Vision for a UAV helicopter

Klas Nordberg, Patrick Doherty[†], Gunnar Farnebäck, Per-Erik Forssén,
Gösta Granlund, Anders Moe, Johan Wiklund

Computer Vision Laboratory, Department of Electrical Engineering, Linköping University, Sweden
[†] *Knowledge Processing Laboratory, Department of Computer Science, Linköping University, Sweden*

Abstract

This paper presents and overview of the basic and applied research carried out by the Computer Vision Laboratory, Linköping University, in the WITAS UAV Project. This work includes customizing and redesigning vision methods to fit the particular needs and restrictions imposed by the UAV platform, e.g., for low-level vision, motion estimation, navigation, and tracking. It also includes a new learning structure for association of perception-action activations, and a runtime system for implementation and execution of vision algorithms. The paper contains also a brief introduction to the WITAS UAV Project.

1 Introduction

The WITAS[1] Unmanned Aerial Vehicle Project[2] is a long-term basic research project at Linköping University (LiU), Sweden, which involves cooperation of different departments at LiU, as well as a number of other universities and companies. The project is currently in its second phase, terminating at the end of 2003, which is focused on integrating software and hardware systems developed in the project on a helicopter platform that will be used for demonstration of various aspects of the project.

One goal of the project is to demonstrate a fully autonomous UAV for applications such as traffic monitoring and surveillance, emergency services assistance, photogrammetry, and surveying. To meet this goal, basic and applied research has been carried out in a wide range of topics; a software architecture for deliberative/reactive behaviour, a helicopter flight control system, a task-based planning system, a chronicle recognition system for identifying complex vehicular patterns on the ground, geographical information and knowledge databases for on-board use, multi-modal interfaces (including dialogue) for ground operator/UAV communication, and an on-board image processing system. Most of these topics are or will be covered in other publications [1], and this paper focuses on the research carried out within the project by the Computer Vision Laboratory, LiU.

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

[2]This work and the project is funded by a grant from the Wallenberg Foundation.

2 The WITAS UAV

The UAV platform used in the project is a slightly modified Yamaha RMAX helicopter manufactured by Yamaha Motor Company and is commercially available in Japan as a radio-controlled platform for spraying pesticides on crops. It is approximately 2×1 meters, has a maximal payload of 30 kg, and can provide approximately 150 W of electrical power for the on-board equipment. In addition to the RMAX sensors included with the platform, a Honeywell HMR3000 digital compass, a static pressure sensor, a Boeing digital quartz INS, a temperature sensor for the PC104, a video link to the ground station, and a differential GPS have been added. The platform is evolving and additional sensors may be added in the future.

A PC104 board (Pentium P5, 266MHz) is currently being used as the heart of the control system which executes all real-time control tasks for both the helicopter and camera system and collects all available sensor data. The helicopter is equipped with a Sony FCB-EX470LP color composite video camera mounted on a stabilized gimbal with a pan/tilt interface which attenuates vibrations. Figure 1 shows the experimental UAV platform.

Clearly, the limitations in payload and electrical power provided by the helicopter impose restrictions on the hardware used on-board. In particular, this situation means that the computational power available for vision is much less than, e.g., a standard PC, and a substantial amount of work has been spent on customizing or redesigning existing vision algorithms for various purposes to fit the available hardware while still providing acceptable robustness.

