Computer Vision
Craft, Engineering, and Science

David Vernon
Commission of the European Communities

Editor

Foreword

This volume is one of the results to emerge from the Working Group in Vision (ESPRIT Basic Research Action 3352). The Working Group, comprising 37 participants, aimed to foster a strong and focussed research community in a specific and important area of information technology.

One of the six sub-groups of the Action examined how the construction of vision systems should be assessed. There is a spectrum of approaches, each approach with its own degree of maturity and level of robustness. The overall goal is to highlight this variety, to understand the way these approaches influence and complement each other, and hence to assist the successful exploitation of computer vision in industry and elsewhere.

As always in research, the results in this concise volume are neither an end in themselves nor a resting place, but pointers for the future.

In computer vision, these indicate very fruitful roads yet to be travelled.

G. Metakides

Preface

The material for this book arose from a workshop which was organized by Trinity College Dublin as part of the activities of a Basic Research Action. This Action, No. 3352, was funded by the Commission of the European Communities under the European Strategic Programme for Research and Technology in Information Technology (ESPRIT) and it was held in Killarney, Ireland, on the 9th and 10th of September 1991.

The orginal title of the workshop was *Vision in Context* and the theme was to discuss and discover whether or not the design of a vision system is helpfully delimited by its environment and, if it is, then how. That we have chosen to change the title of the book to *Computer Vision – Craft, Engineering, and Science* confirms that the workshop worked as it should: as a forum where thought and discourse on a topic lead one to a position which was not, and perhaps could not have been, anticipated at the outset. This is exactly the case here. While we set out to tease out the relationships between application and system, much more was achieved and a clear consensus emerged on the spectrum of vision systems that exist and the attendant architectural and processing paradigms that match the various positions on this spectrum. Without wishing to pre-empt the material which will be presented in the book, and in particular in the last chapter, it might be useful to summarize here the main findings of the workshop. There were three key conclusions.

The first concerns the issue of autonomy in vision-based robotics systems. It was generally agreed that autonomy, *i.e.*, self-governing, self-reliant, and independently robust behaviour, is indeed one of the ultimate goals of vision and robotics – the 'Holy Grail' if you will. However, the consensus was that true autonomy is a much more difficult characteristic to achieve than is popularly realized and may not be achievable in machines in the near future. Happily, the contributors to the workshop also felt that it does not constitute a major obstacle since it is not clear that autonomy is desirable anyway for it raises problems about how to control such systems; external control being contradictory to the concept of autonomy. Rather than opt for complete autonomy, it may be better — and it is certainly more pragmatic — to leave a person in the processing loop. This person takes responsibility for the ultimate interpretation of the information which is conveyed to him or her. The information may be

extremely rich, being the result of very sophisticated visual processing, analysis, and reasoning, but the final decision is taken by a human. In a sense, this type of system acts as an intelligent filter, screening large amounts of data before passing the essential information on in a manner which is easily assimilated or interpreted. Perhaps one of the best examples of such a system is the one which is described in chapter 4. This is a sophisticated model-based computer vision system for monitoring and interpreting the free-moving behaviour of vehicles — such as cars, trucks, motorbikes, or indeed airplanes — in their natural setting on roads or in airports. The purpose of this system is to take the huge amount of visual data being generated by a surveillance camera, to reduce this information to a parsimonious, symbolic, description of the vehicles behaviour and to flag potentially-dangerous behaviours and events such as imminent collisions or the presence of certain vehicles in a restricted area. The final validation of this interpretation and the consequent action is taken by a human controller – the person in the loop — but he or she is assisted in this decision process by the vision system.

There are other, but perhaps less obvious, ways in which the person can be 'left in the loop'. It could be argued that the explicit use of *a priori* knowledge, at either the very simple or pragmatic level such as is the case with the visual inspection system described in chapter 1, or at a more sophisticated and structured level such as is the case with the robotic system which is described in chapter 2, tacitly involves the embedding of a *homunculus* in the system; a homunculus which represents the knowledge, understanding, and intent of the human designer. Certainly, this knowledge base must be well-formulated and well-grounded and it must be accessible in a useful manner to the processes constituting the vision system but it is nonetheless an embodiment in rules of the designer's or the user's anticipated interpretation and actions.

The second conclusion of the workshop concerns the manner in which a system is designed for a particular application. It was generally agreed, after considerable discussion, that, in developing a system for a specific application, the design paths follow a line of first specifying the required functionality, then identifying the knowledge which is appropriate to that task — which is a necessary and sufficient characterization of the task — and finally the appropriate techniques and algorithms are chosen to operate on this knowledge to produce the required result. This was a surprising conclusion since it is commonly assumed that the path is one of functional specification, algorithm selection, and thus, implicitly, knowledge representation.

Finally, we come to the third key conclusion of the workshop and this concerns the methodologies wich are employed in developing vision systems and, in addition, the manner in which these methodologies evolve over time. The subtitle of this book — *Craft, Engineering, and Science* — captures the qualitative difference of the various approaches which are employed in the design of vision systems: from the intuitive approach based on experience (*craft*), through the methodological, pragmatic, but theoretically principled (*engineering*), to

the strictly formal, mathematical, and rigorous (*scientific*). However, these approaches are strongly related. As the discipline of computer vision matures and grows, the craft approach influences the engineering and *vice versa*, just as the scientific influences the engineering. But the relationship is not merely a linear one, with craft at one end and science at the other, for the intuitive understanding of a discipline, grounded in experience, is a key element in the formation of scientific theories and in the creation of scientific paradigms. The relationship, then, is much more: a cyclic one in which the three terms of craft, engineering, and science are mutually relevant and contribute to one another.

This book is stuctured in a slightly different way to the workshop itself in which keynote addresses and workshop discussion on a related theme were held alternately. Both the keynote addresses and the discussion themes spanned, and progressed along, a spectrum of issues, beginning with the pragmatic and industrial and proceeding to the more global issues of computational models of vision. In this book, the keynote addresses are contained in chapters one through five, while the discussions on each of the themes have been incorporated in one single chapter (chapter six) at the end. Thus, each of the first five chapters are relatively self-contained (although they do make reference to issues discussed elsewhere in the book) but taken together they represent the core thoughts of the participants. The final chapter follows through on these thoughts and it represents the evolution of ideas which the workshop fostered. It also represents a summary of the key conclusions and findings of the meeting.

Before finishing, it is perhaps useful to remark that the workshop was intended to be a technical meeting in the true spirit of a workshop: a forum for the presentation of ideas, mature and speculative alike, and for the discussion of these ideas. It was not intended as a medium for the dissemination of results but for the development of a, perhaps new, consensus on the workshop topics. I believe that it was successful in achieving this goal; it was certainly a stimulating and enjoyable experience and the ideas which formed as undoubtedly interesting. However, their validity can only be tested and confirmed by actually attempting to use them in practice, in the design of vision systems. I hope that this slim volume provokes some further thought on the matter and that it captures at least a small part of the open debate which was evident throughout the two days of the workshop. Such free discourse — where ideas clash and are reconciled — is a fundamental part of the development of any science or discipline, and it is one which is all too often neglected.

D.V.
Brussels 23/9/92

Acknowledgements

This workshop was funded by the Commission of the European Communities under the European Strategic Programme for Research and Development in Information Technology (ESPRIT), Basic Research Action 3352.

List of Contributors

J. Mahon, Trinity College Dublin.

Chapter 1: Reliability and Robustness in Industrial Computer Vision.

J. L. Crowley, LIFIA (IMAG) Grenoble.

Chapter 2: The Role of Context in an Integrated Active Vision System.

G. Vernazza, University of Genova.

Chapter 3: From numerical to symbolic image processing: Integration of computational and knowledge-based approaches.

G. D. Sullivan, University of Reading.

Chapter 4: *A Priori* Knowledge in Vision.

A. Sloman, University of Birmingham.

Chapter 5: How to Design a Visual System – Gibson Remembered.

K. Dawson,[1] D. Furlong,[1] M. Jones,[1] N. Murphy,[2] and D. Vernon[3]. [1]Trinity College Dublin, [2]Dublin City University, and [3]Commission of the European Communities.

Chapter 6: Computer Vision — Craft, Engineering, and Science.

Contents

Chapter 1

Reliability and Robustness in Industrial Computer Vision

J. Mahon

Machine Vision Centre,
Department of Computer Science,
Trinity College Dublin,
Ireland.

1.1 Introduction

There are noteworthy differences between the development of a laboratory prototype of a vision system and the production of a machine which can function successfully on the factory floor. In the laboratory, the primary consideration is that of functionality: whether or not the system do what it was designed to do. However, once on the shop-floor, functioning as part of a manufacturing system, the issues of robustness, reliability, and repeatability assume an equal, if not greater, importance than that of functionality which is very much taken for granted. In recognition of this, we begin our treatment of the design of computer vision systems by considering the so-called 'bottom end' of the vision spectrum and, in this first chapter, a vision system is described which has been designed, constructed, and used successfully in a production environment to inspect surface-mounted chip components after placement.

The placement of surface mounted chip components on printed circuit boards

is an imperfect process, especially when combined with a glue dispensing operation. Components can be incorrectly placed in terms of both X or Y directions and their orientation can also be skewed. Additionally, if the glue by which they are affixed to the board is not present in sufficient quantities, the devices may lose their position entirely and either become dislodged or grossly displaced. It is difficult to test for some devices electronically, especially decoupling capacitors, as they may be connected in parallel with IC devices of larger values, whose tolerances ($\pm$5–10%) swamp the value of the chip component. It has been observed [1] that, as the task is exacting, tedious, and extremely repetitive, human inspection is not ideally suited to the task of verifying the correct fixation of these devices. Even well-motivated operators are only 75% effective, while fatigued or distracted operators can be less than 50% effective. Additionally, as technology advances and device sizes shrink, human visual inspection becomes even more difficult and automatic visual inspection in this field becomes essential.

1.2 Review of previous work

PCB inspection can trace its roots back to the pioneering on bare board inspection work of Ejiri et. al. [2] at Hitachi Inc. in 1973, but it was not until the mid 1980's that inspection was performed on Surface Mounted Technology (SMT) components with the work of Buffa [3]. Several systems are available today from a number of sources. These systems use a variety of technologies but they are based mainly on 2-D and 3-D vision. 3-D systems are perhaps bext exemplified by a system produced by SVS Inc. [4] which uses a dense range map derived from a synchronized laser scanner similar to that described in [5] by Rioux. This system can produce registered range and intensity images of flat surfaces such as PCBs. Two inspection tasks are implemented: solder paste inspection and component placement inspection. Each task uses a different resolution and runs at different speeds. The component placement system, Model 7620, uses a 50 μm pixel size and it can inspect between 6.5 and 13 cm^2/second of fully populated printed circuit board while the solder paste system uses a 25 μm pixel size and can inspect between 3 and 6.5 cm^2/second.

2-D systems can be characterised by the Cimflex 5515/6/7 series from Control Automation [6]. These systems use four CCD cameras, each viewing the PCB at an angle of approximately 30° to the vertical, and with the principal ray of each camera oriented at 90° to each other. The board is illuminated by a hemishperical dome of LEDs surrounding the the view area. The activation of individual LEDs is under software control. This allows the scene to be illuminated from any angle, or set of angles. Model 5515 can operate at up to 77 cm^2/second for the inspection of component placement using a 50 μm pixel size and a 2.5 cm^2 field of view. These systems cost in the order of $200,000 to $250,000 depending on what options are chosen.

The 2-D system is much faster than the SVS range imaging system. On the

other hand, it has been argued [7] that 3-D data is inherently more reliable than 2-D grey-scale intensity data and thus, by implication, so too is the inspection process. The argument is that factors such as specularity and variance in component colour from batch to batch make it difficult for a 2-D system to function well. Nevertheless, while 2-D systems may have difficulty with more demanding tasks such as solder joint inspection [1], it seems likely that they still have much to offer in tasks such as component placement. The number of Cimflex systems which have been sold (in excess of 200 units) [8] would seem to confirm this point.

1.3 The Inspection Task

We now come to the inspection task for which the vision system described in this chapter was developed. The system was designed to inspect bottom-side PCBs (i.e. PCBs which can have components placed underneath, as well as on top of, the board) for presence and absence of components and to check component placement in terms of its X and Y position and in terms of its skew angle. It was also designed to inspect for the presence and location of glue spots. One of the main constraints on the system is that it has to run fully in-line in the production facility at a speed no less than the line speed. This constraint allows 40 seconds to inspect a PCB which is 300 $\times$ 280 mm in size. Both the false accept rate and the false reject rate were planned to be less than 1 in 10,000. Of these, the false accept rate is the more critical since the boards are manually reworked, and a false reject can be ignored. The system was to be insensitive to normal manufacturing variabilities including any variations from batch to batch in PCB substrate colour, and chip colour and appearance. The devices are surrounded by white box markings which had been provided for manual inspection and which could not be immediately removed on changeover to an automatic system. One of the problems with the boxes is that they are often broken by vias on the PCB, or they can be badly printed, or they can merge into one another if the device spacing is very close. Some of these features can be seen in figure 1.1.

The components to be inspected differ in two ways: they can have different coloured bodies, and their ends can have different shapes and finishes. The substrates are generally green in colour but they have conductors crossing them and they have copper hatching on their surfaces. Table 1.1 summarizes this scene variability.

1.4 System Hardware

The system, which has been dubbed *VISP* for Visual Inspection, is hosted by an 80386-based PC clone. It consists of an Aerotech Unidex-14 X-Y table with DC

Substrate	Bounding Boxes	Components
Any colour May contain conductors	Broken print Components can merge with boxes and printed characters Boxes can merge with neighbouring boxes	Differently coloured bodies Different shaped and finished ends

Table 1.1: Scene variability

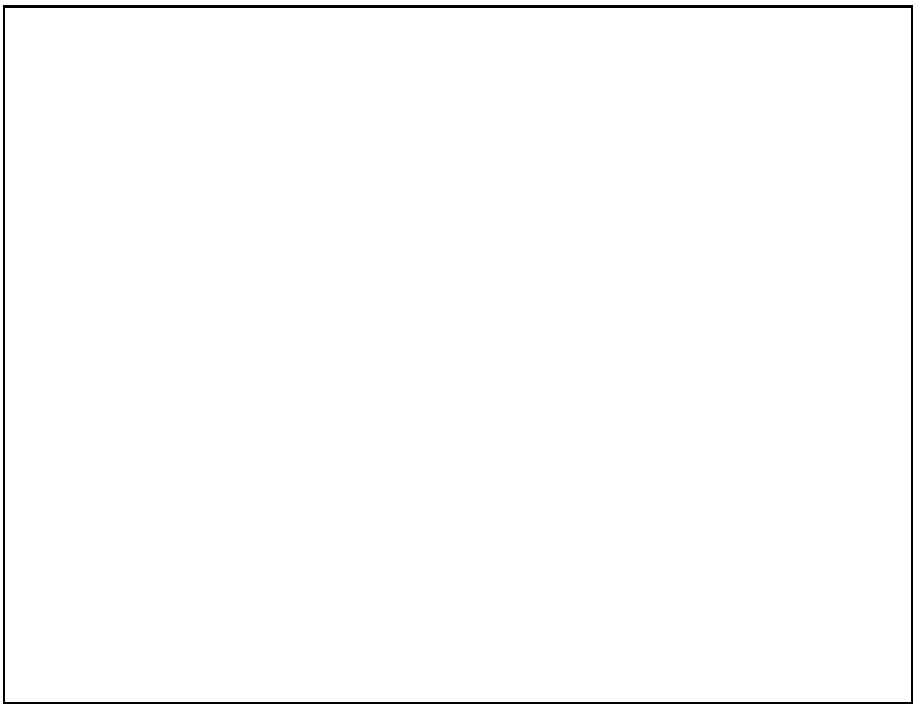


Figure 1.1: The appearance of the PCB under low level Illumination

Figure 1.2: VISP Hardware architecture.

servo drives and a Digithurst TM monochrome transputer-controlled framegrabber. This framegrabber can grab images at a 720×512 pixel resolution and it has 1 MByte of local program memory and a 25 *MHz* T800 transputer processor running at 5 wait states for local memory access. The system has a 24-line Opto22 digital I/O controller. A Sony XC77CERR camera is used with a 35 mm Pentax lens, giving a field of view of approximately 40×30 mm, with a pixel size of 55 μm. Lighting is provided by a compact square of four 20 watt Philips PLC high-frequency fluorescent bulbs, each of which can be driven separately under software control. These are placed approximately 10 mm above the PCB to ensure low level illumination of the devices for best image segmentation (see figure 1.2 for a schematic overview of the architecture).

1.5 Software

The software sub-system comprises one main program running on the transputer framestore. This was coded in 3L parallel C, although the parallelism has not yet been exploited since just one processor provided adequate computational power for the task. The framestore communicates with the PC-host using a version of the Inmos Alien file server (AFS) protocol to perform such tasks as screen and disk I/O and X-Y table control.

1.5.1 Motor control

The control of the X-Y table movement is a critical issue. There are two basic approaches: a non-stop raster-scanning type motion such as that used by Cimflex, or stop/start motion such as that used by IRT [9] and by Mahon *et al.* [10, 11, 12] in earlier systems. Each has its advantages: a scanning system allows a much higher number of images per second (at least six) to be acquired and processed, as there is no need to accelerate and decelerate the camera head between snapshots. On the other hand, much more processing power is required to process the increased amount of data in the time available. This usually requires several image processing modules and a parallel processing architecture, all of which increase the cost, the programming difficulty, and the development effort. A point to point system, though slower, is much simpler. Once the image is acquired, the next movement can be initiated and the image processing and analysis can begin. The movement and processing can continue in parallel until both have terminated, at which stage the next image can be acquired and the cycle can proceed. In most cases, the processing will be faster than the camera movement, but if this is not so (such as in the case of densely packed regions on the board) the system simply waits for the processing to complete. No image frames need be lost. The system can be designed around a single processor working at an average, rather than a worst case, speed without a significant decrease in overall processing time, while the programming difficulty and effort are greatly reduced. In the final *VISP* system which was installed in the factory, the overall inspection time met the required specification, so it was decided to leave the system configured as a point to point system. Another benefit of this approach is the reduced X-Y table wear due to shorter distance travelled since an optimised point-to-point route will be shorter than a raster scan of the PCB.

1.5.2 CAD data

The system can automatically generate from a PCB layout file an optimised list of the inspection sites which the camera must visit. This optimal list minimises the total camera (and X-Y table) travel. The PCB layout file contains the locations, types, names, and orientations of the components, and is cross-referenced with a part description database containing information on the size and shape of the different component types and the sizes of their bounding boxes. These can be combined to generate a set of 'views' for an inspection list where a 'view' is a position on the table which locates a field of view for a camera at a given resolution. The views are initially generated in raster fashion, which yields a sub-optimal inspection path. This is then improved using a simulated annealing algorithm [13] to generate an (almost) optimal inspection path. The entire training process takes about five minutes, assuming no new devices have to be added to the component database, and that all the information is available. Once the view files have been generated, they can be loaded at any time, and

in general the system can be changed over from one product to another of the same board width in under three minutes by a technician.

1.5.3 Imaging Algorithms

The vision task is to locate each component in X and Y directions, and to estimate its skew (ϕ) with respect its bounding box in a short time ($<$ 50 ms) as accurately as possible (see fig. 1.3 for details). The scene variability problems of the inspection task have already been discussed, but to these must be added the possibility of the end of the component covering or merging with the white bounding box which is printed on the PCB. Because of this, a simple thresholding and blob analysis based approach, such as is effected by the so-called SRI connectivity-analysis algorithms[14], could not be used, since it would not be able to deal with the occlusion. An alternative approach is used and is described in the following.

The first task is to locate the bounding box, which, as mentioned above, can be fragmented by via holes (in one instance, the box was broken by five via holes) and it can also merge with other boxes. Since template matching or cross correlation based algorithms exhibit good resistance to local variations, such approaches are very useful, as long as size and orientation of the boxes are known in advance. However unless hardware, such as the Cognex system [15], is available, 2-D cross-correlation is extremely slow. Fortunately, the task can be reduced to a 1-D problem and, thus, it yields to a software-based cross-correlation solution. In this case, since the boxes are reasonably symmetric and regular and since they are aligned along the camera major and minor axes, the box can be located, first along the X axis, and then along the Y axis. The location of the box in the X direction is accomplished by summing the pixel values in the Y direction to form a vector V_x (refer also to figure 1.4):

$$V_x(i) = \sum_{j=j_{min}}^{j_{max}} I(i,j) i_{min} \leq i \leq i_{max}$$

where $I(i,j)$ is the grey-level intensity of the image at point (i,j) and where $i_{min} \leq i \leq i_{max}$ and $j_{min} \leq j \leq j_{max}$ are the ranges of image coordinate values which define the image region in which the box is expected to be found. The x coordinate of the location of the box is then computed from the cross-correlation of this signture with an ideal signature W_x comprising two spikes separated by a distance equal to the theoretical dimension of the box in the X direction. The y coordinate of the location of the box is computed in a similar manner. This localization is performed to approximately 12μm repeatability using sub-pixel resolution techniques to improve the accuracy.

