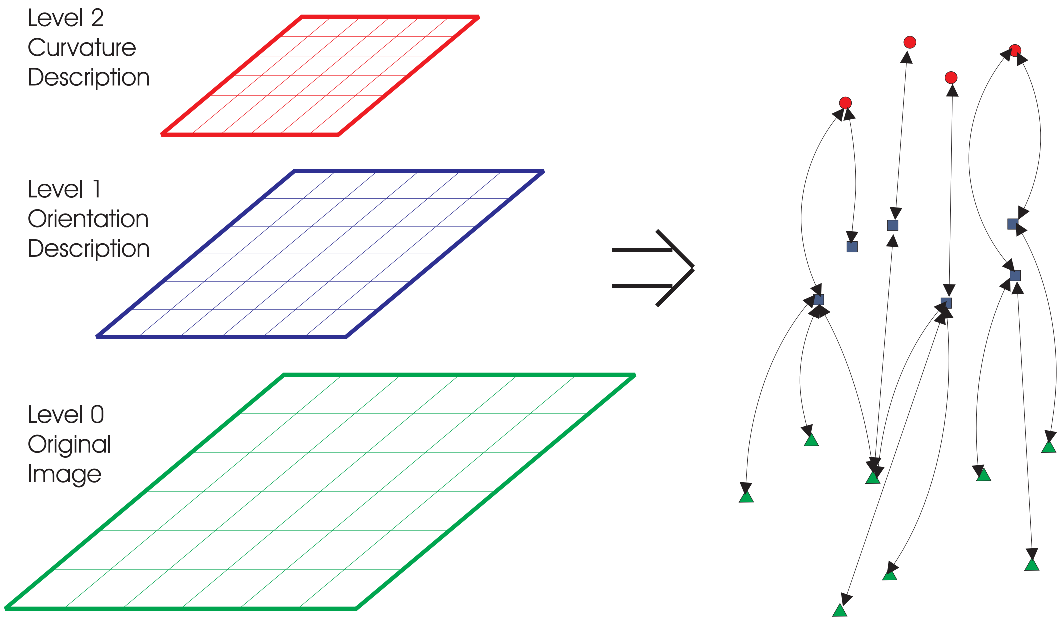

Figure 3: Intuitive illustration of transition between hierarchical array structure and linkage structure.

scenery.

It is assumed that adjacent patches have sufficient overlap, such that they can be related to each other, and to data already available in the resident data bases. This will be the case both in relation to the Dynamic Object Repository (DOR), and to the Geographic Data Repository (GDR).

This is done through matching and fusion with data in the resident data base. See Figure 4. This matching will start with higher level curvature and corner features, as they are assumed to be more sparse, and more significant. Matching then propagates downward, to employ line and edge features. After this fusion, the data is permanently linked into the resident data structure. In the process of matching and linkage, issues of scale and perspective have to be taken into account and normalized for. It also requires knowledge about position, orientation and attitude of the airplane. It is believed that such a symbolic representation level is a far better level for comparison with stored data, than to make the comparison at the image level, and consequently to produce features at image level from the data base.

The information so derived will as well be used to identify regions for roads, where a dynamic scene analysis will take place. These regions will be supplied as ROI-regions (Region Of Interest), to decrease demands on computation power in the dynamic scene analysis.

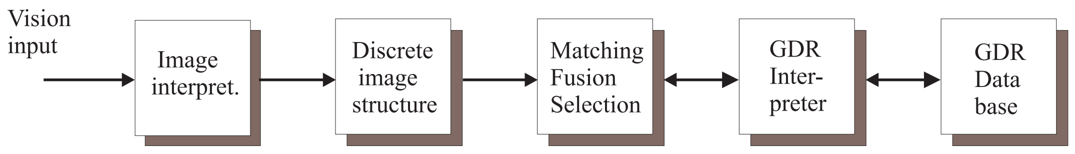

Figure 4: Block diagram of interaction between vision system and GDR structure.

## 9 System components

The functionalities just discussed are implemented by different system components, cooperating in various ways.

### 9.1 Low Level Feature Computation

Images from the capture are transferred to a temporary storage, after which they are processed to generate features. The feature computation is generally assumed to be performed by a special purpose hardware processor, but there will also be versions which can run on standard general purpose computers. We distinguish between the computation of *static* and *dynamic* features:

#### 9.1.1 Computation of static features

Static features are computed from still images:

- Color of regions
- Orientation structure
- Complexity or curvature, as defined by divergence/rotation descriptors

**Color** is measured by averaging over regions of the original color image, over neighborhoods of different sizes. For IR camera images, the color information is substituted by intensity, which is an absolute measure after standard calibration of the sensor.