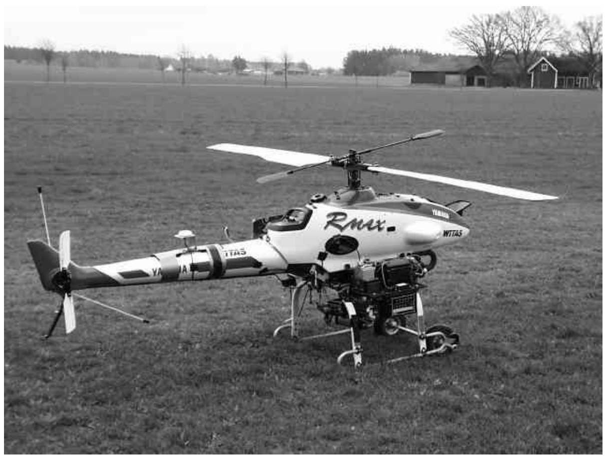

Figure 1: *The WITAS UAV Platform*

3 Software architecture

The software architecture of the system is characterized at a coarse scale as a three level system, with a deliberate system at the top, a reactive level in the middle, and a processing level at the bottom. There is also a set of knowledge databases which can be accessed from all levels, e.g., the Geographic Data Repository (GDR) that contains maps, aerial photos, and terrain models, and the Dynamic Object Repository (DOR) which typically contains dynamic information about ground vehicles. The various modules of the architecture communicate using CORBA, allowing system development and deployment on the UAV to be made in a highly flexible way.

The deliberate layer is devoted to planning, e.g., of mission tasks and flight paths, prediction, and chronicle recognition. The reactive layer includes various high level controllers, e.g., of the helicopter, of the camera, and of the image processing module, and is implemented in the language CONTAPS 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 communication channels to other parts of the architecture. The process layer is responsible for concurrent computations of feedback loops tightly coupling sensing with actuation.

The processing layer contains the image processing module (IPM) which is responsible for all low and medium level processing of images and image sequences. The IPM is based on a data-flow processing model, and contains a library of nodes which process data at various levels of complexity. By means of a simple API (IPAPI), a CONTAP can construct data-flow graphs, e.g., for tracking, motion estimation, etc, and use a runtime system to allocate resources and execute the graphs. The runtime system, IPAPI-Runtime, is developed within the project and is presented in more detail in section 10.

4 Operational scenarios

As mentioned above, the application area for the WITAS UAV is related to traffic situations, e.g., detection of specific events such as overtaking or U-turns, tracking of ground vehicles, and estimation of various features of a vehicle such as velocity and type.

A typical mission of the UAV could be to fly to a certain point close to a road, and then to hover and observe the road segment in order to detect a specific event. This could be that a certain vehicle appears on the road segment, or that a vehicle makes some type of action, e.g., stopping at a crossing. These events are then reported back to the ground operator.

A more advanced scenario could be that the UAV, once it detects a certain vehicle, starts to track the vehicle. This can be made both with the camera and by moving the helicopter. In both cases, the tracking requires a certain amount of planning to be made by the higher levels of the system, e.g., to predict where and how the vehicle is going to move on the ground, and to manage occlusions from buildings, etc.

A still more advanced scenario is to make the UAV assist a ground vehicle to either intercept a tracked vehicle or to get to a specific location in an urban area with traffic jams. The latter case requires the UAV to be able to detect and estimate the size of the traffic jams, and to find free paths around them.

In this and other scenarios, the UAV should be able to autonomously navigate between points, track ground vehicles, and estimate various features and detect events related to individual vehicles or pairs or even large sets of vehicles. All these tasks require various types of subtasks to be executed in the IPM, and the following sections present some of these subtasks which have been developed particularly for the WITAS UAV Project.

5 Fast filtering techniques

The more advanced IPM subtasks, e.g., landmark tracking and motion detection, require spatial or even spatio-temporal filtering of the camera images to be made. This includes filtering for detection of lines/edges and estimation of their orientation, or detection of corners and other local symmetries. One basic research topic of the project has been devoted to simplifying the implementation of such filters, resulting in a set of estimation method of local features which are highly efficient compared to previous implementations. This optimization is a prerequisite for allowing image processing to be made on the limited computational capacity available on-board the helicopter.

The method is based on the idea of approximating local image neighborhoods by polynomials, where the coefficients are determined from a weighted least squares problem. For certain classes of weight functions these polynomial expansion coefficients can be computed very efficiently by a hierarchical scheme of 1D filters, and the filter coefficients as well can be computed quickly in closed form, avoiding the need for any optimization procedures. The expansion coefficients can be used to construct various types of features, e.g., of local orientation, local symmetries, and motion [2, 7].

6 Motion Estimation

One subtask of the IPM is to detect moving objects on the ground. If such an object can be determined to be on a road, it is most likely a vehicle of some type, and is therefore of interest in the various scenarios presented above. However, estimation of local image motion is not enough to detect moving objects since the camera is moving at all times, adding a global motion to the estimated motion field. Instead, more or less standard techniques have been used for estimating the global motion field and subtract it from the estimated motion field it to get a residual field corresponding to moving ground objects.