Once the box has been located, the task remains to locate the component within it in X, Y and ϕ. Again, this is done one dimension at a time, in a manner similar to that for computing the location of the white box. The

board is illuminated so as to emphasise the specular contacts on the ends of the components, and they are located using 1-D image analysis, or classified as absent. The component is first located along its body in the X direction. The colour of the centre of the component is not known at run time (as it can change from batch to batch), and thus constitutes a 'don't care' in the correlation template mask. The pixel values inside of the box are summed to form a 1-D vector of values, and this vector is correlated with the 1-D ideal template. If the maximum of the correlation result is below a threshold for that component type, the component is deemed to be absent. Otherwise, it is deemed to be present and located at the position of the maximum point in the output vector (see figure 1.5).

The system now proceeds to compute the y coordinate of the position of each of the chip contacts using another 1-D technique. A vector is generated by summing the pixel values in the X direction, and this vector is then thresholded at two levels, one corresponding to the contact or bounding box, and the other to the pad or substrate. It is not possible to segment the component contact from the bounding box. Once thresholded, the centre of the contact can be located irrespective of whether or not it was occluding the box. The decision to use thresholding was taken only after it was finally deemed impossible to obtain robust results from correlation techniques due to the occlusion problems. The correlation techniques were found to be accurate to $\frac{1}{4}$ pixel in the absence of occlusion, while the thresholding techniques were only accurate to one pixel but dealt with occlusion properly. In this case, a decision was taken to increase the robustness of the system at the expense of the average case accuracy. After each end has been located, a skew measurement can be generated, and the component can be classified against a set of thresholds for each device type, or against special individual thresholds for exceptional devices. The results can be sent to a printer, disk, data collection system, or it can be displayed on the video monitor for human verification.

1.6 Repeatability and robustness.

The three critical parameters in an inspection system are *speed*, *accuracy*, and *robustness*. Speed is the time taken to perform a given task and can be easily measured. In this respect, it is not a contentious issue. Accuracy can be defined in two ways: *absolute accuracy*, and *repeatability*. Absolute accuracy can be defined as the difference between the value measured and the 'true' value as defined by some more accurate system. Repeatability is a measure of the variance of the values obtained from the measuring system and it is a limiting factor on absolute accuracy. In general, it is very difficult to measure the 'true' value of the location of a component, but it is quite simple to compute the location a large number of times to get a measure of the repeatability. For this (pragmatic) reason, repeatability rather than absolute accuracy is used as the measure of

Figure 1.3: Appearance of a Component and bounding Box.

accuracy. On the other hand, *robustness* is the ability of a system to perform to specification when the original assumptions about the visual appearance of the scene are no longer valid, or when the scene has changed in some unexpected way. Robustness is a critical issue in real world imaging systems; even in a world as constrained as PCB inspection. If the system produces spurious results, and if it does not detect this, it has very little value. Robustness is thus a critical factor in an inspection system and it is one which is rather difficult to measure. To obtain some measure of robustness, an approach based on repeatability was used. This was later validated by a manual inspection when the system was on the production line. The critical difference between robustness and repeatability is that problems with robustness result in large measurement errors, occurring occasionally when the image model breaks down. Repeatability problems result in small measurement errors, when noise or some similar factor causes problems. These variations occur frequently, rather than infrequently.

These inspection problems are detected with a series of graded tests which are designed to locate difficult areas and then to locate the problem precisely. An ideal repeatability test would be to inspect the entire PCB one hundred times, for example, and to compute the mean and standard deviation statistics for each measurement of each of the three hundred or so devices. This, however, would be very slow, and would generate so much data that it would be difficult to clearly see the problem areas, and act on them. This 'whole board' inspection, or 'macro' test as it is called, is indeed used, but only to locate the worst devices, which can then be further inspected with 'micro' tests.

Figure 1.4: Schematic description of the process used to determine the location of the white bounding box in the horizontal direction.

Figure 1.5: Schematic description of the process used to determine the location of the component in the horizontal direction.

1.6.1 Macro Tests

The macro test can be used in two ways: measurement and classification. The measurement approach examines the values measured, and generates mean and standard deviation measures for these. It gives an idea of how accurately the system is able to measure component locations. The classification approach examines the pass/fail results returned by the system, and measures how reliably the system could be expected to perform in practice. It can detect components being passed in some inspections, and failed in others. It also provides a measure of the minimum and maximum values obtained for these 'cross threshold' measures. If the max-min difference is small, the system is said to be performing acceptably, but the data is near the threshold, and nothing can be done about it. If the max-min difference was large, a repeatability error is flagged and investigated further.

1.6.2 Micro Tests

There are two basic kinds of micro tests: static and 'walking'. A static test simply measures a parameter n a given number of times (one hundred times in the present system) and it generates a histogram of the results, computing the mean and standard deviation of the measured values. The obtained accuracy is deemed to be three times the standard deviation (σ) since 99.75% of all results will lie between $\pm 3\sigma$ of the mean if the data is normally distributed [16]. This test will measure the effects of electronic noise, variability due to vibrations, and similar effects or perturbations, but is really only useful for detecting gross errors in robustness or repeatability. Its advantages are the speed of carrying out such a test, and the lack of the need to have any actuation system. Figures 1.6 and 1.7 show examples of these static tests with sub-pixel resolution and without.

A dynamic or moving test is similar to the static test, except that the scene is moved some small amount in the field of view, typically using an XY table. This kind of test can show variability due to movement as well as noise and vibration. Its advantage is speed of operation, and simplicity.

A much more useful test is the 'walking' test. The approach is to move the X-Y table supporting the PCB from under the camera in a regular fashion (say 1/20 of an inter-pixel distance in both the positive and the negative X direction per iteration) and to repeat this 10 times (see figure 1.8). This yields forty sets of 10 measurements at each location. These results can then be graphed against table movement. This should, in general, produce two kinds of graphs: a horizontal line, if the feature under investigation does not vary with movement (e.g., area under X movement – see figure 1.9), or a straight sloped line if the feature does vary with movement (e.g., X location under X movement – see figure 1.9). This test measures standard deviation (for the set of 10 values) at each location, and estimates how much each measure deviates

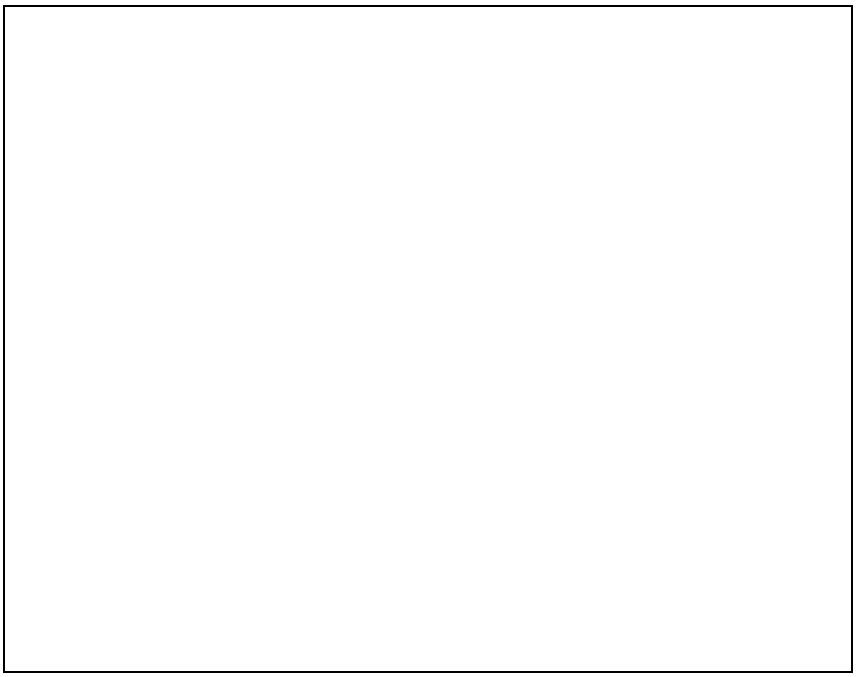


Figure 1.6: Static test using full-pixel resolution

from a best-fit line through the data. It shows how movement affects the results and repeatability, and also shows how sub-pixel location algorithms perform as the edge or feature moves across one pixel and into the next. It can also show and estimate how much better sub-pixel algorithms perform than 'full-pixel' resolution measurements.

1.6.3 View Test

Other problems associated with image variability can arise during inspection. For example, there may well be a difference between the visual appearance of a component when it appears on the periphery of an image rather than at the centre. A view test, in which a device is inspected at the edges and corners of a view, as well as in the centre, is also incorporated in the battery of tests in the system. This test can show up image variations such as illumination, and variations due to the 3-D nature of the scene and the camera viewing angle; see figure 1.10 for an example.

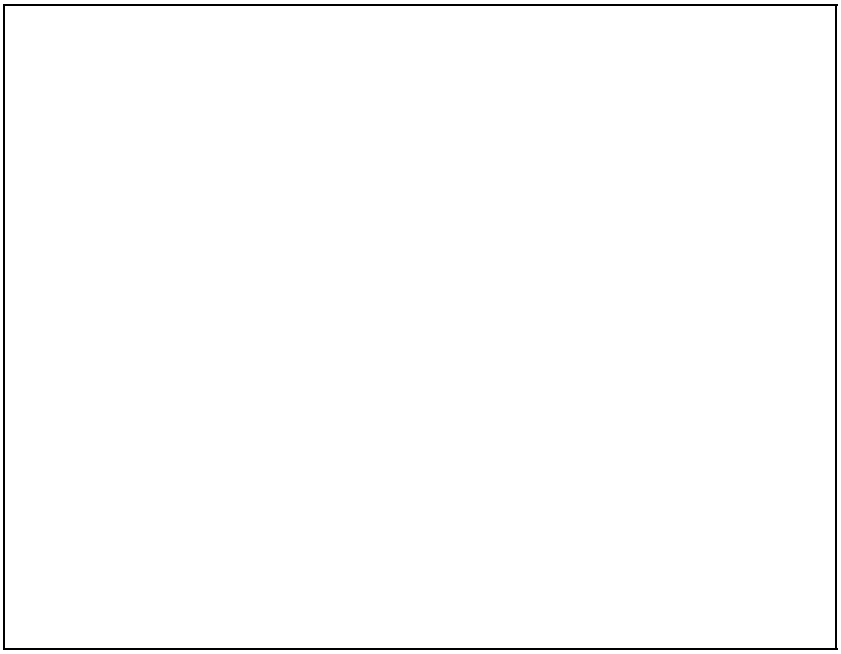


Figure 1.7: Static test using sub-pixel resolution

Figure 1.8: Camera movement in the walking test; all movement is in the X direction (the Y movement is shown only for clarity).

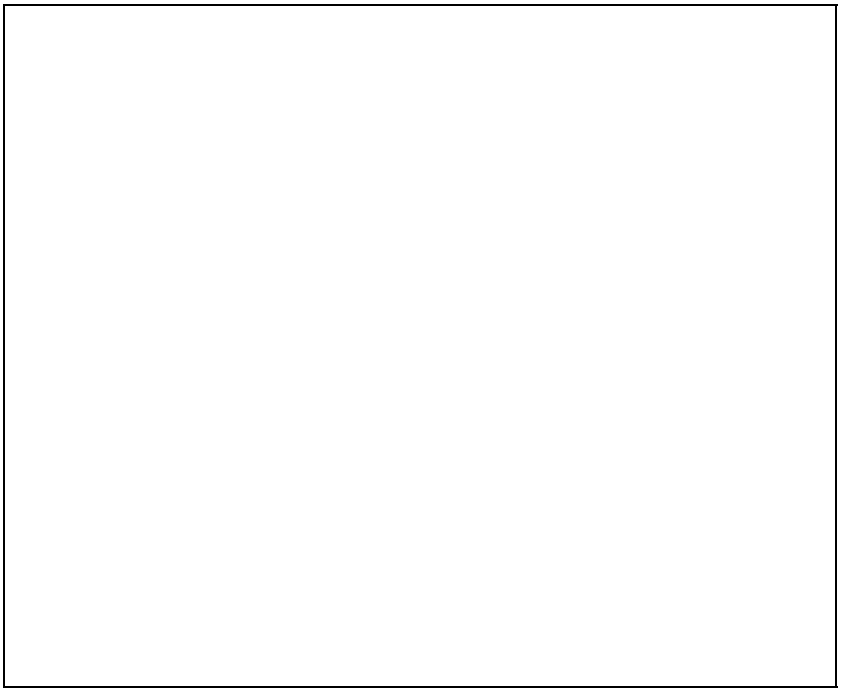


Figure 1.9: Walking test with a 'horizontal' result

Figure 1.10: View area test showing nine test locations.

1.6.4 Software Test Points

To assist with the assessment of the performance of the system, software test points has been incorporated in *VISP*. Software test points are intermediate values that are available to the test data collection system, and which can be plotted and examined as if they were external results. The ability to perform a walking test on intermediate values can quickly track down an error in an algorithm, which would otherwise be very difficult to detect using conventional debugging techniques. This results in a 'visual debugging philosophy', where graphs of intermediate data can be displayed to help in monitoring the functioning of algorithms. This is particularly appropriate, as many of the algorithms use 1-D data vectors which can easily be graphed. Table 1.2 summarizes all of the available tests which are incorporated in *VISP*.

1.6.5 Sub-Pixel Resolution

The speed and accuracy requirements for the system are quite harsh. To keep the number of views down to a number which can be inspected in 40 seconds, the view size has to be quite large (40×30 mm). This gives a pixel size of 55 μm. However, an accuracy of 15 μm was required for accurate classification, and so sub-pixel resolution techniques had to be employed. Rather than resampling the image, or any such 'front end' approach, a 'back end' approach was used to extract the locations of the maxima of the 1-D vectors to sub-pixel resolution by interpolation. This was done by smoothing the vector, differentiating it, and locating the zero-crossing in the resultant signature at the maxima or minima to sub-pixel precision. This has proved to be very useful as all the vision algorithms which are used are amenable to this approach since they all involve locating the maxima of a function represented as a 1-D vector.

The key issue in this technique is the degree of smoothing which should be used so that the subsequent estimate of position can be made with the maximum precision. Smoothing arises from two sources: image blur and applied (digital) smoothing. Image blur is important as it allows the measurement and analysis algorithm to remain valid as the edge moves across an interline transfer region in a CCD sensor. Simulations of edges moving across a pixel boundary showed that a single pass of the vector with a filter kernel with weights (1, 2, 1) or (1, 1, 1) gives better results than either no smoothing at all or two passes of a (1, 1, 1) filter kernel. This has been borne out experimentally in all of the previously-described tests.

Loss of precision occurs when a vector is smoothed. The (1, 1, 1), or box, filter smoothing function will sum 3 values, and divide by 3 to produce the result. This approach discards $1\frac{1}{2}$ bits of precision, which can be retained simply by not dividing by 3. This was called 'raw' smoothing. In this way, vector precision can be increased beyond 8 bits. In a similar fashion, if several lines are summed together to form a vector, they should not be scaled down to 0-255, but simply

Test Type	Test Characteristics
Static (S)	Detects lowest level repeatability. Test Measurements are taken 100 times and results are plotted as a histogram.
Moving (T)	Detects repeatability under movement. Test Measurements are taken 100 times with movement and results are plotted as a histogram.
Walking Test	Detects repeatability and linearity of sub-pixel location algorithms. Measurements are taken 10 times by walking through 40 moves. Results are plotted against movement.
Jump Test.	Measurements are taken at the corners of the field of view. Results are measured against position in view. Detects performance of algorithms across the view in X and Y.
Measurement PCB Test (M_test)	Detects repeatability across the whole board Useful for locating robustness problems The whole PCB is inspected 10 times and means and standard deviations are calculated for every measurement.
Classification PCB Test (C_test)	Detects pass/fail repeatability across the whole board. Useful for locating robustness and 'close to threshold' problems. The whole PCB is inspected 10 times and means and standard deviations are calculated for every classification.
Software test Points	These can be used with S, T and walking tests to locate problems more accurately

Table 1.2: Summary of In-Built System Tests

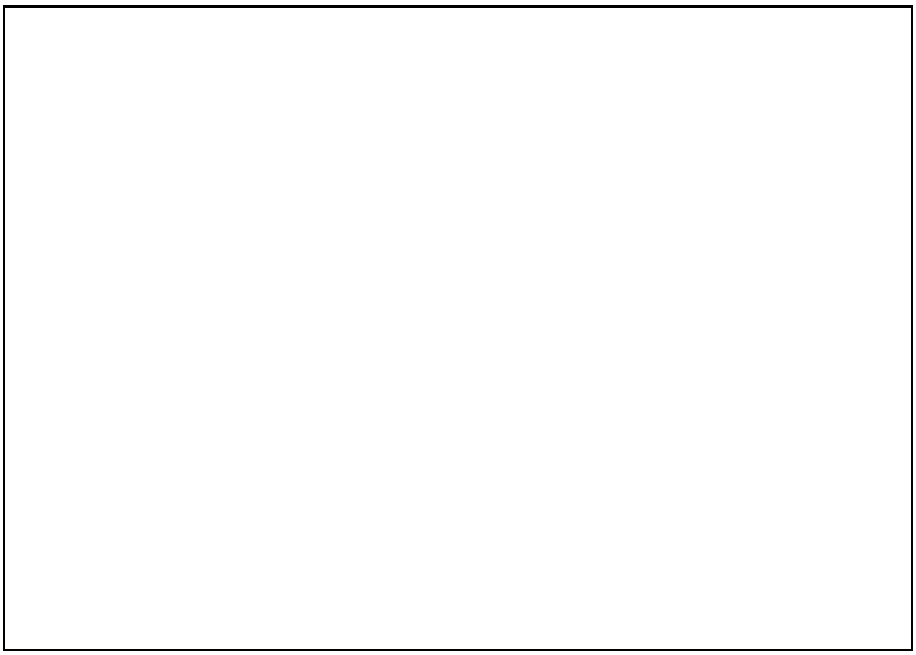


Figure 1.11: Edge location with full-pixel precision in walking test.

summed. This avoids artifacts caused by truncation, and can help locate blurred edges with greater accuracy and discrimination.

However, if it is to be credible, sub-pixel precision must be applied to all processes in the system. Thus fiducial location (i.e. the location of registration points on the PCB), box location, and device location must all be performed to sub-pixel precision. In general, the precision of the system will be somewhat worse than the least precise of any techniques involved in generating a measurement. Nonetheless, the results can be located with $\frac{1}{4}$ pixel 3 sigma repeatability, that is, three times the standard deviation of the measured values are less than $\frac{1}{4}$ pixel, which is the equivalent of having a frame store of size 2880×2048 pixels if it is being used without sub-pixel techniques. The sub-pixel algorithms constitute approximately a 10% overhead on processing time which is a very low price to pay for such a significant increase in accuracy. Examples of the results which have been achieved are illustrated graphically in figures 1.11 and 1.12.

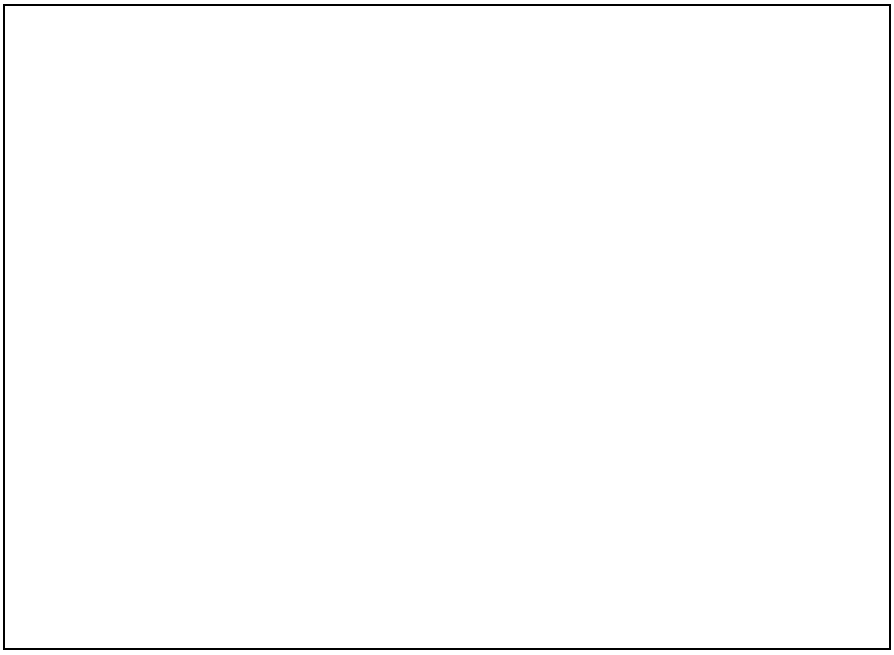


Figure 1.12: Edge location with sub-pixel precision in walking test.