**Orientation structure** is computed by convolution of the luminance input image with a set of orientation filters in different scales. The output consists of filter responses for odd and even filters in different orientations and scales for every neighborhood of the image.

**Divergence and rotation descriptors** are computed by convolution of the orientation structure image with a set of filters in different scales. The output consists of filter responses for divergence and rotation in different scales for every neighborhood.

### 9.1.2 Computation of dynamic features

Dynamic features are computed from burst time sequence volumes consisting of luminance images. Motion flow is computed from the burst sequence regions. Such a burst time sequence contains a number of images corresponding to the depth of a filter kernel in the time domain, which may be from 5 to 9 elements, dependent upon the desired space-time domain properties. Such a burst sequence consequently gives only one motion estimate over the burst.

The output consists of filter responses for odd and even filters in different orientations of planes for every neighborhood of the time sequence volume.

## 9.2 High Level Feature Computation

The low level data from the *Low Level Feature Computation* is employed to produce high level descriptors, which can be used for interpretation or matching.

### 9.2.1 Orientation descriptors

Filter responses are combined to form either of:

- Mutually inhibited pairs of line/edge statements in scalar form for different orientations and positions in the image.
- Combinations of line/edge statements for different orientations in vector form, for all positions in the image.
- Combinations of line/edge statements for different orientations in tensor form, for all positions in the image.
- Mutually inhibited pairs of divergence/rotation statements in scalar form for different orientations and positions in the image.
- Combinations of curvature and color in an image.

### 9.2.2 Image motion estimation

Estimation of local image motion is a useful operation, e.g., in order to detect moving objects, or determine 3D shapes. In its simplest form, the estimate may be in the form of a two-dimensional vector, which represents direction and speed of the corresponding local image motion. However, estimation of image motion is an operation which has an inherent property of uncertainty or undeterminability, i.e. reliable motion estimates cannot be produced in certain parts of the image, e.g. homogeneous regions without texture, and for linear structures only normal velocities can be determined. To manage uncertainty in a robust way, more complex representations than two-dimensional vectors are needed, e.g. tensors.

It should be noted that most subsequent operations which rely on robust motion estimates do not have to be executed at normal video frame rate (25 frame/s). Consequently, the motion estimation can operate in "burst mode", i.e. compute motion estimate from a short sequence of images taken with normal frame rate where each sequences is generated with low frequency.

### 9.2.3 Camera motion estimation

The motion of the camera, both the translational and rotational components, have to be estimated with high accuracy. The primary position instrument of the UAV is GPS or DGPS. This instrument does not provide the necessary accuracy or frequency of the UAV's position in order to produce motion parameters by means of differentiation. Instead inertia based instruments shall be used to give the camera motion. It should be noted that it is the motion of the camera, not the platform, which is measured and input to the system.

## 9.3 Image Data Classification, Matching and Fusion

This unit will handle the comparison between different varieties of image information and references. We can observe a number of different functionality modes:

1. Navigation through comparison between observed image patches and a reference
2. Fusion between different imaging sources
3. Identification of context
4. Identification of object

There are two fundamentally different strategies which are used in the process to identify a particular phenomenon: *Matching* and *mapping.* It can be mentioned that humans likewise employ two different strategies for comparison of structures, such as reading of a map for the purpose of orientation, and for direct recognition[1].

1. **Matching** is preferable for the case of finding the direct correspondence to a complex structure which does not immediately associate to a particular position or context. As an example, we can take the patch derived from a camera view, to be matched to a reference image of the ground or a data base. In this case it is an object which is quite predictable with respect to its structure, scale, orientation, position and transformations, as we have earlier, adjacent samples from the reference. On the other hand it may be a one-of-a-kind structure, for which there is no reason to generate a model which directly ties it to its context.

2. **Mapping** is preferable for the case of recognizing a member of a large set of possible objects, which may be located at unpredictable positions and in unpredictable states of scale, orientation, position, and perspective transformations. This is typical for most cases of what we traditionally denote *object identification.* The structure has to deal with a large space of possible outcomes, even if this can be reduced due to contextual restrictions. Due to the large space of outcomes, it is often not possible to match individually to such a large set of models. Rather, models have to be assembled from fragments or primitives, each determined as mappings from the feature space.