However, the camera motion due to the engine vibrations and the wind are too fast to allow normal spatio-temporal filtering of the camera image sequence for estimation of local motion. Also, the temporal depth of the filtering has to be rather short to fit the available computational power. Therefore a two-frame multi-scale method for estimation of local image motion has been developed, based on the previously mentioned polynomial expansion of the image. By comparing the expansion coefficients of two subsequent frames it can be estimated how large the local displacements are. This works best when the displacements are small relative to the scale of the polynomial expansion but at coarse scales finer details are lost. The solution is to use a multi-scale scheme where the estimates are propagated from coarser scales to finer, thereby keeping the successive displacements small at all scales. The result is a dense motion field which can be computed with a reasonable tradeoff between efficiency and robustness [3]. See figure 2.

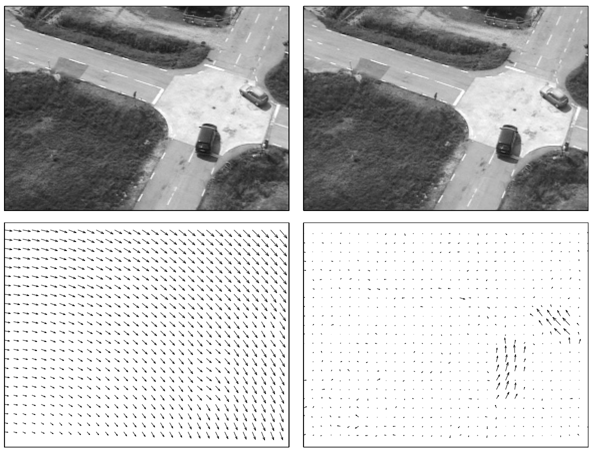
Figure 2: *Two successive frames from a test flight at Revinge (top), the estimated dense motion field (bottom left), and the residual field after estimating and eliminating the background motion (bottom right).*

7 Camera position estimation

The reactive layer on the WITAS helicopter platform deals with objects situated in a 3D World Coordinate System (WCS), while objects as seen by the camera are 2D, and expressed in a Camera Coordinate System (CCS). In order to convert WCS coordinates to CCS coordinates, the camera position, the up vector, and the heading have to be known. Given the world model in the GDR, CCS coordinates may also be converted to WCS coordinates.

One subtask developed for the IPM is a visual odometer. This is a system that computes the 3D displacement and change in orientation of the camera, based on estimates of local image displacements. A visual odometer can act both as a backup, and as a complement to other position sensors, such as DGPS and INS.

As already mentioned, the temporal aliasing in the camera image flow makes it necessary to use two-frame methods for motion or displacement estimation methods. For the visual odometer system, a method based on tracking of a sparse set of image regions was chosen. It typically keeps track of 50–60 local image regions over a longer time sequence. A region is automatically discarded when the matching value drops below a given threshold, and new regions to track are selected automatically by looking at local peaks in the second eigenvalue of the structure tensor.

Regions are tracked by template matching in response images from horizontal and vertical edge filters. Since edge filter responses are statistically

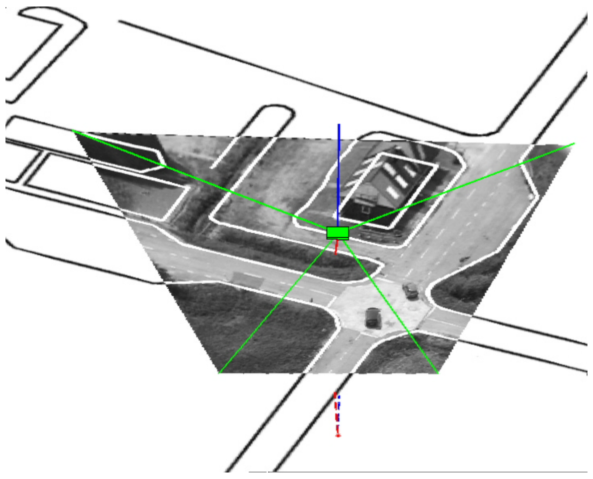

Figure 3: *Position estimation using a homographic model. The estimated camera position is marked as a green box.*

sparse, and low-magnitude coefficients hardly affect the result at all, the templates can be pruned to use only 10% of the total number of coefficients. This results in a speedup of a factor above 3 in the total computations compared to full grey-scale correlation methods [5].