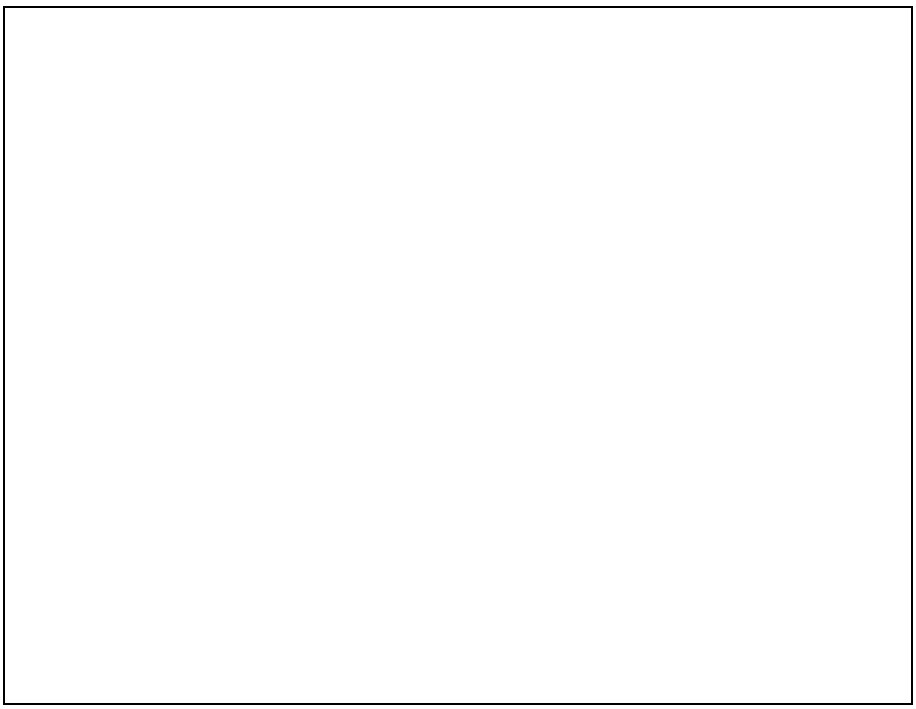


Figure 1.13: The system on the production line.

1.7 Installation

Once the system was moved from the laboratory to the beta test site and a large number of PCBs were inspected, it was noted that the system was inspecting one board type in under 25 seconds while the line cycle time was 55 seconds for that product. It was thus decided to decrease the field of view from 60×40 mm to 40×30 mm to increase accuracy and robustness. This increased the cycle time to 40 seconds, which was still acceptable. It generally increased confidence in the system, especially with the smaller components. The system was then moved out to the production line (see figure 1.13) and, after running alongside it for a week, was moved in-line and performed well from day one, improving yield downstream and finding a high level of acceptance with line supervisors and operators alike.

1.8 Performance

The installed system is able to inspect a 300 × 280 mm board containing 360 components in 50 seconds and 200 components in 39 seconds, averaging from 21.5 to 16.8 cm^2 per second, or about 7 components per second, after the field of view had been decreased. Previously it had been about 40 square cms and 12 components per second, but this speed was unnecessary. The system can measure locations to about 0.25 pixel accuracy (with 3 sigma repeatability) under good conditions increasing to about 1 pixel under less favourable conditions. This results in one false accept error per day: a very significant improvement on the manual inspection which it replaced. The in-circuit test (ICT) yield downstream of the system improved by 4% when the system was installed, and overall product quality was improved as boards with missing components were no longer being shipped out of the factory. The users calculated that the system had a payback time of less than 12 months.

1.9 Conclusions and Future Work

An automatic chip component inspection system has been designed, developed, and is now being used in-line in an SMT assembly process. The success of the project was greatly helped by a thorough approach to testing through the use of layered repeatability tests and the availability of a large (30 board) 10,000 component set of test cases during development. Even so, there was considerable work to be done when the system was brought to the plant for off-line testing as a much greater scene variability was encountered than had been previously observed. The multi-level (macro / micro) tests and the use of software test points and graphics enabled this beta test to proceed rapidly. Even in an ideal situation, the variability of pad surface and component contact appearance caused more trouble than expected, showing the shortcomings of grey-scale analysis for specular objects or objects with surface colour variability and occlusion. An approach more like the SVS 3-D scanner [4] would overcome this, but such an approach would have proved too expensive to implement and perhaps too slow in operation. The 2-D approach eventually proved adequate for the task, but required a great deal of extra development time that a more appropriate (3-D) sensor would have eliminated.

In the future, the the development of a 4 or 5 camera solder joint inspection system and a low cost single axis system for inspecting much smaller PCBs are anticipated. The single axis system will be much smaller and cheaper to produce (less than 40,000 ECU), and could open up new inspection areas which currently cannot afford the technology. At a hardware level, it is hoped to port the system to an EISA-based PC clone with a colour frame grabber which is currently under development. This should further reduce costs while availing of the dynamism of the mainstream PC marketplace. This planned migration away from transputer

technology is regrettable but the pace of transputer development is too slow to keep up with the increasing demands of the application and the capability of its competitors. The EISA bus should allow images to be transferred at 33 MBytes per second which is much faster than that achievable (2 MBytes/second/link) with transputers at present. It is hoped to transfer images to the main Intel i80486 processor, or to other slave processors on the EISA bus for analysis. As long as these processors are programmable in C, porting the code to them should be a realistic proposition.

The work which has been described in this chapter clearly demonstrates that it is feasible to use off-the-shelf technology to solve a non-trivial inspection problem and to do so within stringent reliability and robustness specifications both in terms of speed and accuracy. And it is possible to do so at a reasonable cost.

In conclusion, there are two key points to notice in the context of this book on the issues relating to the design of vision systems. The first is that the necessity to assume the functionality of industrial systems tends to bias the use of image processing and analysis algorithms in favour of the simpler ones. The second, and perhaps more important, point is that the success of a commercial vision system is judged primarily by its demonstrated robustness and repeatability.

1.10 Acknowledgements

This work was funded by Eolas, the Irish Science and Technology Agency, and Apple Computer Ltd., Cork, Ireland.

Bibliography

[1] W. R. Iversen, 'Automated board inspection gets more important', *Electronic World News*, Oct 21, 19–24 (1991).

[2] M. Ejiri, T. Uno, M. Michihiro, and S. Ikeda, 'A process for detecting defects in complicated patterns', *Computer Graphics and Image Processing*, 326–339 (1973).

[3] M. Buffa, 'Process Control for Automated Component Assembly of Surface Mounted Devices Utilising a machine Vision Controller', *SME Proc. Vision '85*, **5**, 180–200, (1985).

[4] R. Kelley, M. Collins, A. Jakimcius, and D. Svetkoff, 'Automatic Inspection of SMD Printed Circuit Boards Using 3-D Triangulation', *SME Proc. Vision '87*, **7**, 33–47 (1987).

[5] M. Rioux et. al. 'Design of a large depth of field 3-D camera for robot vision', *Optical Engineering*, **26(12)**, 1245–1250 (1987).

[6] Control Automation, General product Literature (1989).

[7] D. Trail and A. Jakimcius, 'Inspection Surface mounted PCBs with 3-D Laser Scanners', *Surface Mount Technology*, (April 1988).

[8] N. Zuech, 'Automating Solder-Joint Inspection', *Test & Measurement World*, 68–76, (October 1989).

[9] M. Juha, *The economics of automated X-ray inspection for solder quality*, IRT Corp. Publ., San Diego, CA, (1986).

[10] J. Mahon, N. Harris, and D. Vernon, 'Automatic Visual Inspection of Solder Paste Deposition on Surface Mount Technology PCBs', *Computers in Industry*, **12**, 31–42 (1989).

[11] J. Mahon, 'Automatic 3-D inspection of solder paste on surface mount printed circuit boards', *Journal of Materials Processing Technology*, **26**, 245-256 (1991).

[12] J. Mahon,'Automatic Inspection of Solder Joints on Surface Mount PCBs', *Proc. 7th conf. of the Irish Manufacturing Committee on AMT*, 1990.

[13] C. Kirkpartick, 'Optimisation by Simulated Annealing', *Science*, 671–680, (May 1983).

[14] G. Agin,'Computer Vision Systems for Industrial Inspection and Assembly', *Computer*, 11–20, (May 1980).

[15] W. Silver, 'Alignment and Gaging Using Normalized Correlation Search', *Proc. SME Vision '87*, **5**, 33–56 (1987).

[16] O. Davies and P. Goldsmith, *Statistical methods in Research and Development*, Longman (1986).

Chapter 2

The Role of Context in an Integrated Active Vision System

James L. Crowley

LIFIA (IMAG),
Grenoble,
France.

2.1 Introduction.

In the previous chapter we looked at industrial computer vision systems and we noted that the operational necessity to assume the functional correctness of a system — i.e., that it works — by and large tended to dictate that only simple processing and analysis techniques are incorporated. It is well known that this simplicity is facilitated by heavily engineering the visual environment which is being analysed and by forcing strong constraints upon it. But this cannot always be done and sometimes we are forced into using much more sophisticated techniques such as is the case when attempting to achieve 3-D object recognition and manipulation.

This chapter describes techniques for object recognition within a continuously operating active vision system. The first section presents the viewpoint of active vision,[1] and discusses the problems that occur in the implementation

[1] Active vision is characterized by the exploitation of control of image formation and control of processing in order to make vision fast and robust. In ESPRIT project BR 3038 'Vision as

of a scene interpretation component in such systems. Of particular importance are problems associated with stability, robustness and computational complexity. These considerations are shown to lead naturally to a system in which interpretation is performed by a hypothesis and verification mechanism.

Section 2.3 presents the framework for the problem by describing the architecture of the VAP/SAVA skeleton system. This system permits us to compose an active vision system from a number of independent processes. Each process maintains a description of the scene at a particular level of abstraction. We present a standard module for processes that describe the scene. This standard module is used to construct processes to track edge lines in images, to maintain a 3-D model of a scene, and to maintain the interpretation of the scene.

The construction of the interpretation module within the standard description module is described in section 2.4. Interpretation is performed by a set of recognition procedures in the form of rules written in CLIPS 5.1. Recognition procedures are triggered by demons that watch the contents of the scene for specific events. The detection of an event triggers the relevant procedures. These procedures can make or update a hypothesis or change the current interpretation context. Demons provide a pre-attentive recognition mechanism.

The generation and maintenance of a description of a scene requires geometric reasoning. In our system, geometric reasoning is provided by a vocabulary of perceptual grouping procedures which provide a form of associative data access to the geometric scene descriptions. Data access by grouping provides a form of pure 'post-attentive' processing. A taxonomy of four classes of model access and grouping procedures is described is section 2.5. Some conclusions and extensions are discussed in section 2.6.

2.2 Overview

The present situation in computer vision may be compared to the early years of aviation, in the period between 1870 and 1900. During this period, a number of prototype airplanes were built. However they all shared the same defect: there was no way to control the flight surfaces dynamically. These early prototypes relied on static stability. The innovation introduced by the Wright brothers was dynamic control of the aircraft. I argue that trying to see with a static camera is like trying to fly an airplane with no control surfaces. As with flight, vision must be controlled as a dynamic process.

This point of view changes the nature of the problem of machine vision in the following ways:

Process' we have extended this principle to incorporate object recognition and 3-D modelling within an 'integrated' system: that is, a system which integrates robotic camera control, 2-D image description, 3-D image description, and object recognition. Context plays an important role in both control and processing in such a system.

1. The image formation process is to be controlled on the basis of the current goals of the system.

2. For real-time operation, the amount of information taken from any one image is limited to what can be extracted during a fixed time period.

3. The interpretation of any one image is simplified by the 'context' of the interpretations of the previous images.

4. Discrimination of noise and signal is made easier since noise tends to be transient and unstable. The desired signal is the information which is stable with respect to changes in viewpoint, lighting, and changes in the scene.

During the last few years, there has been a growing interest in the use of active control of image formation to simplify and accelerate scene understanding. Basic ideas which were suggested by [3] and [1] have been extended by several groups [6], and [21]. Brown [11] has demonstrated how multiple simple behaviours may be used for control of saccadic, vergence, vestibulo-ocular reflex and neck motion.

This trend has grown from several observations. For example, Aloimonos and others observed that vision cannot be performed in isolation. Vision should serve a purpose [1], and in particular should permit an agent to perceive its environment. This leads to a view of a vision system which operates continuously and which must furnish results within a fixed delay. Rather than obtain a maximum of information from any one image, the camera is a sensor by which an agent observes a scene. Any one image is a signal which provides only limited information about the scene.

Bajcsy [3] observed that many traditional vision problems, such as stereo matching, could be solved with low complexity algorithms by using controlled sensor motion. Examples of such processes were presented by Krotkov [26]. Ballard and Brown [5] demonstrated this principle for the case of stereo matching by restricting matching to a short range of disparities close to zero, and then varying the camera vergence angles.

These ideas have led to a sub-field which identifies itself with the name 'active vision'. The active vision approach can be defined as the exploitation of the control of camera parameters and the control of processing to make the robust real-time visual observations which are necessary for subsequent control of, e.g., robotic devices. Recognition or reconstruction are important only if they are useful for making the observations which are necessary for such control. An active vision system operates continuously and provides its results in real-time. If the system is not a special-purpose one, it must have the ability to switch between different processing modalities. This implies the need to control the processing of the system. Our goal in ESPRIT project BR 3038 – 'Vision as Process' – has been to extend the active vision approach to higher levels of a

computer vision system. In particular, we have explored contol of perception for a system which integrates robotic camera control, 2-D description, 3-D reconstruction, and object recognition. Integrating these levels within a vision system allows the system to solve vision problems at the level which is most appropriate to the problem.

Such a view-point leads us to propose two essential scientific problems:

1. The study of the integration of separate visual processes which operate at the level of signal, scene, algorithmic control and symbolic control into a cooperating system, and

2. the problem of control of perception in a continuously operating system.

The project 'Vision as Process' has been concerned with both of these problems. In this chapter, we discuss how the active vision approach has been applied in an object recognition process which operates continuously. This system is based on the well-known principle of prediction and verification and has been demonstrated in the interpretation of breakfast-table settings composed of plates, cups, and silver-ware. Typical tasks include 'find the plate' and 'watch the cup'. Recently, we have begun experiments using the same system for mobile robot navigation.

It is important to note that, in this chapter, we will not address the total computer vision problem and its complete solution. Instead, we would like to put forward some ideas on the type of algorithmic architecture which is required to implement control and recognition within active vision systems. This computational/algorithmic infrastructure is the skeleton of the system which will be the more complete solution of the above-mentioned problems of process integration and real-time dynamic control. As the description of the system unfolds in this chapter, we will note where appropriate the type of additional processing and representations which we are currently using, or anticipate using, in fleshing out this skeleton into a full 3-D scene interpretation system.

2.2.1 Scene Interpretation Techniques

The vision community has known for years that *a priori* expectations can be used to greatly simplify the interpretation process. This was the idea behind Minsky's *Frames* [30] as well as a many other well known 'top-down' interpretation systems. The system we describe exploits 'top-down' expectations to direct its interpretation, while 'bottom up' processing is used to trigger interpretation hypotheses and to shift the current interpretation context. The interface between goal-directed and data-directed processing is at the level of the image model.

Most research in scene interpretation is still directed at interpretation of single images or static scenes. Few groups are working on the problem of scene interpretation within a dynamically changing scene. Scene interpretation is

typically based on direct geometric matching to prototypes, as described by [6], [4] and [22]. A group at Stanford University [27] has adopted a strategy where all processing is based on a Baysian formalism, while Jensen *et al.* [24] only uses the Baysian formalism for the interpretation step.

Techniques have been reported in [7] and [19] for geometric scene or object modelling based on a decomposition into a set of geometric primitives (or GEONS [8]). Such techniques seem capable of decomposing objects, described in terms of contours and vertices, into a finite set of geometric primitives (modal models, as described in [34]). Unfortunately, the tests reported with these techniques have so far been based on synthetic data.

The interpretation process presented below is based on the representation of scene knowledge in terms of rules, encoded in the C language production system CLIPS 5.1. This system is inspired by the SCHEMAS system developed by Draper [20]. Surveys of the use of knowledge directed vision can be found in [9] and [37]. For example, Nagao [31] constructed an interpretation system in which the subsystems were organised as rules while Ohta used a more traditional production system for interpretation [32]. The SPAM system [29] was based on over 500 rules written in OPS-5. Glicksman used a frame-based knowledge representation in his human-aided interpretation system. The Codger system [35] used a black-board system in which continuous and concurrent knowledge sources could post conflicting interpretations.

2.2.2 The Scene Interpretation Problem

In our system, scene interpretation is a process of generating symbolic hypotheses about the existence of objects in the scene. Hypotheses are generated and confirmed as groupings of tokens in a dynamically-maintained geometric description. The tokens in the current system correspond to edge segments, edge-chains, and edge ellipses expressed in image coordinates. Object hypotheses represent known classes of objects, and a label may be accompanied by a set of properties. In our first version of the system, the dynamically-maintained description models the contents of images in terms of 2-D edge segments and elliptical arcs at different resolutions. Edge segments are tracked to maintain temporal continuity; arcs are not currently tracked.

The interpretation process is complicated by three problems:

1. The edge-segments in the 2-D description that compose instances of known objects are embedded in a large number of edge-segments from other sources.

2. The edge-segments are corrupted by various phenomena. The existence of segments is uncertain and the parameters of segments are corrupted by random noise.

3. The 2-D edge-segments evolve dynamically. Segments are continuously added and removed from the model scene or the region of interest changes.

The first problem can potentially cause a combinatorial explosion for interpretation [38]. This has led us to use a 'hypothesis and verification' technique. In this technique, a recognition procedure is invoked on the basis of the current context. This recognition procedure generates hypotheses about objects in the scene and about their composition from groups of primitives in the 2-D model.

A brute force matching strategy that explores all possible combinations is an N-P complete process. Our strategy is to provide a large vocabulary of perceptual grouping procedures that may be used to interrogate the 2-D image description. These procedures are called on the basis of *a priori* expectations.

A continuously operating vision system must limit the size of internal models to manage computational cost. We have imposed the constraint that our system must operate with a fixed cycle time. In order to guarantee such a cycle time, we must (a) Limit cost and the complexity of the procedures that are used, and (b) Limit the amount of data treated in any one cycle. We have designed the grouping procedures in order to minimize computational complexity, as well as the actual execution time.

Grouping procedures are part of a larger vocabulary of model access functions. These functions are organised in four classes, according to the nature and complexity of the operation that they perform. Grouping operations from the first two classes have complexities that are dependent on the number n of primitives in the model. Grouping primitives from the third class have a quadratic complexity $O(n^2)$, while primitives from the fourth class have complexities that are $O(n^3)$. These procedures may be composed such that a $O(n)$ grouping procedure can be used to select the subset of model primitives that are considered by a more expensive primitive.

The second problem requires techniques for dealing with uncertainty. Hypotheses of objects are accompanied by a confidence factor. Interrogation of the 2-D description yields a confidence factor, based on the similarity of the predicted and observed primitives. By combining evidence, missing components degrade the confidence of a label, but do not prevent its assignment. The hypothesis and verification process permits us to evaluate conflicting hypotheses in parallel.

The third problem is that the description that is interpreted is dynamically changing. Geometric primitives are continuously added and deleted from the description. In our first experiments, object component hypothesis continually generated grouping interrogation requests. The results were that the communication channel between the modules was very quickly saturated, and that both the interpretation and description modules spent much of their computational 'budget' on handling request messages. In order to reduce this message traffic a scheduling mechanism for grouping 'demons' was added to the description modules. These demons are scheduled to be executed after each update phase

of the predict-match-update cycle within the description module. Each demon calls a grouping primitive or a composition of grouping primitives in order to detect a certain configuration of primitives. If the demon detects the primitives, a message is sent to the interpretation process. If in a later cycle, the demon ceases to detect the primitives, or if the confidence of the grouping changes by more than a threshold, a new message is sent to the interpretation process.

The following section presents the overall architecture of this system. This is followed by the presentation of a standard module for processes in the system.