Before we go into the specifics of the different functionalities, we will discuss some features which are common to all functionalities in matching and mapping:

- Normalization to observed view
- Successive top-down matching
- Use of flexible and context controlled models

### 9.3.1 Normalization of transformations to predicted observed view

The principle used in this system is that matching is made in the perspective assumed for the observed view patch or object observed, and this predicted perspective and other transformations are applied to the reference view to render it into the same form as the observed. The reason is that the world will not appear as a

2-dimensional sheet, in particular at a short distance from ground. The view obtained at a low altitude will depart considerably from a perspective transform of an orthonormal reference version of it. In a perspective view, certain objects may be hidden by others or the ground altitude profile may vary. Given knowledge of the altitude profile, this can be applied to the reference data to provide a good estimate of a perspective projection view. This application of altitude data could in principle be applied to normalize the observed view according to predictions, but there will be an additional noise or ambiguity due to compounded uncertainties and combinations thereof. It is conceivably worse to apply an erroneously selected altitude perspective transformation to the observed image patch and compare to the likewise erroneously selected reference scene, than to compare the observed view to an erroneously selected reference view, which however has an appropriate altitude dependent transformation.

This normalization principle goes well with a consistent use of *view centered* representation, something which is assumed for the development of extended functionalities, in particular the representation of 3-dimensional objects. For objects there may be no such thing as a canonical representation to which observed views can be related, and the knowledge of the object may be incomplete in that only certain views are available for observation or for generation of a reference.

It should be emphasized that the normalization mode proposed is for computation and comparison of spatial data. This does not prevent data in symbolic form from being represented otherwise, such as in a global orthonormal system. It is important that representations are in the form which allow fast and reliable computations. Transformations can be made between these and other systems using the *Image and scene geometry manager*. The preceding is also similar to what is known about the ways the visual system works.

### 9.3.2 Successive matching in abstraction and scale pyramid

The functionalities of comparison, fusion, stereo matching, recognition, etc, imply some form of matching or correlation between at least two data sets. Although the generation of the two sets may have worked out well with preliminary alignments and transformations, there are generally remaining uncertainties with respect to the following variables:

- position
- scale
- orientation

A matching procedure implies in principle that all these parameters are varied within their uncertainty range, within which the best match is determined. As the two sets may contain large amounts of data, the computation work may be large. Given suitable sparse information representations and proper strategies, procedures can be speeded up radically.

After the tentative alignment involving various transformations discussed elsewhere, the two image patch representations are compared for match. This is done in terms of the descriptors used, rather than the original images.

The alignment is made hierarchically, and in the order:

- Curvature and color or intensity
- Orientation

These features are selected for their sparsity and their specificity. Within a given size patch there are generally only a few significant statements of curvature. This means that an alignment can be made in a coarse scale with few points.

Tentative matches using these characteristic points as guides are performed, where additional features are used to either support or reject a particular match hypothesis. This information is used to perform an improved realignment of the image and the reference, after which another match can be performed. Finally the updated set of transformation variable can be computed.

### 9.3.3 Use of flexible and context controlled models

The preceding discussion only assumes a matching with a single, rigid template. In reality it is necessary to match against a large set of possible models as well as to use contextual information to reduce the space of possible models in the actual situation. This implies that the matching is performed as a mapping onto the outcome space, given a set of contextual parameters. The output map contains three different categories of parameters:

- Realignment parameters
- Contextual state parameters
- Object state parameters

### 9.3.4 Motion Tracking

The motion statements are used to:

- Identify the existence and the position of the object from the motion statement.
- Compute the object outline from the motion statement.
- Check if a data record is defined for the object, and otherwise define a new data record.
- Use the motion information and earlier prediction to update information about object.
- For each object compute and list characteristics:
  - Position
  - Velocity
  - Color
  - Type

### 9.3.5 Navigation through comparison between observed image patches and a reference

As the vehicle moves around it will have a current estimate of its state, which implies:

- position in x, y and z
- attitude angles of vehicle
- orientation of camera(s)
- zoom setting of camera(s)

This information is taken partly from the GDR system and partly from prediction using earlier measurements. Using this data, the *Image and scene geometry manager* will compute regions of interest for further image acquisition.

## 9.4 Dynamic Object Repository

The dynamic object repository (DOR) not only contains information about objects that can be encountered by the system. It also acts as an interface in this domain between the visual or spatial-cognitive side and the symbolic side of the system structure. See Figure 1.

The general properties of an object description appears from Figure 5. The description of an object has two sides, one symbolic and one visual. They correspond essentially to what is known as *object-centered* and *view-centered* representations. The symbolic representation is relatively invariant and suited for generalization and reasoning. The visual representation is less general and aspect or view dependent, which however allows it to perform very precisely in estimation of object properties as well as system state properties, and in the guidance of actions.