The computation of updates of the camera position are done using a homographic model, i.e., the ground is assumed to be planar. The homographic approximation of the geometry is chosen since it results in a closed form solution of the estimation, which reduces the computational load. When using a calibrated camera, the camera location, the view vector and up vector can also be extracted from the estimation procedure. See figure 3. [4].

At high altitudes the homographic assumption works well, since regions not conforming to the model are detected as outliers and removed from the estimation. However, when flying near buildings, at low altitudes (below 70 meters), the method becomes increasingly less accurate. However, this situation is detected by the visual odometer subsystem, and signaled to the reactive layer that controls the IPM.

8 Fast tracking

Tracking of ground vehicles is an important functionality, where the primary information about the vehicles' positions is provided by the IPM. Again, because of the limited computational power available on-board the helicopter, computationally optimized methods have to be used and one such method has been developed within the project [8].

The new method is based on traditional template matching, but on binary information rather than on the image intensity. This implies that the matching stage is extremely fast, and even allows multiple vehicles to be tracked simultaneously. To get the binary data on which the matching is performed, both the image and the template image is input to a preprocessing stage that compares the intensity values in pairs of image points. The pairs can be distributed either in a regular array or randomly, and each comparison results in one bit. Each individual bit may not be a robust representation of the corresponding image region, but given that the set of pairs is sufficiently large (256 bits have been used in 32×32 regions with acceptable results), the generated set of bits is sufficiently representative of a region to allow matching on the binary data to be used. Once the preprocessing has been made, the matching reduced to only counting matching bits, for with low level optimization can be done. The resulting algorithm can be described as a maximum entropy matching technique since it represents each image region as a set of bits and measures the number of bits which differ between two regions.

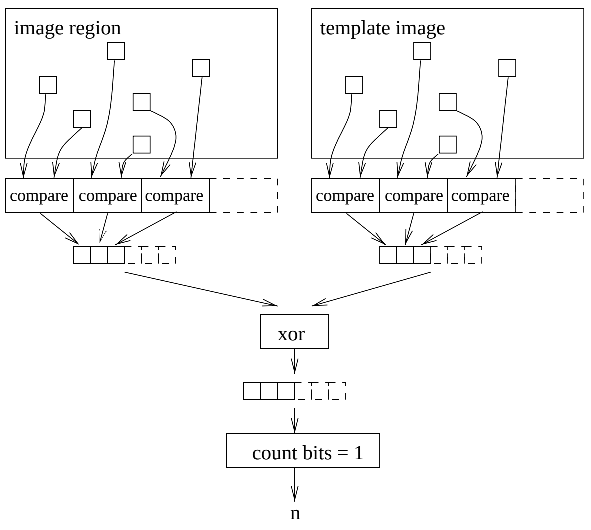

Figure 4: *Fast entropy matching*

The matching procedure is illustrated in figure 4. Note that the preprocessing stage need only be applied on the template each time it is updated.

9 An associative perception-action structure

Low and medium level computer vision can be made using traditional techniques, typically based on convolution in combination with non-linear operations, to obtain feature descriptors of various types. However, already at medium level and in particular at the higher levels of image data processing, where per-

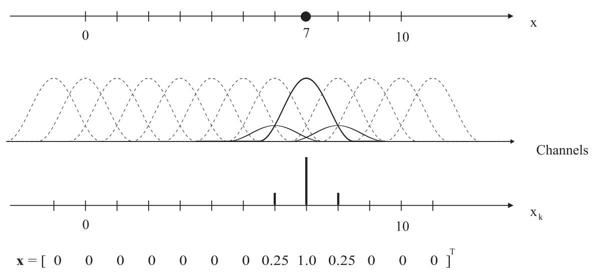

Figure 5: *Channel representation of a scalar $x = 7$*

cepts are used to initiate and control actions of the system, it is becoming more and more apparent that the operations have to be learned rather than programmed. In the project, a new learning method based on an associative perception-action structure has been developed [6].