2.3 An Integrated Continuously Operating System

Our continuously operating vision system is composed of a number of independent processes that dynamically maintain a description of a limited region of the scene at different levels of abstraction. This section discusses the structure of this system and presents the design of standard module architecture. We conclude by presenting the representation for the 2-D edge-segment description of images from the left and right cameras. This representation is necessary for the grouping procedures described in section 2.5. Note again that we are less concerned here with the total computer vision problem and its complete solution and more with the algorithmic architecture which is required to implement control and recognition within active vision systems.

2.3.1 The SAVA Skeleton for a Distributed Vision System

The techniques described below have been built for a continuously operating vision system built up from distributed processes, as shown in figure 2.1. This system includes independent processes for camera control, image acquisition and tracking, 3-D modeling, symbolic interpretation, and system supervision. Each module is a continuously-running cyclic process, implemented as a separate Unix process. This system has been built for experiments in integration and control of a continuously operating vision system.

Descriptions of image information are maintained for both the left and right stereo cameras. At the current time these descriptions are maintained by tracking edge-segments from a multi-resolution region of interest. The 3-D scene description module uses information from the 2-D model to infer the 3-D form and position of structures in the scene. The symbolic scene interpretation module uses information from the 2-D image description and the 3-D image description to describe the scene in terms of known objects. This chapter is concerned with techniques for interrogating the 2-D model to aid interpretation. Although these techniques have been developed for the 2-D model, we have since begun to integrate 3-D perceptual grouping into the interpretation. In order to perform experiments in integration and control, we have developed a skeleton system

whose components are shown in figure 2.1. This system, named SAVA, is composed of a number of UNIX processes which communicate by a mail box system as well as by sockets. Experiments in integration or control are performed by configuring a vision system from subsets of these modules. In this chapter, we illustrate the role of context in recognition and control and we will restrict our presentation to the techniques that permit the interpretation module to interrogate and label structures found in the image description modules.

2.3.2 The Standard Module for Image and Scene Description

The modules for maintaining a 2-D description, for maintaining a 3-D description, and for maintaining a symbolic interpretation are all examples of 'scene description modules'. We have developed a standard architecture for such modules. At the heart of this standard module is a process based on the cycle: *predict-match-update.*

A continuously operating vision system can be constructed from a cyclic process as shown in figure 2.2. In such a process, observations are continuously acquired and integrated into a description (or model). The integration process operates by predicting the state of the model at the time of the observation. The predicted and observed information are then matched, and the result is used to update the model. In such a process the prediction from the previous model serves both to accelerate the description of additional observations, and to reduce the errors in the description. The cyclic process presented in figure 2.2 is a form of tracking. This process has been used extensively for world-modeling with ultra-sound [16], for tracking 2-D edge lines [16], and for maintaining 3-D models [17]. For numeric information such as 2-D or 3-D description, this tracking may be based on techniques from estimation theory [10].

The components of this standard module are shown in figure 2.3. This standard module is composed of a number of procedures (shown as rectangles) that are called in sequence by a scheduler. Between each procedure call, the scheduler checks a mail box to see if any messages have arrived. Such messages may change the procedures that are used in the process, they may change the parameters that are used by the procedures, or they may interrogate the current contents of the description that is being maintained. The main topic here is the procedures by which such interrogation is performed.

At the end of each cycle, the scheduler calls a set of demon procedures. Demons are responsible for event detection, and play a critical role in the control of reasoning. Some of the demon procedures, such as motion detection, operate by default, and may be explicitly disabled. Most of the demons, however, are specifically designed for detecting certain types of structures. These demons are armed or disarmed by recognition procedures in the interpretation module according to the current interpretation context.

Figure 2.1: A Distributed Multi-process Vision System.

Figure 2.2: The components of Continuously Operating Description Process.

We have defined a standard protocol for model interrogation based on the procedures 'Find', 'Verify', and 'Get'. This protocol uses the model access and perceptual grouping procedures described below in section 2.5. The role of the standard access functions is illustrated by figure 2.4.

The interrogation procedures have the following form:

<List of IDs> ← Find <Grouping Primitive> <Parameters> <ROI>

<List of Parameters> ← Get <ID>

<CF> ← Verify <Grouping Primitive> <Parameters>

The procedure 'Find' searches for class of structures in the module using a set of parameters and a region of interest. 'Find' returns a set of identities. 'Get' permits access to the parameters of a token based on its ID. 'Verify' returns a confidence factor for the existence of a primitive or grouping with a set of parameters.

Figure 2.3: Architecture of a Standard Module for image or scene description.

Figure 2.4: The Model Access Primitive: Find, Get and Verify.

2.3.3 Parametric Representation for Line Segments

Our first experiments with the standard module involved tracking and interpreting 2-D edges. More recently, we have implemented tracking and interpretation of 3-D edges obtained by stereo matching. In this section we will illustrate the process by describing tracking and grouping for 2-D and 3-D information.

The predict-match-update cycle for edge-line segments can be simplified by using a representation composed of parameters for the mid-point, half-length and spatial extent [16]. Such a representation is also useful for many of the matching operations involved in grouping and model access.

Tracked edge lines in our 2-D description are expressed by a set of four primary parameters, and a set of dependent parameters, as shown in figure 2.5. The primary parameters for an edge line are given by the vector $S = (P_m, \theta, h)$, where:

P_m is the mid-point, expressed as (x, y),
θ is the orientation of the segment, defined on the interval [0, 2π], and
h is the half-length of the segment.

Each of these parameters is accompanied by a temporal derivative and a covariance maintained by tracking [16]. A number of dependent parameters are used in many of the grouping operations. Such parameters are computed when needed from the primary parameters and then saved within the edge segment. These parameters are:

P_1, P_2	The end-points.
A, B	Line Equation Coefficients ($A = \sin\theta, B = -\cos\theta$)
C	Perpendicular distance to the origin
D	Distance from projection of origin to mid-point

Segments also include a confidence factor, CF $\in \{1, 2, 3, 4, 5\}$, and a unique integer identification number, ID. A similar representation has been used for 3-D edges [18]. The scene model and observations are expressed as 3-D segments represented by both a midpoint-direction-length representation and by the end-points as shown in figure 2.6. As with 2-D segments, each 3-D segment has associated with it a confidence factor, an indication of how recently it has been included in the model, and a unique identification number ID. The ID label is inherited from the ID label which is assigned by the tracking process of the left camera to the 2-D token on which the 3-D segment is based. The minimal parameters for a segment are:

P	The centre point of the segment (x, y, z)
C_P	The 3×3 covariance of the centre point
ΔP_s	An un-normalized direction vector
CF	The segment confidence factor
ID	The identification number of the segment (derived from the 2-D model)

Each 3-D segment also contains a number of redundant parameters as shown in figure 2.6:

P_1	The position of the first end-point
C_1	The 3-D covariance in the position of the first end-point
P_2	The position of the second end-point
C_2	The 3-D covariance in the position of the second end-point
D	The normalized direction (expressed as Δx, Δy, and Δz, normalized to unit length)
C_D	The 3×3 covariance of the direction
H	The half-length of the segment

A set of redundant parameters are used to verify the match of observed segments to the corresponding composite model segment. The minimal parameters are initially calculated from the 3-D end-points P_1, P_2, and their covariances, C_1 and C_2, by:

$$P_s = \frac{(P_1 + P_2)}{2} \tag{2.1}$$

$$C_P = \frac{(C_1 + C_2)}{4} \tag{2.2}$$

$$\Delta P_s = \frac{(P_2 - P_1)}{2} \tag{2.3}$$

From which we can calculate the half-length and unit direction vector and its covariance as:

$$H = ||\Delta P_s|| \tag{2.4}$$

$$D = \frac{\Delta P_s}{||\Delta P_s||} \tag{2.5}$$

$$C_D = \frac{(C_1 + C_2)}{||\Delta P_s||^2} \tag{2.6}$$

The covariance of the direction, C_D, is an approximation of the uncertainty angle with respect to each of the axes.

The parameters and uncertainty of a 3-D segment are refined by multiple observations. The 3-D reconstruction algorithm exploits the segment identities which are maintained by 2-D tracking. The 2-D segment IDs are combined to form a 3-D segment ID. A reconstructed 3-D segment can be directly associated with a segment in the 3-D model. Correspondence is verified by comparing 3-D segments for similar orientation, co-linearity, and overlap. The 3-D segment parameters are then updated using Kalman filter equations [18].

The 2-D predict-match-update cycle is based on a first-order Kalman filter. The 3-D system uses only a zeroth order Kalman filter; that is, there is no estimate of velocity. In both systems, the contents of the model are managed using an integer 'confidence factor'. Each time a segment is observed, the confidence factor is incremented by one, up to a maximum value. When the segment is not observed during an update, the confidence factor is decremented. If the CF of a segment drops below the value of 1, the segment is removed from the composite model. Thus, a segment must be present in at least five observations in order to be considered reliable enough to be preserved in the model. For symbolic labelling, a somewhat different set of tools are required.

In addition to tracking of 2-D and 3-D descriptions, the image acquisition and processing module also contains an ellipse extraction procedure. On demand, edge chains are processed by the ellipse extractor to provide a list of ellipses with their parameters. Ellipses have proved very useful in detecting cylindrical objects, such as cups and plates.

2.4 Continuous Interpretation by Recognition Procedures

The interpretation module may be constructed using the same predict-match-update cycle as other scene description modules. Nonetheless, the nature of the scene description as well as the nature of the procedures that produce it, are somewhat different (see figure 2.7).

The objects that can be recognized by the system are described in terms of object schema. The structure of these representations is inspired by classical work on structured knowledge representation and are exemplified in object-oriented programming languages.

Each object schema is composed of:

Figure 2.5: Midpoint-Distance-Length representation for Edge Lines.

Figure 2.6: Representation of a 3-D segment and its associated uncertainties.

Properties A data structure composed of 'slots' which can be used to represent properties of the object.

Procedures A set of procedures that can detect instances of the object and to post an object hypothesis to the interpretation model.

A scene description is a list of object instances. Each object instance contains a list of properties for an object in the scene. The properties are dynamically updated by the recognition procedures associated with that object class. Object instances contain a unique ID and a confidence factor. Each object class is associated with a number of recognition procedures.

A recognition procedure is composed of three parts (see also figure 2.8):

Trigger A test that causes execution of the procedure.

Condition A set of measurements based on interrogation of one or more of the description modules.

Action A set of actions taken by the procedure, including posting an object hypothesis, updating an object's properties or changing the current context.

The system's knowledge base about objects is organised as a network of contexts. Each context is a list of object classes and their associated recognition procedures. For each recognition procedure, a message is sent to the appropriate demon in one of the description modules (2-D, 3-D or interpretation).

When a demon detects one or more instances of its event, it sends a message to the interpretation module with information about the events. For each instance, the trigger places one or more 'rules' on an agenda. A demon continues to monitor the event in subsequent cycles without generating new messages. If the demon ceases detect its event, a new message is set to 'de-activate' the recognition procedure.

The condition part of a recognition rule is composed of a set of calls to the relevant model using the primitives 'Find', 'Verify' and 'Get'. These recognition rules interrogate the various description modules using the model interrogation and grouping procedures, and then create or update object hypotheses based on the result. The expressive power of the recognition procedures is based on a vocabulary of primitives for interrogation and grouping of the data in the 2-D, 3-D and descriptive modules.

2.5 Procedures for Model Interrogation and Perceptual Grouping

This section presents the procedures for interpreting the 2-D image descriptions from the left and right cameras. Similar procedures are planned for access and

Figure 2.7: (a) The interpretation knowledge is represented by object classes. (b) The scene description is a hierarchical composition of instances of objects.

grouping for the 3-D description module, and for determining spatial relationships within the interpretation module. These procedures (2-D and 3-D) are organised in four classes according to their computational complexity and the nature of the interrogation. This library of procedures continues to grow as we develop additional recognition procedures.

These procedures for perceptual grouping are based on calculations that use the numerical attributes of symbolic tokens maintained by tracking. These numerical attributes provide a means for a low computational-cost grouping that is more stable and less expensive than techniques that use the connectivity of symbols.

Model interrogation procedures may be organized into a taxonomy according to the nature of the interrogation that they make. We have identified four classes of interrogation procedures:

Class 1 Procedures for extracting parametric primitives that are *close to an ideal geometric entity* such as a line or a point. Such primitives are easily implemented and have a complexity that is proportional to the number of primitives in the model.

Class 2 Procedures for extracting parametric primitives that are *similar to an ideal 'prototype'*. The prototype is given an estimated value and a standard deviation for each of the primitive parameters. Similarity is measured by

Figure 2.8: The interpretation process. Recognition procedures are triggered by demons. When triggered, a recognition procedure uses perceptual grouping to interrogate the 2-D image description.

a Mahalanobis Distance. Parameters may be labeled as a 'wild-card' by specifying a large (or infinite) standard deviation.

Class 3 Procedures for extracting *pairs of primitives* that satisfy some geometric relation, such as a junction, an alignment, or a parallelism. The geometric relation is defined as a set of parameters and specified as a estimation and a standard deviation. These procedures have theoretical complexity of $O(n^2)$.

Class 4 Procedures for extracting *grouping of groupings.* If performed by brute force, complex forms such as lines and contours composed of sets of line segments can require an exponential number of computations. By structuring the interrogation as a grouping, we can limit the theoretical complexity to $O(n^2)$ or $O(n^3)$.

In addition to these four classes, it is also possible to interrogate the model using a composition of groupings. That is we can apply the grouping operations in sequence, for example, extracting all junctions of a certain form that are near an ideal line. Thus the $O(n)$ primitive can be used to restrict the number of entities interrogated (the value of n) by an $O(n^2)$ or $O(n^3)$ procedure.

The result of an interrogation is a list of segments or groups of segments that satisfy the specified geometric property. These segments or groups are accompanied by a 'quality factor' based on the similarity between the requested and observed entity, the list of IDs for the entities that satisfy the relation is returned to the calling module, and a small set of properties of the grouping. This list is sorted based on the quality factor.

The grouping operations described above were initially implemented with 2-D edge lines. For each of these classes we have found natural extensions to 3-D. For example, in class 1 we have defined an extraction for 3-D segments near a 3-D point, line, or plane. In class 2, we have defined functions to detect 3-D segments that are similar to a prototype. In class 3, we have defined procedures for 3-D junctions, parallelism, and alignment. In class 4, we can detect groupings of groupings.

2.6 Conclusions

In this chapter we have described a system for actively maintaining an interpretation of a scene. This system is part of a larger distributed system that actively describes a scene in terms of 2-D images, 3-D geometry and a symbolic labeling. The system described is a skeleton system which has been constructed for experiments in integration and control. This chapter is mainly concerned with the interpretation part of the system.

The scene interpretation technique uses rule-based 'recognition procedures'. These rules are organised by contexts that provide the set of interpretations that

the system may impose on a scene at a given moment. Recognition procedures are triggered by demons that watch the contents of the continuously updated scene descriptions for certain events. The detection of such an event triggers a rule that will then make a measurement in one of the descriptions and then make or modify a hypothesis in the image interpretation.

Interpretation requires geometric reasoning. Such reasoning requires frequent access to the geometric data. We have extracted the geometric portion of the interpretation into a vocabulary of procedures for associative model access and model grouping. We present a natural taxonomy for such procedures in terms of four classes, based on the nature of the data access and the complexity of the computation. The list of procedures which we describe is not exhaustive. We continue to construct additional grouping procedures as we write recognition procedures.

Some previous efforts at perceptual grouping had based the computations on the use of graph representation for the edge segments [39]. Our experience is that such representations are inappropriate for continuous operation. In any edge-based description of an image there are always gaps along edge segments and at junctions. Closing such gaps requires the specification of a tolerance. Such a tolerance will depend on the contents of the scene, the lighting, and the optical parameters of the cameras, and thus must be actively controlled. Our experiments have convinced us that it is more appropriate to treat the edge segments as a list of property vectors and to optimize the tests that are applied to this list.

Dynamic vision requires pre-attentive grouping for *event detection* as well as post-attentive grouping for *interpretation* [36]. We have found that such pre-attentive event detection may be performed by arming a set of demon procedures that search the dynamically updated model for desired events. Some demons are armed by default, other are armed by request. We are currently trying to understand better the relative roles and computational costs of these different kinds of demons.

All of the above procedures are based on edge segments that are inherently noisy. Another line of work in our group involves by-passing the extraction of edges and directly measuring curves and junctions in the image data. Koenderink et al. [25] has shown families of receptive fields for such image events as corners and T junctions. We have recently implemented such operations in order to directly extract junctions from the image data. We have also begun to experiment with extending our edge detector to extract curves.

2.7 Acknowledgments

This work was performed in collaboration with University of Aalborg (DK), University of Surrey (UK), University of Linkoping (S), Swedish Royal Institute of Technology - KTH (S), and the LTIRF - INPG (F). This work has been

sponsored by CEC DG XIII ESPRIT Basic Research grant BR 3038 'Vision as Process'.

Bibliography

[1] J.Y. Aloimonos, I. Weiss, and A. Bandopadhay, 'Active Vision', *International Journal on Computer Vision*, 333–356 (1987).

[2] Y. Aloimonos and D. Shulman, *Integration of Visual Modules*, Academic Press, Orlando, Fla. 1989.

[3] R. Bajcsy, 'Active Perception', *IEEE Proceedings*, **76(8)**, 996-1006 (1988).

[4] R. Bajcsy and D. Rosenthal, 'Visual and Conceptual Focus of Attention', in *Structured Computer Vision*, S. Tanimoto and A. Klinger (Eds.), Academic Press, New York, N.Y., (1980).

[5] D.H. Ballard and A. Ozcandarli, 'Eye Fixation and Early Vision: Kinematic Depth', *IEEE 2nd Intl. Conf. on Comp. Vision*, Tarpon Springs, Fla., 524-531 (Dec. 1988).

[6] D. Ballard, 'Animate Vision', Artificial Intelligence, **48(1)**, 1–27 (1991).

[7] R. Bergevin and M. Levine, 'Generic Object Recognition: Building Coarse 3-D Descriptions from Line Drawings', *Computer Graphics Vision and Image Processing*, Submitted May 1989.

[8] I. Biederman, 'Matching Image Edges to Object Memory', International Conference on Computer Vision, London, (1987).

[9] T. Binford, 'Survey of Model Based Image Analysis Systems', *International Journal of Robotics Research*, **1(18)** (1982).

[10] K. Bramler and G. Siffling, *Kalman Bucy Filters*, Artech House Inc., Norwood MA, USA (1989).

[11] C. Brown, 'Prediction and Cooperation in Gaze Control', *Biological Cybernetics* **63** (1990).

[12] A. Califano, R. Kjeldsen, and R.M. Bolle, 'Data and Model Driven Foviation', 10th International Conference on Pattern Recognition, Atlantic City, (1990).

[13] Chehikian A., 'A One Pass Edge Exctraction Agorithm' 7th Scandanavian Conference on Image Analysis, Aalborg, (1991).

[14] J. Clark and N. Ferrier, 'Modal Control of an Attentive Vision System', IEEE 2nd Intl. Conference on Computer Vision, Tarpon Springs, Fla., 514–523, (1988).

[15] Crowley, J.L., 'Coordination of Action and Perception in a Surveillance Robot', *IEEE Expert*, **2(4)**, 32–43 (1987).

[16] J.L. Crowley, P. Stelmaszyk, and C. Discours, 'Measuring Image Flow by Tracking Edge Lines', 2nd International Conference on Computer Vision, Tarpon Springs Florida, (1988).

[17] J.L. Crowley, 'Towards Continuously Operating Integrated Vision Systems for Robotics Applications', Seventh Scandinavian Conference on Image Analysis, Aalborg (1991).

[18] J.L. Crowley, P. Stelmaszyk, T. Skordas, and P. Puget, 'Measurement and Integration of 3-D Structures By Tracking Edge Lines', *International Journal of Computer Vision*, July (1992).

[19] S. Dickinson, A. Pentland, and A. Rosenfeld, 'Qualitative 3-D Shape Recovery using Distributed Aspect Matching', University of Maryland, Technical Report CAR-TR-505 (1990).

[20] B.A. Draper, R.T. Collins, J.Brolio, A. R. Hansen, and E. M. Riseman, 'The Schema System', *International Journal of Computer Vision*, **2(3)**, (1989).

[21] J.O. Eklundh and K. Pahlavan, 'Eye and Head-Eye System', SPIE Applications of AI X: Machine Vision and Robotics, Orlando, Fla. April 92 (to appear).

[22] R. Fisher, 'The design of the IMAGINE II Scene Analysis Program', in *3-D Model Recognition from Stereoscopic Cues* J.E.W. Mayhew and J. P. Frisby (eds.), MIT Press, Boston, Mass., 239–245 (1991).

[23] E. Grimson, *Object Recognition by Computer*, MIT Press, Boston, Mass. (1990).