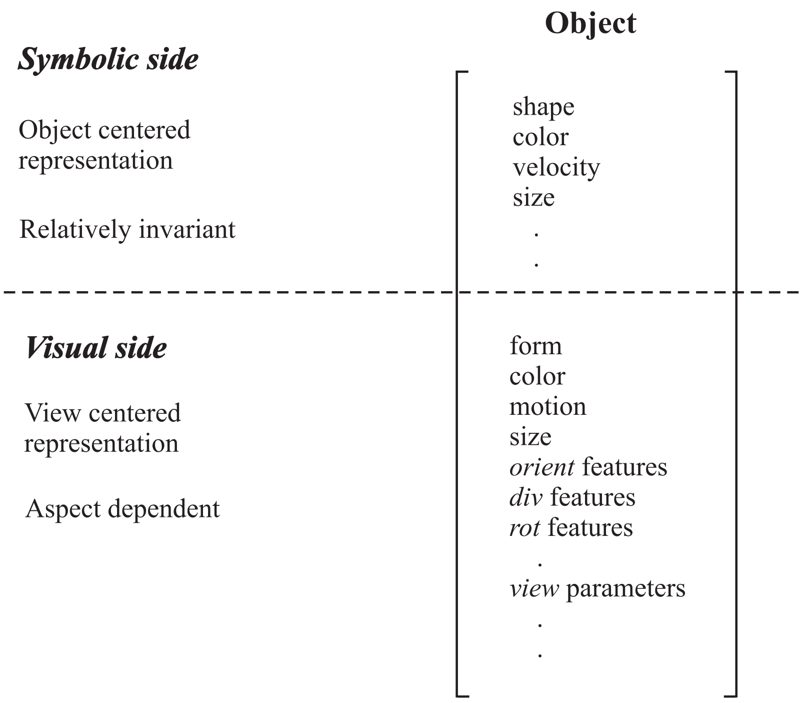

Figure 5: Example of properties of object in Dynamic Object Repository

There will in some sense be two sets of descriptions of an object, which however are strongly linked.

On the visual side, object properties are represented as the set of features corresponding to the particular object, given the actual view or observation state. For a view centered representation, the aspect or view state is part of the object description, as the set of descriptors derived will depend upon the actual view.

On the symbolic side, object properties are represented as a set of variables, which may in turn consist of other variables, along the lines of traditional data structures.

While the variable on the symbolic side may be `red`, such as for `find red car`, the representation on the visual side will be in color coordinates as they are measured from the object. In more sophisticated implementations, it will also take into account the illumination, shadowing, and other variations due to actual geometric setup, etc. The variable `red` will imply different things if it is about a car or if it is about a tile roof, and the linkage structure within the object description shall implement the transformation required.

The variable **size** on the symbolic side may be used to distinguish between a small car, a large car, a bus, a truck etc. The variable **size** on the visual side may be in terms of number of pixels, give a certain state of view parameters. Similarly about other parameters.

The degree of specification and parameterization of variables on the symbolic side can obviously be defined and controlled as desired. An important principle is however that the symbolic representation shall be sufficiently invariant, to allow simple specification of essential variables and parameters. The detailed handling of features and geometry requires a close integration of transformation procedures.

### 9.4.1 Shape modeling

The actual shape modeling which operates on image motion data or depth maps must of course be designed with care. However, it should be emphasized that the quality of this step to a large extent relies on the quality of the functionalities mentioned above. Furthermore, shape modeling techniques are still in the scope of current basic research which implies that there are no established standard techniques which have proven valid for general scenes. For simplified scenes and objects, there exist techniques which segment the image into regions, where the image motion field is consistent with a particular motion model. The motion model is derived from planes or conics. The available methods for establishing these regions are fairly complex and computationally demanding since the segmentation is iterative, e.g. using region growing techniques. A technique of this type has already been developed at CVL.

An alternative to the above mentioned technique, is to make explicit the depth information in terms of homogeneous position tensors. Local operations on such tensors can then be used to determine local shape descriptions, e.g. of planar surfaces and lines, which again can be represented in terms of tensors. By combining tensors of this type, in particular by means of tensor products, it is possible to represent convex objects like polyhedras and ellipsoids. Such a representation can be obtained by means of local operations on increasingly coarser scales, i.e. the more complex structure the tensors represent, the sparser can the representation be. An approach of this type should be much more efficient in terms of computation compared to simple region growing techniques.

The suggested approach is based on a tensor formalism which is well-established at CVL, and extends it into the domain of projective spaces which also is a well-established area in computer vision. The result is an approach which allows reasoning on uncertain or partial representations of lines, planes, etc.

## 10 Example of Processing Results for Navigation

For the related tasks of matching and recognition, it is necessary to use sequential and hierarchical structural procedures, rather than parallel statistical procedures of classical estimation type. The reason is that we will suffer great penalties in computation, if we can not keep a limited dimensionality of the data set as we go about the processing. It is also well known that humans use relatively few, but more reliable clues to sequentially build up the knowledge of a situation. Limited knowledge and limited aspects are in addition fundamental obstacles, which prevent the use of any global methods.