The method is based on the so-called channel representation of information, for which a real valued variable is represented by a set of channels where each channel represents the certainty or strength that the variable has a certain value, see figure 5. The variable can for example represent local orientation or position of a line or edge. The distribution of the channels in the range of the variable, and the specific function of each channel, is rather arbitrary as long as a certain overlap between the channels is maintained. Given a specific value of the variable, we get a specific activity in the corresponding channels, typically with non-zero values in only a few channels. Vice versa, with appropriate choices of channel functions, the activity of the channels can be used to reconstruct the value of variable. Hence, from a formal point of view, the channel representation is a one-to-one mapping from (typically) a real value number to a set of real numbers. This representation has some advantages which makes it useful for information representation. First, multiple hypotheses about two or more values of the variable can be represented in an easy manner. Second, learning based information processing methods can be greatly simplified and become extremely fast since the representation, in general, is sparse as only a fraction of the channels are active at the same time, and since the channels have values in a limited range (typically [0...1]).

The newly developed learning technique associates a set of input channels to a set of output channels, by means of a linkage matrix **C**, i.e., each output activation **U** is computed as **C A**, where **A** is the input activation. A set of input and output activations can then be represented as columns in matrices **A** and **U**, and the goal of the learning stage is to minimize the difference between **C A** and **U** (supervised learning). Traditionally, this is done by trying to minimize a least squares error, and is solved using either steepest descent techniques or explicit inversion of a correlation matrix.

The channel representation allows both the training and the subsequent processing to be made much more efficient than the traditional techniques because of the sparsity of the data and that simple linear operations can be used in the processing. Fast numerical methods for inversion of sparse and range limited matrices are now available and can be used to compute **C**, even for very large matrices. The sparsity of the input data implies that also **C** is sparse, consequently associating only a few input channels to each output channel. This mean that once the training has been performed, each output channel can be computed using a simple and fast linear combination of only a few input channels.

Figure 6 illustrates how the learning technique can be used for recognition of vehicles. The system has first been trained to associate image features, corresponding to images that contain the specific vehicle at different orientations, to a channel representation of the vehicle's orientation (parameters θ and ϕ). The top image shows a test image containing two such vehicles on a structured background, and one vehicle partially occluding the other. The image also shows the curvature based features as clusters of bright lines. The bottom image shows two distinct clusters, corresponding to activation in the output channels that represent the orientation of the two vehicles. These can then be projected back to the test image to give the position of the vehicles, represented by the two larger circles.

10 Image processing runtime system

The various vision subtasks presented above takes place in the IPM, for which dedicated hardware resources are allocated. In the project different hardware platforms are being used for the IPM, in particular the development platform is different from the target platform. Also, during the operation of the system one essential property is the possibility of reconfiguring the IPM for different types of processing, e.g., having different accuracy, robustness, and speed.

Taking these demands into account implies that a highly flexible runtime system for the implementation and execution of vision algorithms is needed for the IPM. The solution consists of an API (IPAPI) which is the declarative interface that other components in the architecture can use to create, manipulate, configure and execute various vision algorithms. It also includes IPAPI-Runtime, the runtime compo-