[24] F.V. Jensen, J. Nielsen, and H.I. Christensen, 'Use of Causal Probabilistic Networks as High Level Models in Computer Vision', Technical Report R-90-39, Institute of Electronic Systems, Aalborg University, (1990).

[25] J.J. Koenderinck, 'Local Image Structure', Proc. 7th Scandinavian Conference on Image Analysis, Aalborg (1991).

[26] E. Krotkov, 'Focusing', *International Journal of Computer Vision*, **1**, 223-237 (1987).

[27] T.S. Levitt, T. Binford, G. J. Ettinger, P. Gelband, 'Probability based Control for Computer Vision', DARPA Image Understanding Workshop 1989, Palo Alto, Ca., 355-369, (1989).

[28] D.G. Lowe, *Perceptual Organization and Visual Recognition*, Kluwer Academic Press, Boston (1985).

[29] D.M. McKeown, W. A. Harvey, and J. McDermott, 'Rule Based Interpretation of Aerial Imagery', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **7(5)** (1985).

[30] M. Minsky, *A Framework for Representing Knowledge*, MIT AI Memo 306 (1974).

[31] M. Nagao, and T. Matsuyama, *A Structural Level Analysis of Complex Aerial Photographs*, Plenum Press (1980).

[32] Y. Ohta, 'A Region Oriented Image analysis system by Computer', Ph.D. Thesis, Dept. of Information Sciences, Kyoto University (1980).

[33] A. Pentland, 'A New Sense for Depth fo Field', IEEE Transactions on Pattern Analysis and Machine Intelligence, (1987).

[34] A. Pentland and S. Sclaroff, 'Closed-form solutions for shape modelling and recognition, MIT Media Lab Technical Report 135, Boston, Mass. (1990)

[35] Thorpe, C.S. Shafer, and T. Kanade, 'Vision and Navigation for the Carnegie-Mellon Navlab', IEEE Expert, (1987).

[36] A. Triesman, 'Features and Objects', *The Quarterly Journal of Experimental Psychology*, **40A(2)** (1988).

[37] J. Tsotsos, 'Image Understanding', *The Encyclopedia of Artificial Intelligence*, S. C. Shapiro (Ed), Wiley, New York (1987).

[38] J. Tsotsos, 'A Complexity Level Analysis of Computer Vision', *International Journal of Computer Vision*, **1(4)** (1988).

[39] R. Horaud, F. Veillon, and T. Skordas, 'Finding Geometric and Relational Structures in an Image', Proceedings of the First European Conference on Computer Vision, Springer Verlag, Antibes, April (1990).

Chapter 3

From numerical to symbolic image processing: Integration of computational and knowledge-based approaches

G. Vernazza

Dept of Biophysical and Electronic Engineering (DIBE),
University of Genova,
Italy.

3.1 Introduction

Having looked first in chapter one at the design of an industrial vision system to perform a reasonably straight-forward inspection task and in chapter two at the design of a much more complex active vision system which is potentially capable of dealing with a much less structured environment, we turn our attention now to the more general issues associated with the requisite architectures of sophisticated visual systems.

One of the most interesting goals of computer vision is *image understanding*, that is, the construction of intelligent systems for extracting a more or less detailed description of a scene from images[1]. In the Encyclopedia of Artificial Intelligence [2] we find the following statement about the purpose of an image understanding system: 'The goal of an image understanding system is to transform 2-D data into a *description* of the 3-D spatial temporal world: such system must *infer* 3-D surfaces, volumes, boundaries, shadow, occlusions, depth, colour, motion'. This definition, even though restricted to 2-D data, can also be applied to the understanding of tomographic images, 3-D voxel structures, spatial-temporal image sequences, etc. It contains two key words:

1. *infer:* the image acquisition process introduces a gap between input data and the world's properties, and such a gap needs to be bridged by some appropriate technique (e.g., by using the general perspective theory, we can obtain a 3-D interpretation from the analysis of 2-D lines);

2. *description:* the result is a more of less detailed or abstract description of the world.

The application domains of image understanding systems are numerous and different; for example: robotics, autonomous vehicle guidance, medical images, and remote sensing. The purpose of this chapter it to address the image understanding problem, in the context of complex real environments. In particular, the focus is on the above-mentioned process of inferring the world's properties from image data, on the organization of such a process, and on the techniques and approaches that should be used. This topic is debated by various researchers who propose alternative approaches such as that described in the previous chapter. The position which we put forward in this chapter is that the integration of techniques developed by different research groups may allow one to build very powerful systems that, like the human visual system, can exploit many aspects of the visual information available.

3.2 The Image Understanding Process

The development of systems which are capable of simulating the visual and cognitive ability of biological systems is deceptively difficult. It is deceptive because the data which are to be analysed by the computer system are almost always presented to the human designer in the manner which is easiest for him or her to understand, i.e., as images, and the inherent human interpretive processes remain implicit. As Horn has remarked, 'The subtlety and difficulty of describing the exact operation of subconscious functions presents significant difficulty in developing algorithms to emulate human visual behaviour' [4]. Nonetheless, it is widely recognized at present that the understanding process consists in performing a set of progressive transformations of an input image into its final

description, via intermediate levels of abstraction. At successively higher levels, the description of image information is made more and more expressive and explicit while disregarding lower level details, by using appropriate mechanisms (tools). For example, a possible structure for an image understanding process is shown in figure 3.1. Here, several levels of abstraction, or representation, are shown, beginning with simple pixels where all information is implicit, and concluding with a symbolic description of the scene where all information is explicit. The transition from one level to the next is made, for example, by processes of segmentation and classification.

In essence, to perform the transformations at the various levels, three main components are fundamental to the image understanding process:

- data structures for representing abstraction levels;
- algorithms for moving from one level to the next one;
- techniques for managing components (1) and (2).

Two main approaches to the image understanding problem have been presented in the literature. First, there is *the computational approach* where emphasis is placed on recovering information distorted by the image formation process (i.e., lower abstraction levels are paramount). Second, there is *the knowledge-based approach* where emphasis is placed on a symbolic description and processing of the extracted structures or patterns of the related models for interpretation of the environment (i.e., higher abstraction levels are used)

3.3 The Computational Approach

According to Horn, 'A system that successfully interacts with the environment can be understood, in part, by analyzing the physics of this interaction. In the case of vision, this means that one ought to understand image *formation* if one wishes to recover information about the world from images. Modeling the *physical* interaction naturally leads to equations modeling that interaction. The equations, in turn, suggest *algorithms* for recovering information about the 3-D world from image'[4]. Therefore, in the computational approach, emphasis is put on the 'early vision' stages, that is, on the analysis of the image formation process by making physical and mathematical studies and by using numerical algorithms.

The main characteristics of the computational approach are that it responds 'quickly' to the environment and that it is not based on any 'explicit representation' of the world on which it operates.

Many highly-qualified researchers have developed this approach, such as M. Brady, W. Grimson, B. Horn, D. Marr, and T. Poggio. They emphasize in much of their work the development of techniques for achieving single visual abilities

Figure 3.1: Computerized image understanding process.

(e.g., edge detection, 'shape from x', etc.) rather than design complete computational systems. Various types of complete computational systems are however available. As a typical example, one might mention the Horn-Ikeuchi system, in which each 3-D model object can be described with the aid of a numerical transformation (the Extended Gaussian Image: EGI). Input images are projected on an EGI, and a purely numerical comparison of data with models is performed for the recognition purpose[5]. This method requires simple description levels and matches but it tends to be constrained to convex objects and sensitive to occlusions.

3.4 The Knowledge-based Approach

The knowledge-based approach has proved very powerful, not only for the interpretation process but also for algorithm management and parameter tuning. This approach is especially useful in accomplishing those tasks that have eluded attempts at automation which are based on 'image processing and pattern recognition methodologies, yet are routinely done by trained human technicians' [8] . The knowledge-based approach relies heavily on AI techniques to attain three main goals, as stated in [7]: representation of knowledge, the inference handling mechanism, and the control structure.

The main assumption of the knowledge-based approach is that any situation or object to be recognized can indeed be explicitly represented. The whole vision process, up to the final interpretation, involves an extensive use of knowledge: knowledge to describe spatial and relational properties of the objects to be recognized; rules to manipulate data and models; procedures to perform the whole interpretation process (e.g., important parts which are easier to recognize are searched for first); and strategies to recover misinterpretations. From a theoretical point of view, the use of AI techniques is not strictly required; however, utilizing explicit knowledge representation provides more facilities than embedding knowledge in the code of a conventional sequential algorithm. Explicit knowledge representation allows the criteria applied by the system to be easily verified or modified. Moreover, the application of flexible strategies based on heuristic rules ensures solutions even in those situations which are characterized by too many possible cases. Heuristic knowledge suggests the most plausible hypothesis, consistent with detected patterns and with the current recognition status and as a result the search for objects can be faster and more reliable.

The recognition process is distributed at several levels of abstraction which interact during processing. Information propagates from the lowest level of abstraction (say, pixels) to higher ones (e.g. edges, region, objects, etc.), under the control of knowledge sources; conversely, verification of hypotheses is based on interactions between higher levels and lower ones.

At the lowest level, knowledge sources are simple procedures that maintain the consistency of local hypotheses, through a projection of model information

from higher levels (back-projection). On the other hand, at the highest levels, knowledge sources provide qualitative evaluations and take into account constraints among objects. These feedback mechanisms for consistency maintenance are more applied at lower levels, while at higher levels, the process is more complex and global [6]. In summary, the most important characteristics of the knowledge-based approach are [7]:

- real data are readily available;
- there exist known methods for extracting features from data;
- the system is part of the environment;
- human expert knowledge and protocols for the interpretation process are available;
- a specific identifiable structure can be used for the interpretation process
- simple evaluation criteria can be adopted.

Several types of KB systems are available; for example, the Nagao-Matsuyama system[8] , which is based on a blackboard structure, and which, after a simple numerical stage (smoothing and segmentation), performs the recognition process using a control structure. Other systems worthy of mention include the ones developed by A. Hanson and E. Riseman, by H. Niemann, and by J. Tsotsos.

3.5 Limitations of Both Approaches

Both approaches exhibit some drawbacks. In particular for the KB approach, we recall Horn's statement: 'simple heuristic methods produce apparently startling results; later, significant limitations of these *ad hoc* approaches become apparent..., Machine vision needs to be supported by an understanding of image formation.' [4]. Weymouth and Amini observed that the 'type of scene should be as general as possible, however, (referenced) experiments are restricted to only few examples' [6]. Integration of lower-level knowledge sources (mainly numerical ones) is not an easy task; some *a priori* knowledge is often embedded in numerical algorithms, and an integrated control structure (e.g., adaptive algorithms) may give rise to some difficulties during integration into a fully knowledge-based structure.

Proponents of the computational approach too have their critics: 'Their efforts have led to mathematical models that, although powerful tools, ... are applicable in restricted cases... [which are] not general enough to encompass many realistic situations' [6]. Moreover, the computational approach does not allow easy representation of heuristic and semantic aspects, which are very useful in reducing the search for solutions to the recognition problem.

In summary, I would like to argue that the two approaches should be used according to the following considerations.

- When the recovery of 'sensor' information is complex and important *the computational approach* is very appropriate (e.g., the reconstruction of 3-D structure from 2-D images). This approach is characterised by:

  1. good mathematical tools;
  2. a significant amount of information at the lowest abstraction levels; and
  3. relatively poor semantic analysis.

- When complexity mainly affects the interpretation process, the *knowledge-based approach* is more suitable (e.g., in interpreting the structure of tomographic medical images). This approach provides specific information at the highest abstraction levels (i.e., the main data analysis is carried out at semantic levels, possibly taking into account *a priori* knowledge).

3.6 Integration of the two approaches

It would appear from the foregoing discussion that the advantages of the knowledge-based approaches could compensate for the weaknesses of the computational approaches, and *vice versa*. This could be achieved by integrating the two approaches. The question which remains, then, is: Is such an integration feasible and, if it is, how might it be accomplished? In my opinion, such integration is indeed feasible. To see how, it is first necessary to note the redundancy of techniques in the arsenal of the computational approaches; that is, more often than not, there exist several computational techniques for achieving the same or similar goals (e.g. range from stereo or visual motion; shape from shading or texture). Each of these techniques has its own particular strengths and domains of applicability. I would contend that knowledge-based approaches can be effectively used to select and tune the computational tools for the early vision stages. This can be achieved by a knowledge-based control structure using *a priori* knowledge. Thus, I expect increasing use of knowledge-based methods for algorithms management, i.e., in monitoring and controlling the computational approaches. In the same way, the knowledge-based approach can also be deployed in the development of distributed backtracking techniques (within and between levels of representation) in order to prevent the weakest module from limiting performances.

3.7 An Example – Control in the MuSIP System

To lend some substance to what has been said in the previous section, we now come to illustrate this argument with a specific example.

The MuSIP (Multisensor Image Processor) system is a knowledge-based image understanding system which was developed within the framework of ESPRIT project 2316. The MuSIP system comprises a Man-Machine Interface (MMI), a knowledge-base, an algorithm-base, a database, a blackboard, and a supervisor (i.e., a module devoted to control tasks); see figure 3.2.

The MMI is a module which assists the user when he has to alter or monitor the system activities. The knowledge-base contains information associated with the problem and the application in hand (e.g., knowledge about algorithms, object models for the purpose of image recognition). The algorithm-base contains a number of routines for low-level image processing, data fusion, and image recognition. For instance, there can be many digital filters for image processing, together with different data-fusion algorithms to merge the information provided by imaging sensors. Image data (raw data and their symbolic descriptions, including the results of interpretation) are stored in the database. Information on the status of processing and on the control status is contained in the blackboard.

3.7.1 Hierarchical organization of control knowledge

The MuSIP knowledge-based approach to controlling computational vision depends on a decomposition of the image recognition process into a set of simpler sub-processes. This decomposition constitutes a hierarchical organization of control knowledge (see figure 3.3). This organization effects a strategy of controlling actions in accordance with a goal and with the sub-division of the goal into sub-goals (or processes into sub-processes). In more detail, the control representation is hierarchically organized according to the decomposition of the recognition goal into a structured network of sub-goals (e.g., the low-level processing sub-goal and the interpretation sub-goal). Each node in this hierarchy is a knowledge source called a 'task' and is devoted to the control of its sub-goal. The definition of a particular network topology (i.e., the choice of control representation at the task level) is based on problem and application knowledge and is acquired by means of expert interviews [9].

These tasks constitute the coarsest level in control representation. Each task is devoted to the control of a sub-process (e.g. the filtering sub-process). It is composed of a set of production rules and static information. Each rule contains control modalities specified by computation procedures named 'keywords' (i.e. coded symbolic words associated with computation procedures). All information for every task is represented by 'frame-like' data-structures [10].

The whole control process is performed by an inference engine that aims to move control among all tasks and to fire the rules contained in the current active task. Specific rules are fired according to the processing status. The inference engine is activated at the beginning of the recognition process, and remains active until a termination condition is fulfilled. In particular, this engine has the control of a stack where activated tasks are pushed or popped, depending

on the order in which they have been called. Backtracking mechanisms are implemented among tasks.

3.7.2 Control in the low-level phase

To illustrate this general structure, we will look briefly at how this control strategy is put into operation in the context of filtering and segmentation. This is termed control in the low-level phase (see figure 3.3).

The two main control sub-tasks correspond to the two main sub-processes of the low-level phase phase (filtering and segmentation). These sub-tasks are then subdivided into other sub-tasks which are related to automatic selection and tuning of algorithms and to error handling [11]. Control moves sequentially from the filtering task to the segmentation one, through the related three sub-tasks. Backtracking strategies can be activated by the error-detection and recovery sub-tasks. These strategies can move control back to any previous sub-task (e.g., a poor quality of image segmentation can move control back to the filtering sub-task). The control knowledge contained in these sub-tasks will be explained in detail in the following sections.

First, let us consider the main characteristics of the control knowledge. It is based on both problem knowledge (i.e., knowledge about multisensor image processing and recognition) and application knowledge (e.g., knowledge about magnetic resonance – MR – images). The former kind of knowledge is provided by an image-processing expert and is represented by production rules in the aforesaid sub-tasks. This procedural knowledge decides what actions must be carried out, including backtracking actions. The latter kind of knowledge provides specific information on the image that has to be recognized, and is presented in frame-like data structures. This declarative knowledge is mainly used to choose appropriate sequences of image-processing algorithms, and their parameter values for a given application. In addition, application-dependent information for quality control and error recovery is also provided. For instance, a possible range for the number of meaningful regions to be expected in an image is provided for each application and it can be exploited to assess the quality of image segmentation. The procedural knowledge decides only the control actions to be carried out (e.g., it decides that quality control must be performed on segmentation results). The particular kind of control action and/or the particular parameter values for a control action are application-dependent; therefore, they are selected using the declarative knowledge.

3.7.3 Automatic selection and tuning of image processing algorithms

The automatic-selection and tuning sub-tasks contain the control knowledge for selecting the most appropriate algorithm sequences and the related parameter values to perform the filtering and segmentation subprocesses in an effective

way. Let us describe the organization of this knowledge. The procedural part of this knowledge is contained in the automatic selection and tuning sub-tasks and it decides what kind of process has to be executed (e.g., a 'noise reduction' process). The declarative part is contained in frame-like data-structures and it provides the sequences of algorithms to carry out the process selected by the procedural knowledge. Each frame is related to a sensor (e.g., an MR imaging sensor) and to one of the two low-level sub-processes considered (i.e., the filtering and segmentation sub-processes). The declarative knowledge is represented as a set of advised sequences of algorithms, with their parameter values. This kind of organization has been chosen because we believe it closely resembles the mental organization of an image-processing expert. In addition, it can be easily handled by the user (by adding and/or removing algorithms, or by changing parameter values). When a low-level sub-process (e.g., the filtering process) is to be performed, the first sequence is selected and run. Other sequences will be chosen according to the results of the Quality Control sub-tasks.

3.7.4 Quality control

Filtering and segmentation algorithms require a certain number of parameters whose values determine the quality of the results. The Quality Control tasks contain control knowledge to perform the appropriate quality tests on filtering segmentation results.

Concerning the filtering sub-process, after the selected sequence of algorithm has been executed, the control switches to the related Quality Control sub-task. It performs a quality test based on edge detection preservation and detection. Edge points are detected on both original and filtered images by standard algorithms (e.g., the Sobel filter), and the goodness of the image quality is assessed according to the following parameters:

- The percentage variation in edge intensity before and after the filtering process. Edge intensity is expected to increase after the filtering process.
- The percentage variation in the number of edges. This is expected to decrease after the filtering process (in the applications being addressed by MuSIP, a large amount of spurious edge points, generated by noise, exist before filtering).

Suitable values for the above parameters are given in the frames containing the declarative knowledge for quality control.

As regards the segmentation sub-process, the related Quality Control sub-task is mainly based on application-dependent information. For instance, for a given kind of image (i.e., for a given application), good results should provide a number of regions in a predefined range. Consequently, very important information for quality control is provided by the number of meaningful regions which are expected. In particular, the frames related to quality control contain

two main types of information for each application: the number of meaningful regions expected and the expected sizes of some image regions. The latter information concerns the sizes of the largest regions corresponding to the largest objects that are to be present in the image related to a given application. For instance, let us consider a medical application using MR images representing axial sections of the human head at eye level. In this case, the largest regions which we expect correspond to the image background and to the brain. Values can be fixed for the minimum and maximum sizes of the corresponding regions expected in the image.

3.7.5 Error detection and recovery

The error detection and recovery sub-tasks contain the control knowledge to detect filtering or segmentation errors and to carry out effective error-recovery actions. Error detection is based on the results of the Quality Control sub-tasks. Error recovery is based on control strategies implementing backtracking mechanisms. Different backtracking mechanisms are available:

- iterative applications of one low-level algorithm using either the same or different parameter values (i.e., a backtracking on the current algorithm);
- repetition of the same sequence of algorithms with a different set of parameter values (i.e., a backtracking on the current algorithm sequence);
- selection of another advised sequence of algorithms (i.e., a backtracking involving a change in the algorithm sequence).

All the information required to handle these backtracking mechanisms is stored, in symbolic form, in a data-structure called the 'frame tree'. Each node in the frame tree is associated with an algorithm of a sequence and contains some useful information to perform the backtracking mechanism (e.g., the parameter values used, the type of error detected, the kind and number of backtracking actions already performed, and the links with the previous and the following nodes). When an error occurs, the backtracking actions to be carried out depend on the reasons for the error and on the actions already performed to attempt a recovery. Different sequences of algorithms are related to different branches in the frame tree. When an error is detected, the controller consults the current frame-tree node in order to decide the backtracking actions. New branches of the frame tree are created if a backtracking action involving a change in the algorithm sequence is performed.