In this section we will look at the procedures for fast identification of a particular object. See the road section of Figure 6.

### 10.1 Orientation

In Figure 7, a description of orientation is illustrated, using a vectorial combination of filter outputs for different orientations[5]. An important property is that orientation is a *double angle feature*[5]. The reason being that the same orientation returns after 180 degrees. Use of orientation features provides an improved discrimination at higher resolutions.

### 10.2 Curvature and Rotation

The orientation information described above can be used to produce statements about curvature or rotation[7, 8]. To recapitulate, we have the angular dependencies

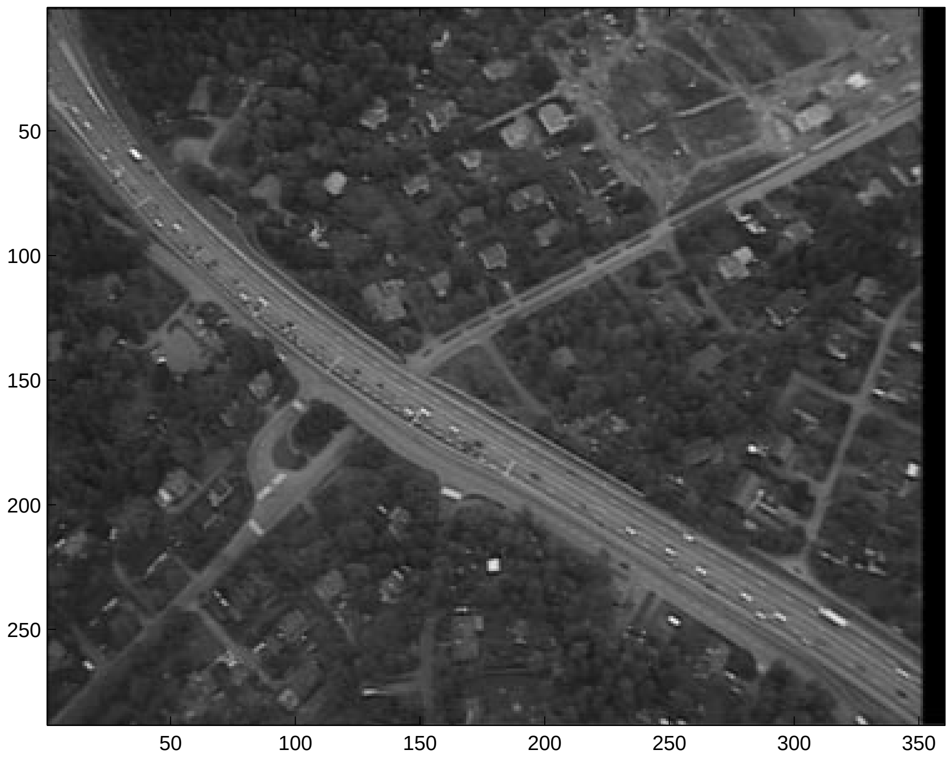

Figure 6: Image of road section

as follows:

|  |  |
|---|---|
| • Orientation | $2\phi$ |
| • Curvature | $\phi$ |
| • Rotation | 0 |

It should be observed that a characteristic object of rotation such as a circle, is scale dependent, while a corner is essentially scale independent. The corner itself is fully scale independent, but it is generally part of some object of limited size, which indirectly imposes a restriction with respect to scale.

A successful use of rotation requires more attention to proper scale, as well as a use of multiple scales. Given proper selective mechanisms for this, the rotation feature should be useful.

In Figure 8 is illustrated the computed curvature using *divcons* from the orientation transform in Figure 8. The operation *divcons* computes the normalized divergence from an orientation transform image[7, 8].