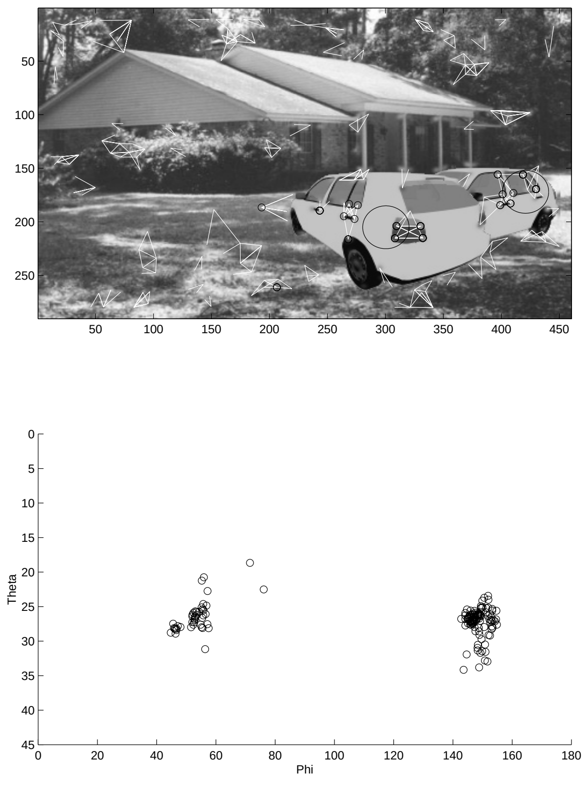

Figure 6: *A test image with two vehicles (top), and the corresponding output channel activation (bottom)*

nent that manages the configuration and execution of vision algorithms, dynamically allocates memory for buffers needed during execution, interfaces to the image controller and other components in the architecture, and manages an existing library of pre-defined vision algorithms [9].

IPAPI-Runtime is implemented in the Java programming language. The use of Java offers a number of advantages; rapid prototyping, support on a number of different hardware/software configurations, access to an ever growing number of APIs for various purposes, e.g., both CORBA support and a powerful toolkit for building graphical applications is available. Thanks to the recent JIT technology for Java, it is also reasonable to implement the actual data processing in Java, i.e., the execution inside the nodes of the graphs. Only certain nodes for which the execution time is critical, e.g., convolution, have also been implemented at native level using JNI. In these cases the optimization has been targeted to use processor specific instruction sets for acceleration of floating point operations, e.g., Altivec for PowerPC or SSE for Pentium. A result of this work is a growing library of nodes which implement various image processing operations, some of them in Java, and some optimized at native level.

In IPAPI, graphs can be defined as new node classes and instantiated as nodes in other graphs. This implies that relatively complex graphs, containing a large number of nodes, can be implemented without too much work. Furthermore, memory management, scheduling of execution and data flow can be customized for various purposes. This flexibility in scheduling is lacking in existing systems of similar type such as AVS, Khoros, JAI, etc.

References

[1] P. Doherty et. al. The WITAS unmanned aerial vehicle project. In *Proc. of the 14th European Conf. of Artificial Intelligence*, 2000.
Project URL: http://www.liu.se/ext/witas.

[2] G. Farnebäck. Spatial Domain Methods for Orientation and Velocity Estimation. Lic. Thesis LiU-Tek-Lic-1999:13, Dept. EE, Linköping University, SE-581 83 Linköping, Sweden, March 1999. Thesis No. 755, ISBN 91-7219-441-3.

[3] Gunnar Farnebäck and Klas Nordberg. Motion Detection in the WITAS Project. In *Proceedings SSAB02 Symposium on Image Analysis*, pages 99–102, Lund, March 2002. SSAB.

[4] Per-Erik Forssén. Updating Camera Location and Heading using a Sparse Displacement Field. Technical Report LiTH-ISY-R-2318, Dept. EE, Linköping University, SE-581 83 Linköping, Sweden, November 2000.

[5] Per-Erik Forssén. Window Matching using Sparse Templates. Technical Report LiTH-ISY-R-2392, Dept. EE, Linköping University, SE-581 83 Linköping, Sweden, September 2001.

[6] G. H. Granlund. An Associative Perception-Action Structure Using a Localized Space Variant Information Representation. In *Proceedings of Algebraic Frames for the Perception-Action Cycle (AFPAC)*, Kiel, Germany, September 2000.

[7] B. Johansson. Multiscale Curvature Detection in Computer Vision. Lic. Thesis LiU-Tek-Lic-2001:14, Dept. EE, Linköping University, SE-581 83 Linköping, Sweden, March 2001. Thesis No. 877, ISBN 91-7219-999-7.

[8] Frans Lundberg. Maximum Entropy Matching: An Approach to Fast Template Matching. Report LiTH-ISY-R-2313, Dept. EE, Linköping University, SE-581 83 Linköping, Sweden, October 2000.

[9] Klas Nordberg, Per-Erik Forssén, Johan Wiklund, Patrick Doherty, and Per Andersson. A flexible runtime system for image processing in a distributed computational environment for an unmanned aerial vehicle. In *Proceedings of IWSSIP'02*, Manchester, UK, November 2002.