Concerning the filtering sub-process, the automatic selection and tuning sub-task suggests several sequences of algorithms with their parameter values. As an initial attempt, the first sequence is run. Depending on the quality-control results, a recovery action may be required. For example, when a filtering sequence causes edges to be detected, a backtracking mechanism involving a change in the algorithm sequence can be performed.

Concerning the segmentation sub-process, if the number of image regions is not in the prefixed range, an error is detected. 'Underflow' and 'overflow' segmentation errors may be detected, and the parameter values of the segmentation algorithms are changed accordingly. Analogously, error-recovery actions are performed if the sizes of some image regions are not what were expected.

3.7.6 Results

An example of the results which have been achieved using this approach in correction of under-segmentation errors is shown in figures 3.4 and 3.5. Specifically, figure 3.4 depicts an unprocessed computer tomography (CT) image representing an axial section of the human head at eye-level, while figure 3.5 demonstrates the result of error handling during the segmentation sub-process. A region-growing segmentation algorithm was advised for both images. After the algorithm was applied to the CT image, quality control was performed on the result of the segmentation. A 'region underflow' error was detected since the parameter values used for the segmentation provided an image with too few regions. A recovery was performed by changing the parameter values (i.e. the thresholds for the segmentation algorithm) in order to obtain a larger number of regions; then the algorithm was fired again, following which a satisfactory result was achieved.

3.8 Summary and Conclusion

In this chapter, the image understanding process has been defined as a set of progressive transformations of input data into the final description of a scene. Two basic approaches have been considered (i.e., the computational and the knowledge-based approaches), pointing out the related advantages and drawbacks. In my opinion, the computational approach is the most appropriate for performing lower-level transformations; in particular, to recover information distorted by the image acquisition process and to produce 'physical' descriptions (orientation, shape, etc.).

The knowledge-based approach is the most suitable for performing transformations and to represent data at higher levels (i.e., symbolic data representation and processing). This is important when contextual and semantic information is available, or when we want some kind of interpretation of the content of a scene.

We conclude that a general system for understanding complex images should be effective for all the above tasks and, hence, the integration of the two approaches seems to be the best solution. An integrated knowledge-based-computational system can also exploit the remarkable potential of A.I. control techniques to manage processing tools and to apply backtracking mechanisms

Figure 3.2: Components of the MuSIP system.

Figure 3.3: Hierarchical organization of control functions in the MuSIP system.

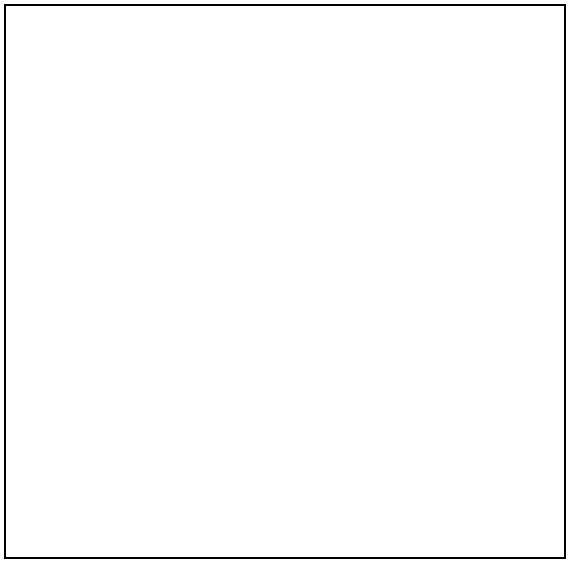


Figure 3.4: Correction of an undersegmentation error in a CT image: original image.

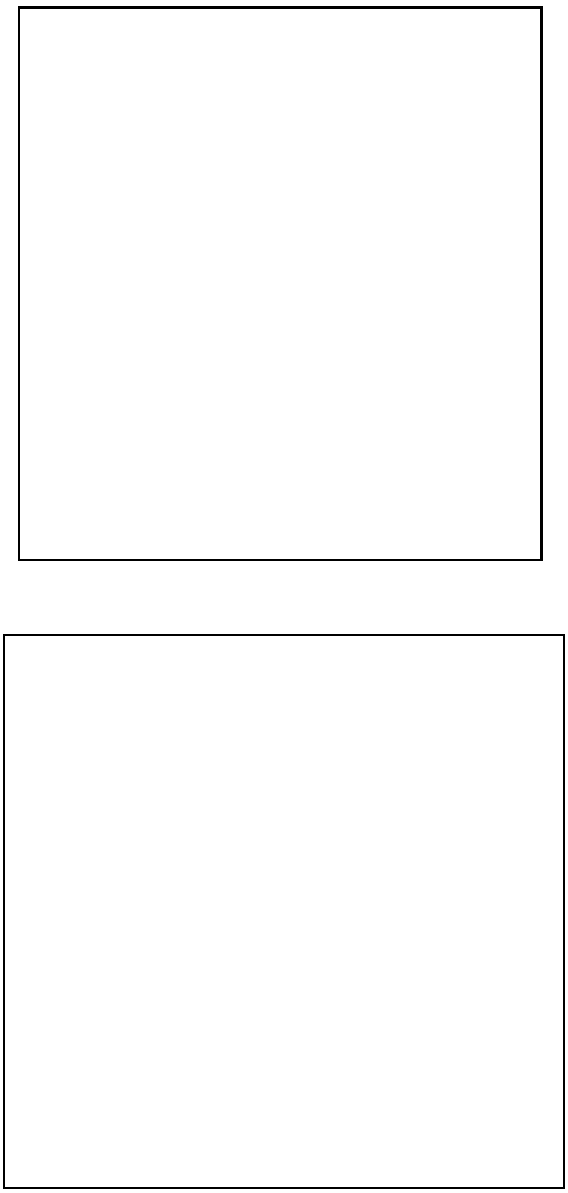


Figure 3.5: Correction of an undersegmentation error in a CT image: first segmentation (top); segmentation after correction (bottom).

for error recovery. An example of such a hybrid system has been given to convey some initial experience in applying this approach.

3.9 Acknowledgements

The author would like to thank Dr. S.B. Serpico for his helpful comments. The work was supported in part by the Prometheus program (CNR-Pro-Art) and by the Progetto Finalizzato CNR 'Biotecnologie'.

Bibliography

[1] W.K. Pratt, *Digital Image Processing*, J. Wiley, N.Y., 1978 (p. 568).

[2] S. Shapiro, Ed., *Encyclopedia of Artificial Intelligence*, J. Wiley, N.Y., 1989 (p. 6).

[3] R.J. Schalkoff, *Digital Image Processing and Computer Vision*, J. Wiley, N.Y., 1989 (p. 6).

[4] B.K.P. Horn, *Robot Vision*, The MIT Press, Cambridge, MA, 1986 (p. IX).

[5] K. Ikeuchi, 'Recognition of 3-D objects using the extended Gaussian Image', Proc. 7th Int. Joint. Conf. Artificial Intelligence, Vancouver, B.C., Canada, Aug. 24- 28, 1981, (p. 595-600).

[6] T.E. Weymonth, and A.A. Amir, 'Visual perception using a blackboard architecture' in *Image Analysis Applications*, R. Kasturi, M.M. Trivedi, M. Dekker, (Eds.) N.Y., 1990, pp. 235-280.

[7] N. Nandhakumer, and J.K. Aggarwal, 1985. 'The artificial intelligence approach to pattern recognition: a perspective and an overview', *Pattern Recognition*, **18(6)** (383-389).

[8] M. Nagao, and T. Matsuyama, 1980. *A structural analyst of complex aerial photographs*, Plenum Press, N.Y..

[9] S. Dellepiane, G. Venturi, and G. Vernazza, 1992. 'Model generation and model matching of real images by a fuzzy approach', *Pattern Recognition*, **25(2)** (115-137).

[10] A. Barr and E.A. Feigenbaun, 1981. *The Handbook of Artificial Intelligence; Vol. 1*, William Kaufmann, Inc., California.

[11] T. Figone, F. Fontana, A. De Salabert, S.B. Serpico, and G. Vernazza, 1991. 'Automation selection and tuning of multisensor image-processing tools by a knowledge-based control structure', *Digital Image Processing*, Elsevier Science Pub., Florence, (338-343).

Chapter 4

A Priori Knowledge in Vision

G. D. Sullivan

Intelligent Systems Group,
Department of Computer Science,
University of Reading,
United Kingdom.

4.1 Introduction

Computer vision provides a clear example of a conflict which is prevalent throughout advanced computing between research motivated by technical advances, and that motivated by practical requirements. On the one hand, the long-term objective of much 'academic' visual research is to develop generic functional components (sometimes called 'visual competences') which can be applied to a wide range of visual tasks. In low-level systems, where the visual task is typically to derive a general description of an image or a geometrical description of objects in the scene, this seems to be an attainable goal. Such tasks are generic and deal with a level of understanding wwhich is close to the image domain and which is largely independent of the specific semantics of objects in the scene. In high-level vision, where the meaning and purpose of object play and increasingly important role, the goal seems less realistic — the perceptual tasks are intrinsically more complex, and they inevitably involve application-specific concepts and methods. Instead of developing generic visual competences, research in high-level vision often leads to specific methods, which provide specific perceptual requirements which are difficult to generalise. This results in a disjoint

set of isolated 'perceptual competences'.

This state of affairs arises from two different motivations for research in vision systems, dominated respectively by academic and industrial research. These motivations are:

1. The desire to develop a deep understanding of the visual process, manifested by generic visual competences, such as methods for segmentating images, extracting features, detecting motion, deriving depth from stereoscopy, motion or symmetries. All these can be regarded as 'knowledge-free', in that they make no use of specific information about the semantics of the scene. Such work produces good incremental science, in the sense discussed by Marr, but developments have often proved to be of very limited immediate use to industry.

2. The need to produce hand-crafted solutions to specific practical problems. Such work has often led to one-off *ad hoc* methods. Systems are 'knowledge-based' in the sense that they make explicit use of the special semantics of the task. They have proved difficult to generalise. Research of this kind is often ingenious (for example, the visual inspection system which was described in chapter 1) but it may contribute little to the development of vision theory

In this chapter, and following on from the remarks on chapter 3, I will argue for a third approach to research, a middle way, which seeks to develop knowledge-based solutions to generic problems. The objective is to try to develop the perceptual competences needed for a wide application domain, and thus to make possible the solution of many practical tasks. An essential part of the endeavour is the development of application support tools, which allow the generic perceptual competences to be to specialised to suit specific visual tasks, as defined by an application engineer.

I illustrate the approach with examples drawn from the VIEWS project, funded under the CEC ESPRIT programme. VIEWS is a collaborative project, involving six partners from three European countries, and in such a project research objectives must be balanced with the complementary need to develop clearly identifiable commercially-oriented products in new areas of technical opportunity. These constraints force us to tread the middle way. I argue that this path is very productive, and can provide new insights of benefit both to the scientist interested in technical advances, and to the craftsman seeking practical solutions to current problems.

4.2 ESPRIT Project 2152: VIEWS — Visual Inspection and Evaluation of Wide Area Scenes

The VIEWS project is a collaborative research and development venture which is partially funded under the European Strategic Programme for Research and Development in Information Technology. The objectives are to demonstrate the use of vision systems in a wide range of traffic monitoring applications. The project has mainly concentrated on two types of scene: urban road intersections and airport ground traffic.

Our work takes a strongly knowledge-based approach to the solution of specific vision problems. By doing so we are able to avoid many of the problems of general vision and yet still provide very useful vision systems. Practical applications of traffic monitoring systems will normally have access to a great deal of contextual information about the perceptual task in hand, such as the lay-out of the scene, the position of the camera(s) with respect to the scene, the expected routes of vehicles in the scene, and the expected types of vehicles. Such *a priori* knowledge may be highly specific, as in the airport monitoring task, or it may be relatively uncertain, as in the census of road junctions. In either case, the proper use of expectations can greatly simplify the visual problem, and we show below that we can achieve very useful performance with quite modest computing facilities.

Knowledge such as this can be encapsulated in models: of roads, vehicles, expected dynamics, and specific events and behaviours. Models provide the means to interpret data, using higher levels of understanding provided by the knowledge used. The purposes of the perceptual system — in this instance, to monitor the positions and behaviours of vehicles with respect to known scenes — define goals which make possible a 'top-down' approach which greatly circumscribes the visual processing. This leads us to adopt the classical *cue-hypothesis-test* strategy: we first seek 'cues' in knowledge-free representations of the image, which suggest specific perceptual hypotheses, which in turn make knowledge-based predictions about the image which can be tested to confirm or refute the hypothesis.

The critical stage in this strategy is the ability to test the hypothesis; in our case it requires us to be able to evaluate a hypothesised vehicle of a particular type at a particular pose with respect to the camera.

4.2.1 Pose estimation

In the VIEWS system, the geometrical properties of the vehicles, and the layout of the roadways, are represented as wire-frame models expressed in individual object frames. Using simple geometrical reasoning, vehicles can be placed on the ground at arbitrary locations in the world model. We also have a carefully constructed a camera model, which allows us to project an instance of the 3-D

scene onto the 2-D image plane. Given an hypothesis of a vehicle in the scene, we can therefore use techniques of computer graphics to derive the 2-D image features we would expect to see. Each feature comprises a line segment (the projection of a single wire of the model), which is tagged by a description of its likely image properties, according to its role in the instantiated model (fold-, crease-, or extremal-edge; bar; mark on a surface; shadow edge; etc).

The agreement between the prediction and the image data can be tested by a process which we have called 'iconic evaluation' [1, 2]. Each linear feature is evaluated by applying simple criteria, based on derivatives of smoothed intensity values in the direction perpendicular to the feature. Thus edge features are scored according to the average strength of the first derivative, and bars according to that of the second derivative. The evidence from all the visible lines of the object is pooled, by expressing each score as a probability, taking account of the lengths of the lines and the type of feature. Probability tables are established empirically, by placing lines of various lengths randomly in a calibration image (ideally, this is the image under investigation, but in practice we use a previously computed temporal median image), and determining the score. Any given feature score is thereby associated with a probability that a score at least as high would have been obtained by chance from a randomly placed feature of the same type and length.

The individual model features are treated as independent samples from the empirical probability distributions, and are pooled using a χ^2 test. The result is a single scalar in χ^2 units, having expectation 0, and typically ranging from -3 (in areas of the image with far less than average detail) to 5 or more when the projected model fits the image very well. Scores over 2 are treated as significant indications of a vehicle.

The resulting evaluation score for a vehicle is reasonably independent of the position and pose of the object, since it measures (to a first approximation) how likely the score was to have been obtained by chance, taking into account the number of visible features, their types and their lengths in the image.

4.2.2 Pose refinement

In unconstrained viewing of a known object, the evaluation score defines a scalar function of six dimensions; in world coordinates these are most simply defined as the three Cartesian coordinates of the object's position and the three angles of orientation. In general, we expect that peaks in the six degree of freedom function will indicate likely matches between the model and the image. The problem is to locate the peaks, and thereby to determine the pose of the vehicle.

A major computational simplification can be made by limiting the object's position to the ground plane, thus permitting only two dimensions of translation and one of rotation about the vertical axis. Using these simple but (normally) realistic physical assumptions, only three independent dimensions remain. Figure 4.1 illustrates part of the surface of the evaluator function. Only two degrees

of freedom can conveniently be shown. Pose may be parameterised as pan angle (horizontal position in the image), depth on the ground plane towards and away from the camera, and rotation of the object about its coordinate centre. Pan corresponds approximately to simple translation in the image, so that the estimation reduces effectively to correlation between a fixed template and the image. The other two parameters introduce non-linear distortions, due to projection distance and pose angle, respectively. These are the two axes shown in figure 4.1. It can be seen that even for this 'difficult' cross-section, the surface is fairly smooth and well-behaved. The (correct) central peak extends for approximately $\pm$1m in depth and $\pm 15^\circ$ of rotation; any initial guess within this range (i.e. which estimates the pose of the vehicle accurately to within these bounds) should be sufficient to discover the correct pose.

On a serial machine, an exhaustive search of the evaluator surface over three dimensions is computationally too expensive. We have therefore explored a number of iterative methods for locating maxima in the surface.

Separated Ascent

This technique successively decomposes the problem into three separate one-dimensional searches, each based on the current estimate of the object's coordinate frame. This is illustrated for the x-coordinate (the left-to-right axis of the car) in figure 4.2. For each sample the model is displaced by an appropriate amount (in the object coordinate frame), instantiated into the image, and its fit to the image is evaluated. The results obtained for a typical search along a single dimension are shown in figure 4.1(b) where the abscissa represents the displacement and the ordinate represents the evaluation score obtained (the higher the score the better the fit). Note the subsidiary peak, due to accidental alignment of one side of the car model with the wrong side in the image. The best score and its position are noted and the process is repeated for the other two variables (here, y and rotation about z), each time starting from the same initial pose.

When the search has been completed in all three dimensions, the pose having the highest score found is adopted. It then becomes the initial position for the next iteration, and the search coordinate frame is changed accordingly. If no higher score is found then the three ranges of the search are reduced to be equal to the previous sampling interval. The process is repeated until all the sampling intervals fall below pre-determined values (see below).

Simplex methods

Cheaper methods for maximisation have also been investigated and their relative merits explored. The Simplex algorithm [9] starts with a seed position, and samples the evaluation function at nearby points in the configuration space defined by the domain (here position and rotation with respect to the seed

pose). In a three dimensional search, the points form a tetrahedron about the seed in configuration space. The point at which the function is lowest is replaced by a better point along the line joining it to the centre of the opposite face; this point is found by binary search. The process iterates until the values at the four points become closely similar.

Gradient Ascent

The most direct method for maximisation is simple gradient ascent. A discrete estimate of the local partial derivatives of the surface for each parameter is computed, and the seed is moved in the direction of their vector sum. The process is iterated until all partials are near zero.

Initial and termination conditions

The initial search range, and the terminating conditions are strongly object- and pose-dependent. To compute them automatically, the object model is approximated as a sphere of diameter equal to the greatest diameter of the model. Simple geometry uses the 'seed' pose (in the world-coordinate frame) to compute the displacements in x and y and the rotations about z (in the object coordinate frame) which would cause a 1 pixel change in the image for a worst-case object feature. These determine a set of scaling parameters, which are weighted to give the termination conditions required by the application (we typically use 0.5 pixel change). The initial search range is set by similar reasoning, and we typically use bounds corresponding approximately to ± 0.5 times the dimensions of the vehicle and $\pm 10°$. At extreme distances, or in poses in which one of the object coordinate axes is directed towards the camera, the initial ranges may already imply a sampling interval below the terminating conditions, in which case only one iteration (for that axis) is performed.

Performance

Figure 4.3(a) shows a model that was very approximately instantiated near the vehicle by hand. This is fairly typical of an initial estimate of the position obtained from simple image-based cuing processes. The search algorithm was allowed to run, and the result is shown in Figure 4.3(b). Informally, by eye, it seems that the fit is very good.

To estimate the overall signal-to-noise ratio of the method, we have taken the best fitting position and orientation for the model in this image (as shown in figure 4.3(b)), and then evaluated this particular model instance in a sequence of 500 images taken at 25 *Hz*. The results are shown in figure 4.4. We see one conspicuous peak, with several subsidiary peaks. The images corresponding to the points on the function labelled a to f are shown in figure 4.5(a)-(f). The response of the evaluator is typically about -3.5 in the absence of vehicles.

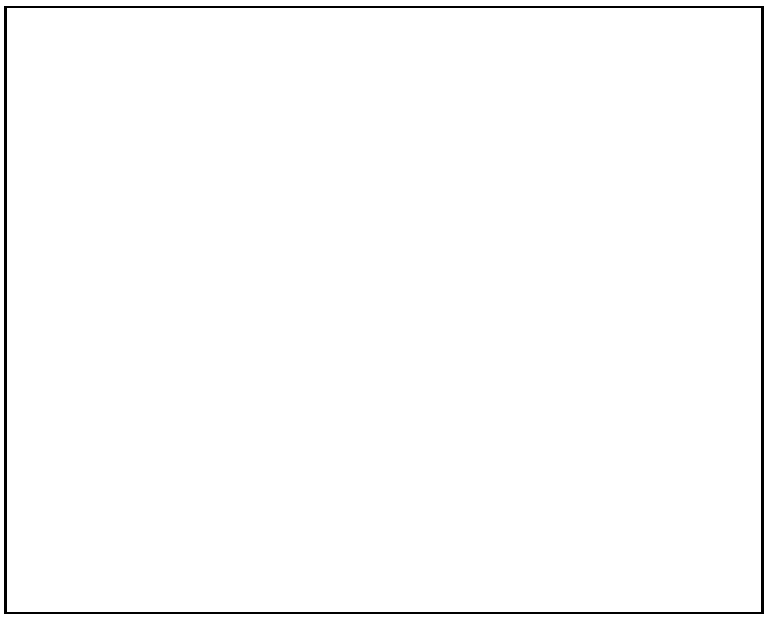


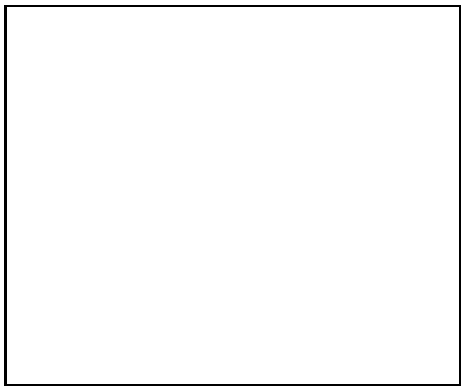
Figure 4.1: Cross-section of the evaluator function, showing variations in score (E) with depth from the camera (d), and rotation about the vertical (θ), of an object constrained to the ground plane (see text).