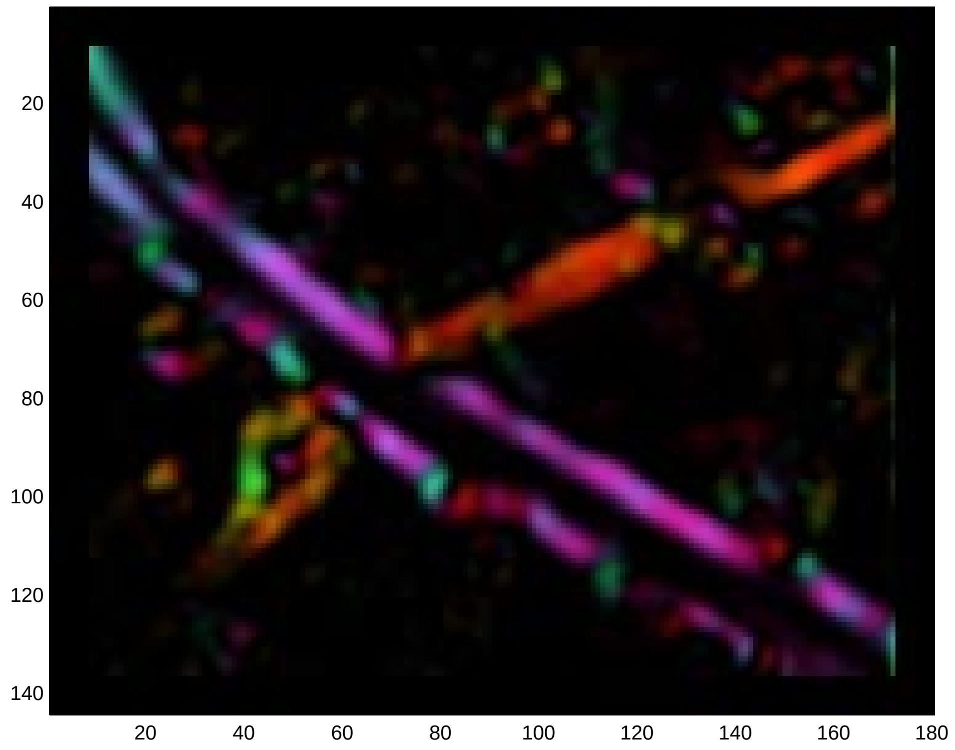

Figure 7: Orientation transform of image in Figure 6, using low frequency orientation filters

The vectors pointing in the direction of the angle, are displayed as color.

The information so derived can be used in a number of ways to match to a reference to establish a match. One such way is to obtain a description of features without the structural, geometric information. Figure 9 illustrates the transition from vector field patches to a compact, linear modular vector representation. Figure 9a shows the stylized result from running a curvature description operator upon some orientation image. This produces small patches of vector fields, illustrated in color, which give the orientation of the curvature and its magnitude. The average orientation of the vector field is indicated by the vector attached to each patch in Figure 9a.

A mapping is made from this 2-dimensional vector field, to a modular, one-dimensional scalar map. We can illustrate what happens in Figure 9b. Vector patches are brought over to provide a contribution to the scalar map at the position indicated.

The procedure intuitively described is equivalent to generating a histogram of the vector orientation distribution over the curvature vector field image. Every pixel position of the curvature vector field image gives a contribution to the proper angular window, with a value proportional to its magnitude.

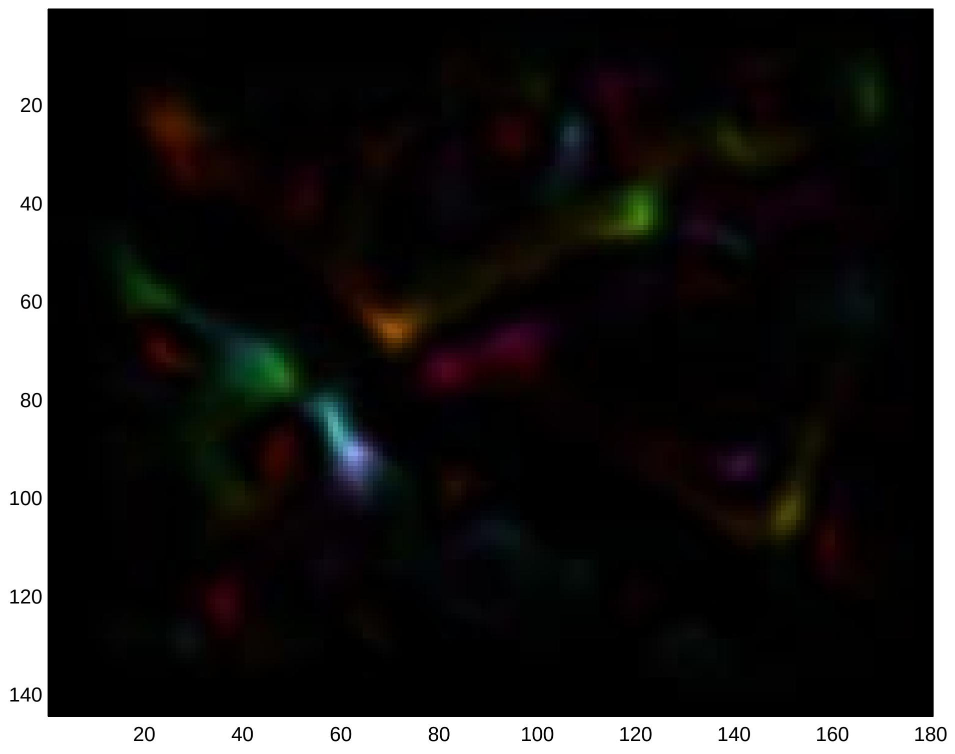

Figure 8: Result from computation of curvature using *divcons* on orientation transform in Figure 7

This produces a map of the feature distribution within the window under consideration, disregarding spatial position of these features. This in turn produces a representation which is invariant to position and scale, as long as the same features remain within the window. The representation is however dependent upon the orientation of the window.