The minor peaks correspond to accidental alignments between the model and a 'wrong' vehicle, but these scores never exceed 0, and the peaks are fairly flat and ragged. The score for the 'correct' vehicle reaches 3.7, and is well-localised. In this simple case, where there is little distracting background image detail, the evaluator provides an intuitively acceptable measure, and has good signal-to-noise ratio.

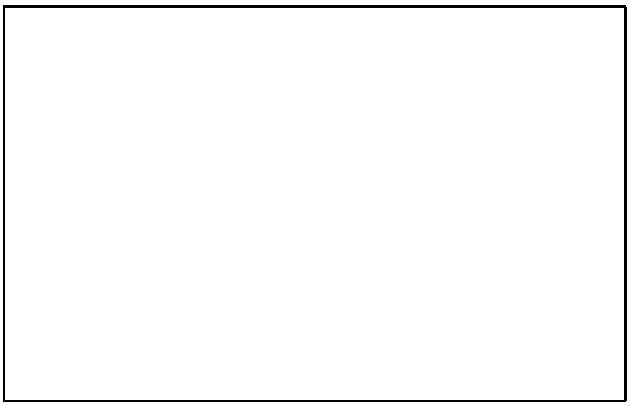
4.3 Assessment of search methods

A continuing problem in object recognition research is the need to compare different algorithms, both to compare methods which differ fundamentally, and also to assess the impact of the many design options within a given approach. This section describes methods developed within the VIEWS project to quantify the performance of the overall pose-recovery sub-system.

The performance of an iterative maximisation algorithm is mainly determined by (i) the smoothness of the function, (ii) the prominence of the maximum amongst (false) local maxima, (iii) the search routine used, and (iv) the

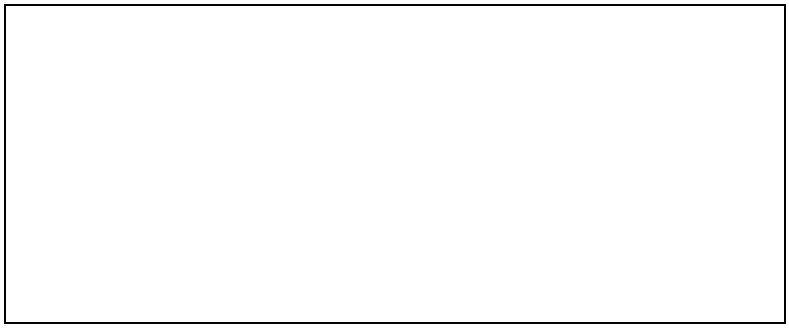


(a)



(b)

Figure 4.2: (a) For each degree of freedom the model is displaced, instantiated, and its 'goodness-of-fit' evaluated. (b) Evaluation scores obtained from a model by displacement along a single dimension



(a) (b)

Figure 4.3: (a) Before and (b) after gradient ascent

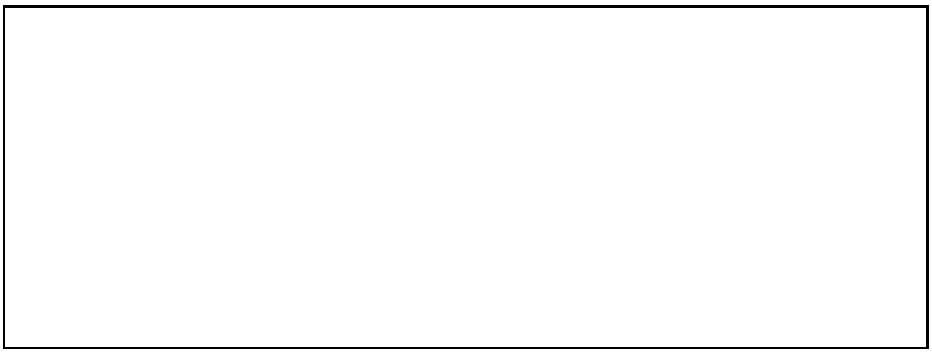


Figure 4.4: Evaluation function for a fixed position on different images (a)-(f); refer also to the next figure.

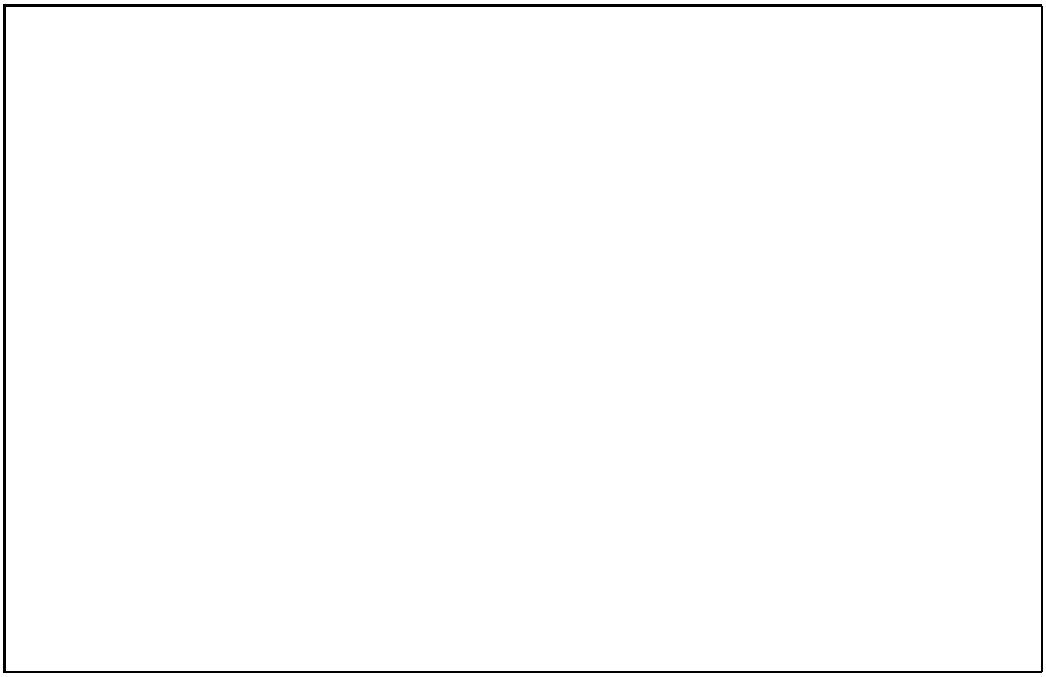


Figure 4.5: Fixed model in different frames (a)-(f); refer also to the previous figure

initial (seed) pose. The first three factors are, in turn, affected by many free parameters of the algorithms, which must be established in the course of the implementation. To do this efficiently and effectively we need to define a summary measure of performance.

One such measure can be obtained by studying the range of seed positions which converge to the correct answer. In principle, a search algorithm defines an equivalence relationship on the domain of seed positions, in which all seeds giving rise to the same result form an equivalence class. In practice, we must tolerate a small error in the result, to take into account inaccuracy in the resulting poses, due to fine-grain noise in the evaluation function and (more importantly) the terminating conditions adopted by the algorithm. One such equivalence class can be identified (by eye) as correct. The size, convexity and compactness of the correct class provides important measures of the quality of the search algorithm.

Figure 4.6 shows the results of trials in which the search space (x, y, θ) was sampled to provide seed positions within a range close to the correct result. The search algorithm was run and the recovered pose was identified. The resulting points in (x, y, θ) are shown in three orthogonal projections in figure 4.6, as scatter diagrams. We see a strong cluster of points around the origin (the correct pose), with significant clusters at false maxima.

These data can be used to estimate the size of the correct equivalence class, by computing the probability of success as a function of the initial displacement of the seed. A simple measure of performance is then given by the distance at which the probability falls to a criterion value (say 50%). This approach depends on a distance metric (r) which normalises the three axes. At present we scale the axes so that the scatter of points in diagrams such as figure 4.6 seem fairly isotropic, but a more formal statistical approach to normalising the axes could be adopted.

The data for figure 4.6 were derived from the image shown in Figure 4.7(a). Figure 4.7(b) shows the data plotted as probabilities of success as a function of r, for three different search algorithms: (i) the separated ascent; (ii) the simplex algorithm; and (iii) steepest ascent. Figure 4.7(c) shows the average computational cost as a function of r; as usual with such algorithms, this is dominated by the number of computations of the evaluation function.

The results shows that the steepest ascent algorithm performs very badly, and only rarely recovers from seed positions beyond 0.5m or 5°. The Simplex algorithm achieves far better performance, mostly at a lower cost. The separated ascent algorithm has somewhat better convergence properties, but at considerably greater cost. For this image and object model, we obtained the criterion 50% success rate for seed errors equivalent to 0.3m or 3°, 0.7m or 7°, and 1.1m or 11°, for the three search algorithms respectively.

Results such as these are strongly dependent on the object and image under consideration. The vehicle in figure 4.7(a) is well isolated from distracting image detail. Figure 4.8(a) shows results obtained when the same model is fitted to a

second object in a part of the image where there is considerably more irrelevant detail. The convergence properties of all three algorithms is significantly poorer, but the same preference order of performance occurs.

This method of comparing search algorithms has also been used to assess other minor changes to the pose-recovery method. Systematic analyses are currently being made of alternative methods for feature evaluation and the method used for updating the visibility calculations (Section 4.2.1), as well as the significance of individual features of the model, the feature aggregation method, and the sensitivity of the system to small geometrical errors between the model and the object. It also provides us with a formal measurement of the performance of the system as a function of object types, viewing distance, and occlusion. These data are fundamental to any principled design of a surveillance system for practical use.

4.4 Dynamic Tracking

The pose recovered from the iconic search is used as an input measurement to a Kalman filter, which allows us to impose simple dynamic constraints appropriate to vehicles. Instantaneously, a car has only two degrees of freedom - its forward (or backward) velocity, v, and its angle about the z-axis (θ). We have implemented a Kalman filter [8] which allows freedom in v, $\frac{\partial v}{\partial t}$, θ and $\frac{\partial \theta}{\partial t}$. The three parameters v, $\frac{\partial v}{\partial t}$ and θ are constrained to be within plausible bounds and, since the filter is expressed in the object-centred coordinate frame, these constraints may be estimated on the basis of common knowledge of car dynamics. Note that this characterisation of the vehicle's dynamics prohibits it from sliding sideways, but does not (at present) couple v and θ. The measurement error assumed in the Kalman filter is also pose-dependent, and is estimated on the same basis as the terminating conditions for the pose refinement process. We typically assume that the error of a recovered pose is Gaussian distributed with a standard error equivalent to a 1 pixel worst-case displacement (*c.f.* section 4.2.2, above).

Tracking proceeds as follows. The initial 'cued' pose hypothesis is first refined as described above. Starting with plausible default values, the Kalman filtered estimates of v and θ, together with the newly measured position, provide a prediction for the object's pose in the next frame. This becomes the seed pose for the next search, and the process continues. When the filter parameters become stable, the forward prediction improves in accuracy, and fewer iterations of the separated gradient ascent algorithm are required; in these conditions we have found that a simpler (3-D) steepest ascent algorithm is sufficient. It also becomes possible to predict several frames in advance and thus reduce the computational burden of the process (though there is a trade-off with the requisite size of the initial search space).

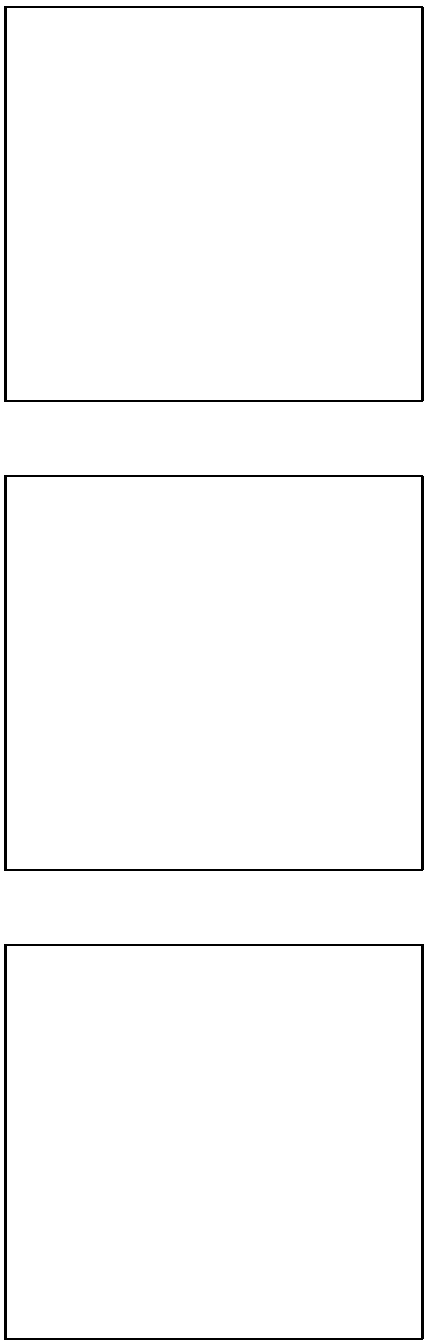


Figure 4.6: Orthographic projections of sampled results $-30°$ – $+30°$; $-30°$ – $+30°$; -3.0 x (metres) +3.0; -3.0 y (metres) +3.0; -3.0 y (metres) +3.0; -3.0 x (metres) +3.0;

Figure 4.7: (a) Segment of uncluttered image; (b) probability of correct convergence; (c) average number of evaluations. In both graphs 1.0 on the abscissa represents 2.0 metres or 20°.

Figure 4.8: (a) Segment of cluttered image; (b) probability of correct convergence; (c) average number of evaluations. In both graphs 1.0 on the abscissa represents 2.0 metres or 20°.

4.4.1 Results of model-based tracking

The Kalman-filtered iconic search has been used to track vehicles in a number of scenes. Typical results are illustrated in figure 4.7 for the vehicle illustrated in figures 4.2–4.5, for every 10th frame (i.e. every 400 msec). Figure 4.7 shows the results of the pose-refinement algorithm, i.e. the measurement input to the Kalman filter, not the smoothed result of the filter, which in the present car-centred implementation, is difficult to reconstruct in the world coordinate system. In the main the performance seems very good, and the position and orientation of the vehicle was recovered accurately. Some instability was observed, especially in frames 220–240, where the angle of the vehicle was miscalculated, and 310–350, where the model begins to spin on the spot. Note that at various stages the car was occluded, either by objects in the scene (e.g. the lamp post, frames 270–310) or by other vehicles (frames 320–350). Both types of occlusion are taken into account by 3-D reasoning: the lamp post forms part of the scene model, and, though not shown in figure 4.7, the position of the occluding car was known since all vehicles in the scene were tracked simultaneously.

Initial seeding

So far, no account has been given of how the model-based tracking is initially seeded. In the system being developed, the primary seed hypotheses are provided by the analysis of movement in the image. Two methods have been developed, which offer complementary properties in practical applications. The first (due to work carried out at the Fraunhofer Institute, Karlsruhe, Germany) is based on the detection of extremal points in a band-passed image, which move consistently between successive frames as coherent clusters. The second (due to work by Marconi Radar and Control Systems) maintains a running 'temporal median image', and detects regions of change with respect to this. The former method is fast and has already been developed to run on special hardware at video rate (25 Hz). Its main draw-back is that the extrema are clustered together purely on the basis of image velocity vectors, so the process tends to fragment objects which rotate before the camera, and merge adjacent vehicles having similar movement. It also loses vehicles that become stationary. Region-based methods have better ability to segment overlapping vehicles, and to overcome brief periods of immobility, but is more sensitive to sudden overall changes in illumination. In addition, it cannot at present be made to run at video rates.

In either case, it is possible to obtain fairly reliable tracks in the image, assumed to be due to single objects moving in the scene. The centroid of the moving image data, projected back onto the known ground plane, provides a very approximate indication of the position of the object. By noting how this evolves over time, the likely pose of the object can be estimated. In turn, an analysis of the distribution of extrema, or of the moving region, gives an

indication of the broad class of object (e.g. large, small, long) taking into account the distance from the camera and the approximate pose, given by the detection of movement. In highly constrained traffic scenes, this information has proved sufficient to initiate the model-based methods. Further classification, into the precise type of vehicle, can be carried out by examining the relative scores of the iconic evaluator (after pose-refinement) for different models.

Once started, the model-based method is usually capable of running without further input from the movement detection systems. It has good immunity to rotation and partial occlusion, and does not fail when the vehicles are stationary.

4.5 Performance

The traffic understanding system has been successfully applied to a number of scenes at complex road intersections, and at airports. It results in a recovery of the full 3-D position and pose of vehicles in the known scene, which can be passed on to the other major component of the VIEWS system, the Situation Assessment Component. This maintains a long-term history of the vehicles in view, and detects events and behaviours of significance to the end-user of the system. These include unary events (such as a vehicle entering a zone of special significance), binary relations (such as one vehicle causing another to give way, or one vehicle overtaking another), as well as higher order interactions (such as queue formation and dissipation). An application area receiving special attention is that of monitoring traffic at stand-areas of airports, where the airport authority needs to verify that aircraft are visited in the correct order by ancillary vehicles such as baggage handlers, water tankers, fuel tanker, cleaners, service engineers etc. Figure 4.9 illustrates a typical frame from one of our test sequences; the recognized vehicles are superimposed as wire-frame models on the image (a), and shown from overhead view in the scene representation (b).

4.6 Discussion

The primary merit of the model-based method for object perception is that it works and seems reasonably robust. The outcome of the system is a full 3-D interpretation of vehicles, moving in the known 3-D scene. We have applied the method without significant changes to a wide variety of image sequences, taken under different environmental conditions, involving more than a dozen different vehicle types, partially occluded vehicles, different viewing distances and camera angles, in bright sunlight (giving heavy shadows), at night, in rain and in fog. The method deals naturally with the type of variation to be expected in practical traffic monitoring applications, and is currently being developed within the ESPRIT VIEWS project as the perceptual input to systems which monitor vehicle movements in real-time for safety and scheduling purposes.