This representation can be used for matching to a reference. If it can be assumed that there is no difference in orientation between actual sample and reference, it is sufficient to check the degree of matching or correlation with a single orientation of the actual sample. The matching can be made to a large set of reference structures, and the best match is selected.

If the relative difference between the orientation of the sample and the reference is unknown, either completely or within some range of uncertainty, it is necessary to displace the histogram of the sample within the actual range of relative orientation uncertainty, and compute the match for each displacement.

This computation performed upon the rotation transform in Figure 8 is illustrated in Figure 10. The particular structure matches uniquely at a particular point of the

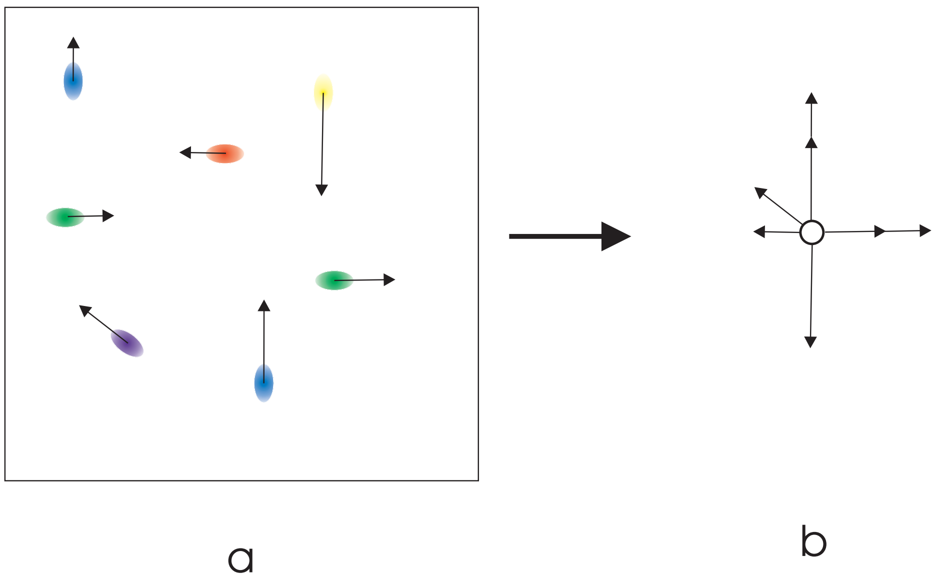

Figure 9: Illustration of curvature patches with dominant vector direction indicated a), and the one-dimensional scalar map produced b).

road scene.

## 10.3 Influence from perspective transformations

The preceding discussion assumes that both sample and reference are given in the same perspective transformation. This is normally the case, as the matching assumes a particular state of the reference, which has been used to compute the predicted perspective transformations. In a case where the observation angle and consequently the perspective projection is unknown, partially or totally, it is possible to recompute the effect of a modified perspective on the linear vector map much faster than it takes to recompute the perspective of an entire image.

This recomputation implies intuitively that the mapping onto a circle will be replaced by a mapping onto an ellipse. This is similar to the change of mapping obtained for computation of texture gradients. A deviation from the orthonormal projection will give a bias towards vectors along the perspective horizon.

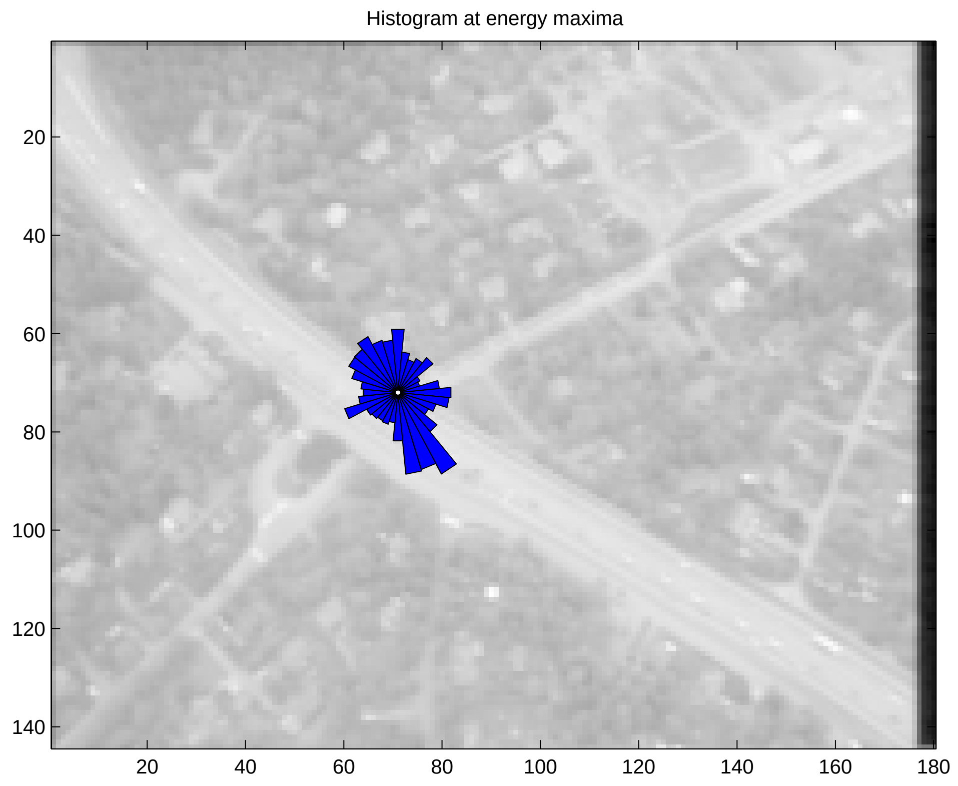

Figure 10: Illustration of feature vector histogram signature of structure at the point of largest feature energy density

## 10.4 Mapping window

Although the mapping is position invariant in itself, the mapping will depend upon what features are included within the window under consideration. It is assumed that the histograms forming the maps will be robust enough that they normally can retain sufficient uniqueness, although they may not contain exactly the same features. Sufficient uniqueness implies here that the number of tentative possible outcomes is limited, to allow more demanding computations to be made on a smaller number of alternatives.

As indicated earlier, sufficient uniqueness requires that the number of curvature patches within a window is limited. It is simply not possible to represent too much structure in a compact map. For that reason, it will be necessary to relate the window size to the structural complexity expected. This can e.g. be estimated from potential maps generated by the curvature descriptors.

### 10.5 Computational considerations

The preceding structure allows fast computations. A hierarchical arrangement of the matching is assumed, such that fast initial procedures are used to reduce the possible space of outcomes.

The initial matching is assumed to use the one-dimensional map of *div* distributions. This is a vector, which for the initial test may contain only 32 scalar values. This can quickly be matched to a large number of prototypes, with or without displacement to test for different orientations.

The best matches are then tested further involving information about color and orientation.

## Acknowledgements

The WITAS project is funded by a generous grant from the Knut and Alice Wallenberg Foundation, Sweden. Credits go to several people in the Labs of Linköping University, for discussions around ideas. Special thanks go to Björn Johansson for his implementation of the processing steps on the road scene.

## References

[1] D. H. Ballard. Animate vision. In *Proc. Int. Joint Conf. on Artificial Intelligence*, pages 1635–1641, 1989.

[2] P. Doherty. The WITAS integrated software system architecture. *Linköping Electronic Articles in Computer and Information Science*, 4(17), 1999. http://www.ep.liu.se/ea/cis/1999/17.

[3] R. James Firby, P.N. Propopowicz, and M.J. Swain. The animate agent architecture. In D. Kortenkamp, R.P. Bonasso, and R. Murphy, editors, *Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems.* AAA Press/The MIT Press, 1998.

[4] Erann Gat. Three-layered architectures. In D. Kortenkamp, R.P. Bonasso, and R. Murphy, editors, *Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems.* AAA Press/The MIT Press, 1998.

[5] G. H. Granlund and H. Knutsson. *Signal Processing for Computer Vision.* Kluwer Academic Publishers, 1995. ISBN 0-7923-9530-1.

[6] G. H. Granlund, H. Knutsson, C-J. Westelius, and J. Wiklund. Issues in robot vision. *Image and Vision Computing*, 12(3):131–148, April 1994. Invited paper.

[7] Björn Johansson and Gösta Granlund. Fast selective detection of rotational symmetries using normalized inhibition. In *Proceedings of the 6th European Conference on Computer Vision*, Dublin, Ireland, June 2000. Accepted.

[8] Björn Johansson, Hans Knutsson, and Gösta Granlund. Detecting rotational symmetries using normalized convolution. In *Proceedings of the 15th International Conference on Pattern Recognition*, Barcelona, Spain, September 2000. IAPR. Accepted.