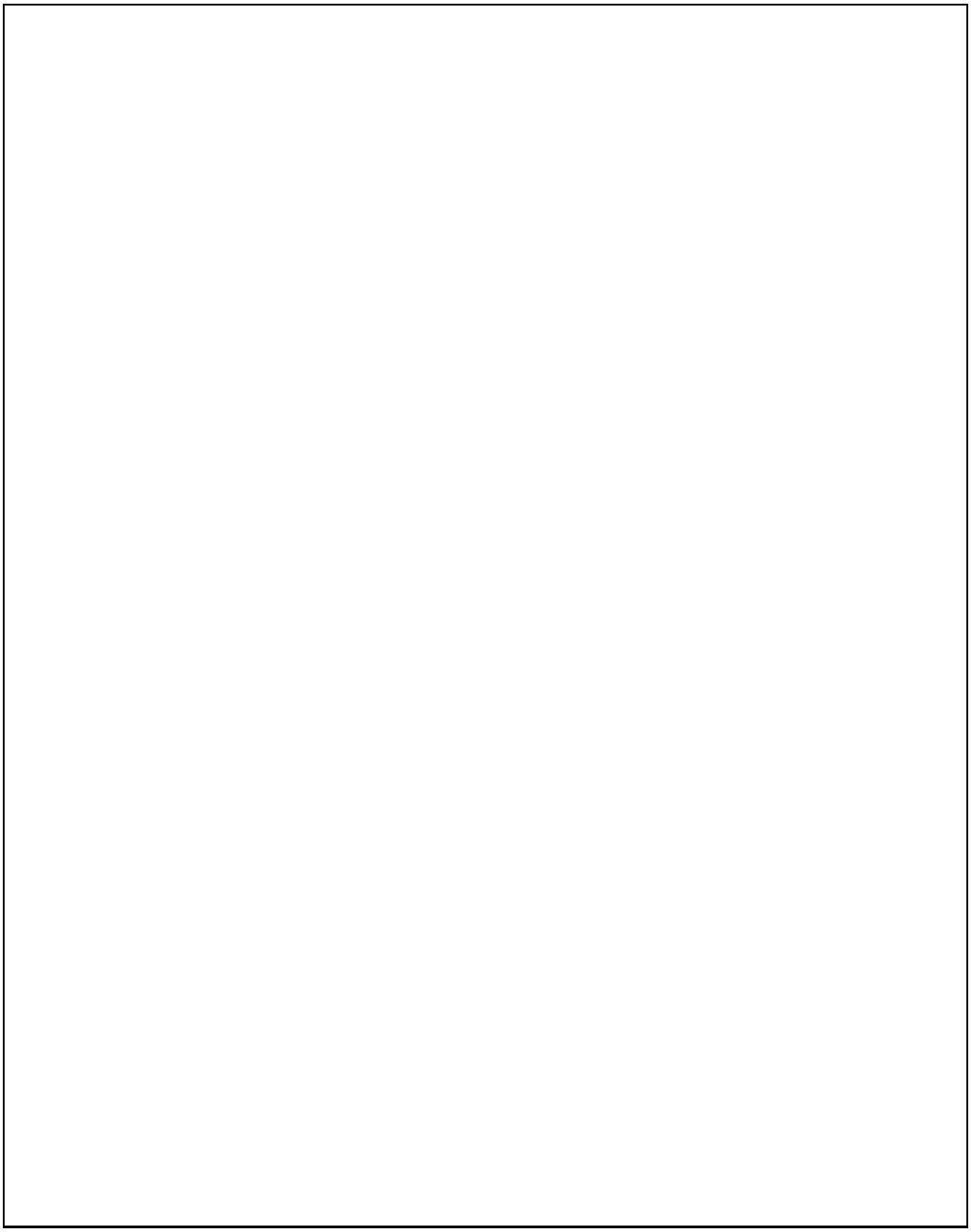


Figure 4.9: Tracking a single car (every 10th frame from frame 80 to 350)

The key to success is clearly our ability to make use of detailed knowledge of the expected objects and their behaviour. The perceptual problem then becomes sufficiently constrained to be tractable. The 'top-down' use of *a priori* knowledge permits an hypothesis-driven approach to image analysis, and finesses many of the traditional problems in object recognition. In particular in the model-based tracking part of VIEWS, all feature analysis and all grouping is carried out under the control of explicit perceptual hypotheses.

4.7 Application Support Tools

The methods for 3-D tracking developed in VIEWS depend completely on specific knowledge of the perceptual tasks involved. Without the *a priori* knowledge of the geometry of the camera, the scene, the expected objects, and of the expected behaviour of vehicles, the top-down approach to vision would be impossible. We therefore need to address the question of how such knowledge is acquired. A major part of the VIEWS project has therefore been devoted to creating software tools to facilitate the capture of application-specific knowledge.

4.7.1 Camera calibration

Methods for using perspective inversion for recovering the intrinsic (focal-lengths) and extrinsic (position and orientation with respect to a world coordinate frame) of cameras are poorly conditioned, and surprisingly difficult in practice. In principle, we need only to identify a set of points in the image with the known world coordinates of the corresponding points in the scene, to be able to recover the camera parameters. VIEWS has been applied to more than 6 different scenes (three airport scenes, and three public road traffic scenes). In each case proper camera calibration is required for the ground plane constraint to be exploited.

The main difficulty has been the fact that accurate world data has been hard to obtain. Our initial attempts were based on large scale maps provided by the airport (or road) authority. These proved totally inadequate for the task: since the maps were initially compiled, walls, pavements, road signs and lamp-posts had often been moved, airport taxi-ways had been extended and widened, even buildings had sometimes been added! What is clearly needed is a full geometrical survey of the site, as it exists. However the project had no authority (or funds) to disrupt the continued use of the scenes, and we have had to have recourse to a combination of special reasoning and guesswork to recover the camera parameters with any degree of accuracy. This has proved time-consuming and unprincipled, and we know that our estimates are poor.

In more controllable scenes (on private roads, where we have been free to use surveying techniques) the problems become much less difficult. Here, we have simply used the camera to capture images in which a surveyor's pole was placed

vertically on the (assumed flat) ground, at a set of positions having known distances apart from each other. The operator indicates the top and bottom of the pole in each image, and a perspective inversion algorithm, based on that of Worrall *et al.* [14], recovers the camera parameters. This has proved effective and very efficient: the entire process takes two operators about 30 minutes per site.

4.7.2 Object modelling

Assembling an object model by hand is tedious and error-prone. As with the scene measurements, detailed data is rarely available and even then, it is often inaccurate. Initial models were hand-crafted to fit the images we have, using information from engineering drawings, measurements taken directly, and a lot of simplifying assumptions. This approach proved adequate for the purposes of research, but would clearly be impossible in commercial practice. We have therefore adapted methods for recovering the shape of vehicles from images automatically, to work under interactive control of the application engineer.

With a single camera,'shape from movement' techniques are obviously attractive. Algorithms for recovering shape from unknown (six degree-of-freedom) motion reported in the literature, such as Wenn *et al.* [12], proved hopelessly unstable for our purposes. However in our situation, the vehicles are constrained to the ground plane and this fact makes it possible to use far more accurate and robust techniques for obtaining structure from (three degrees of freedom) motion, see Tan *et al.*[11].

The recovered shape is still not in the form of a model for use in the vision system: individual points must be grouped into lines (to define the features), and associated with surfaces (to determine their visibility and their type, in different poses). One further observation about vehicles which has turned out to have considerable power for model-building is the fact that they are usually symmetrical about the vertical plane containing their central axis. Using this symmetry, pairs of corresponding points in a single image (say, the bottom-left and bottom-right corners of the windscreen) define a line in space which is known to be parallel to the ground plane, but at an unknown height. If we set the height of one such pair (by using specific knowledge), then we can compute the position (in 3-space) of the symmetry plane. All other such pairs of symmetrical points can now be located, by requiring their height to be such that the points are equally distant from the plane. Furthermore, points which are invisible from one particular view can be inferred from knowledge of the coplanar properties of their visible symmetrical mates, also defined interactively by the application engineer.

Software tools to facilitate such model building have been constructed and are in regular use. The tools output source code (in pop-11) which can be interpreted directly by the model-based vision system.

4.7.3 Behaviour definition

The third source of *a priori* knowledge used in VIEWS concerns the expected behaviour of the vehicles in the scene. This is of two types: (i) the physical dynamics of the vehicles, as imposed by the Kalman filter, and (ii) the routes and interactions of vehicles within the scene, according to the purposes of the drivers. The former represents constraints on vehicles and is encoded as part of the models. In VIEWS applications the latter represents the output of the perceptual system, and is strongly dependent on the application domain. In the airport scenes, the main objective is to monitor the compliance of the vehicles with airport rules and instructions, and to maintain an understanding of the how the servicing task is proceeding. To do this we need to recognise 'events' (such as the fact that a baggage-handler has taken up the correct position for unloading the aircraft), as well as extended 'behaviour' (sequences such as the baggage-handler spending some time at the fore cargo hold, then at the aft hold, and then exiting the scene).

To accomplish the monitoring role intended for VIEWS, the Conceptual Component of VIEWS must have the ability to recognise such behaviours, and to reason with incomplete and uncertain data. To date, events have been defined as the presence of a vehicle in a special region, as recorded on a semantic map. The map may be fixed with respect to the scene (to define entrances, exits, roadways, etc.), or with respect to an object (to define positions of hatches, service points etc.). Software tools have also been developed for use by the application engineer to define such information interactively.

4.8 Knowledge in Vision

The design philosophy of the VIEWS project is seen to have two main components: firstly, to develop special-purpose perceptual competences that allow the critical data to be extracted from image sequences automatically, and second to develop ways in which the general-purpose visual abilities of an applications engineer can be used to define specialised high-level knowledge about the particular scenes. The high-level knowledge serves two purposes: (i) it constrains the visual task, and makes it feasible, and (ii) it defines the semantics of the scene as dictated by the application. This approach to developing vision systems has proved highly productive.

At the present state of development in computational vision, completely knowledge-free methods are too unreliable in the types of scenes we wish to interpret. Furthermore, low-level vision *per se*, however successful, is incapable of producing a level of understanding and interpretation demanded by end-users of vision systems. This seems inevitable: as is argued by Sloman in the next chapter, vision is intimately concerned with purpose, and without high-level purpose vision does not advance beyond image processing. For the foreseeable

future, progress towards practical vision systems is likely to depend on the ability to use human intelligence to define and delimit the automatic processing. Our experience in the VIEWS project has shown that the combination of perceptual competences, and user-defined constraints and goals, defined using application support tools, provides a powerful way to proceed.

4.9 Vision: craft, science, or engineering?

A central issue debated in the workshop was the status of current vision systems - are they the result of craft, science, or engineering? At one level VIEWS is an example of vision developments which are very much at the craft stage: the algorithms have been created and collated with the prime purpose of demonstrating a practical system, often with only a very poor understanding of the properties of the underlying methods. However, in keeping with many areas of human creativity, the craft stage is an important precursor to deeper understanding of a technology.

It is impossible to study the properties of a system as complex as VIEWS without having a working prototype system. For example, there have been many *ad hoc* decisions in the choice of system parameters. Particularly clear examples are the choice of maximisation routine, and the methods used to sample the image in the feature evaluator. In the experiments shown here we have integrated the image along only five normals, spaced at 1 pixel intervals. Current work is aimed at exploring the options more fully. To do this we need to define a suitable performance criterion, and this is impossible in isolation from the rest of the system.

However, having developed a prototype complete system, it becomes possible to carry out performance tests, and tune the parameters to optimise the overall performance. Systematic studies of this kind take the exercise beyond that of a craft towards a quantitative understanding of the methods, more akin to conventional engineering practices. In turn, the insights derived from the quantitative studies feed back to the design of the algorithms, and thus lead to better scientific understanding of the underlying methods. Here then, is the reward of the 'middle way' forced on us by the demands of pre-competitive, yet commercially relevant research. The achievements of the research (craft?) aimed at practical commercial requirements can lead to advances in both science and engineering.

Bibliography

[1] K. S. Brisdon, G. D. Sullivan, and K. Baker K D. 1988. 'Feature Aggregation in Iconic Model Matching', *Proceedings of the Alvey Vision Conference*, Manchester, 19–24 (1988).

[2] K. S. Brisdon, *Hypothesis Verification using Iconic Matching*, Doctoral Thesis, University of Reading (1990)

[3] R. A. Brooks, 'Model-based three-dimensional interpretations of two-dimensional images', *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **5**, 140–150 (1983).

[4] W. E. L. Grimson and D. P. Huttenlocher, 'On the verification of hypothesised matches in model-based recognition', *Proceedings of the First European Conference on Computer Vision*, Antibes, France (1990)

[5] D. C. Hogg, 'Model-based vision: A program to see a walking person', *Image and Vision Computing*, **1**, 1–20 (1983).

[6] D. G. Lowe, *Perceptual organisation and visual recognition*, Kluwer Academic Publishers, Boston (1985)

[7] D. G. Lowe, 'Fitting Parameterized 3-D Models to Images', *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **13(5)**, 441–450 (1991).

[8] R. F. Marslin, G. D. Sullivan, and K. Baker, 'Kalman filters in constrained model-based tracking', *Proc. British Machine Vision Conference*, Springer-Verlag (1991).

[9] W. H. Press *et al. Numerical Recipes*, Cambridge University Press, (1986).

[10] E. M. Riseman and A. R. Hanson, 'A methodology for the development of general knowledge-based vision systems', in *Vision, Brain, and Cooperative Computation*, A. Arbib and A. R. Hanson (eds.), MIT Press, Cambridge, MA (1987).

[11] T. N. Tan, G. D. Sullivan, and K. Baker, 'Structure from Constrained Motion, using Point Correspondences', *Proceedings of the 2nd European Conference on Computer Vision*, G. Sandini (Ed.), Lecture Notes on Computer Science, **588**, Springer-Verlag (1992).

[12] J. Y. Weng, T. S. Huang, and N. Ahuja,'3-D Motion Estimation — Understanding and Prediction from Noisy Image Sequences', *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **9**, 370–89 (1987).

[13] A. D. Worrall, R. F. Marslin, G. D. Sullivan, and K. D. Baker, 'Model-based tracking', *Proceedings of the British Machine Vision Conference*, Springer-Verlag (1991).

[14] A. D. Worrall, K. D. Baker, and G. D. Sullivan, 'Model-based perspective inversion', *Image and Vision Computing*, **7(1)**, 17–23 (1989).

Chapter 5

How to Design a Visual System — Gibson Remembered.

A. Sloman

School of Computer Science,
The University of Birmingham,
Birmingham,
United Kingdom.

5.1 Introduction and Key Ideas.

In this book so far, we have looked at several aspects of the configuration and design of computer vision systems. For the most part, we have considered the particular relationship between the system design and the functional specification of the task which the system must execute. In this chapter, we will broaden the discussion somewhat to consider more global systemic issues which arise when we are dealing with very complex agents which exhibit a diversity of behaviour rather than the well-defined (but not necessarily easily-achievable) response of, say, a PCB inspection system. We will argue that, in the design of a visual system, it is necessary to consider the role that vision plays in the context of the total system architecture of which the visual system is a part. In an organism, for example, a visual sub-system interacts with many other sub-systems which serve different purposes and the input to the visual system may include information from several of these sub-systems. In the same vein,

the output from the visual system, which may include descriptive and control information, may be sent to many other sub-systems. The interaction and inter-relationships among the sub-systems which constitute the entity of which they are a part needs to be considerably more complex than has been the practice heretofore. For agents which are as complex as humans, vision has to do far more than simply inform the agent (or perceiver) about the shape and motion of the objects in its local vicinity. The 'affordances' in the environment include many other kinds of states, processes, causal powers, and, significantly, it may include many possibilities for interpretation. The requirements for the design of the visual system thus depend in part on the environment; but the environment is defined in part by the needs of the perceiver. Consequently, we will argue that there is no unique 'correct' design; rather, there is a space of possible designs and we should explore the options and the trade-offs [1].

5.2 Modular *vs.* Labyrinthine Systems.

To begin this discussion on the design of a vision system, it is perhaps appropriate to look first at the approach which has had a pervasive influence on the thinking of the vision community, i.e., the approach of David Marr. According to Marr, the essence of vision is the use of retinal data to produce information about shape and motion, and possibly colour. From this point of view, a vision system is a well-defined module with well-defined inputs and outputs and, if any information is needed about the environment other than shape or motion, then it must be inferred *from the output of the visual module* or from other modules. The archetypal 'Marrian' vision system (see figure 5.1) comprises, as its inputs, a set of images of the visual scene. Descriptions of visible surfaces are derived from these iconic representations, descriptions which in turn feed visual processes which derive object or scene-centred descriptions of shape and motion. Cognitive processes, such as planning, learning, inferring, control, monitoring, and so forth, then use these representations of shape and motion. This approach to vision could be termed the 'one-way front-end' model of vision.

Implicit in this Marrian approach is the assumption that the global architecture of the system which we are designing and building — the cognitive system — is itself modular (*c.f.* [2]). The other modules constituting the system include, for example, touch, taste, hearing, and smell, each of which, along-side vision, feeds a 'cognition' module (refer to figure 5.2). The cognition module interfaces then with both memory and control modules.

This modular approach extends to and includes the composition of the vision sub-system itself, so that the 'one-way front-end' model comprises intermediate processes and data-bases (repositories or representations) connected in a modular and sequential way. Each of these sub-modules fulfills a particular information processing task, such as computation of optical flow, texture, stereo disparity, and shape from, e.g., stereo, motion, and texture. Each of these pro-

Figure 5.1: The 'one-way front-end' model of vision.

Figure 5.2: The implied global architecture.

cesses uses and generates databases of information, such as edges, lines, regions, binocular disparity, colour, specularities, and so on. But despite the *internal* richness of this model, it allows only input via the retina (i.e. the images) and it allows only a limited class of outputs to the central cognitive processes. The question which then arises is whether or not such a restricted output would be useful for an intelligent organism or an intelligent robot. If other inputs and outputs were allowed, would not a vision system be able to contribute more effectively to the functioning of the system of which it is a part? I would argue that the answer to this question is: 'Yes, there is a far richer role for vision'.

5.3 Towards a Richer Theory of Vision

Vision has many practical uses and serves many cognitive purposes. These uses include planning, feedback control of actions and posture (including so-called 'reflex' control signals), monitoring conditions for triggering new reflexes, and the detection of danger signs and opportunities. The descriptions — or outputs, if you will — which are provided by vision are many. For example, strictly two-dimensional descriptions are adequate for many tasks: painting, sighting a gun, or steering a boat in a straight line; while the results of the intermediate stages in the visual process, e.g. optical flow, histograms (which indicate global information such as 'there is a lot of green in the scene') or texture gradients, are also useful. On the other hand, an example of a non-geometric output of a visual system is a description of the causal power of objects and situations in the environment, including life-relevant properties such as edibility, rigidity, the ability to support or the usability for various purposes [3]. The list of examples of non-geometric output could continue. For instance, the presentation of the behaviour of other agents (walking, eating, threatening, protecting); or the presentation of the mental state of other agents (what they are looking at, what mood they are in, what are their intentions, whether they are happy or sad).

To appreciate the distinction between the geometrical and the non-geometrical output of the visual system, consider the two line drawings in figure 5.3. The first drawing is the familiar Neckar cube while the second can be interpreted either as a duck facing left or as as rabbit facing right. Both of these are examples of *visual* ambiguity. However, one of them is geometrical, while the other is not. Specifically, when our interpretation of the second drawing flips back and forth between 'duck' and 'rabbit', what changes is the functional role associated with the parts of the perceived entity (e.g. a bill or a pair of ears), and which way the animal is facing. These are not differences in size, shape, or motion of visible surfaces, but more abstract differences.

In a similar way, it can be argued that there are other non-geometrical outputs of vision. For example, the four pictorial examples in figure 5.4 suggest that *causation is seen.*

Figure 5.3: Geometrical and non-geometrical visual flips.

Vision too offers the possibility for seeing the potential for change and it is regularly used in a type of visual 'hypothesise and test' exercise to see the potential for change. This spatial reasoning is very much an active process where the visual appearance is 'imagined', manipulated, or moved around until the required configuration is achieved. For example, the drawing in figure 5.5 shows a triangle and a circle. The objective is to ascertain whether or not the circle and the triangle can be re-arranged so that they intersect in exactly five points. Trying to do this by spatial reasoning (rather than by mathematical reasoning) leads one inevitably to imagining, or visualizing, the circle in many different configurations, one of which may satisfy the requirement.

The Marrian framework also suggests a (predominantly) bottom-up flow of information in the visual process. However, as the 'droodles' shown in figure 5.6 clear illustrate, there is a very important expectation-driven top-down aspect to vision. Each of these four pictures cannot be interpreted until a word or phase suggests an interpretation, after which they suddenly *look* different. The power of top-down processing is clearly shown in the fourth picture of a mexican riding a bicycle by the way in which an unusual view-point is created.

5.4 Towards Richer Visual Architectures.

The arguments in the previous section point clearly to a more 'interconnected' architecture than the one shown in figure 5.2. Instead, a more 'labyrinthine' architecture is suggested where intermediate results and other outputs from other sensory modules (e.g. hearing, touch, ...) together with prior knowledge about the environment are connected and act as inputs for each other. Figure 5.7 shows a schematic representation of this type of architecture. However, even this architecture is itself inadequate in some respects and a more complete theory would need to take into account the motivation producing or produced by visual processes.

Figure 5.4: Causation is seen.

Figure 5.5: Seeing potential for change.

This type of interconnected architecture offers interesting possibilities for learning mechanisms. Specifically, if rich enough kinds of learning are possible, both the input and the output links, as well as the set of descriptive concepts, can change. The resultant architecture, then, would not be static. Clearly, some of this learning — new ways of classifying things, for example — could make use of connectionist mechanisms, although the setting up of new links for control flow and data flow would require different operations. Indeed, the visual system itself may need different kinds of sub-mechanisms for different sub-tasks and its integration depends on the use of data-structures which are 'indexed' via the optic array[4].

5.5 Unsolved Problems.

There is, however, one major unsolved problem in all of this: How to represent spatio-temporal structure in such as way as to integrate the information required for recognition, control, planning, monitoring, explaining, solving problems, etc.. This leads directly to the issue of *what* is represented: if you represent only what is there, you leave our the rich collection of *possibilities* inherent in empty space and in spatial structure. These possibilities include movement and the alteration of spatial relations (many different kinds of motion are possible only in certain contexts: e.g. approaching, entering, screwing into, etc.); deformation of existing objects (stretching, bending, twisting); or the introduction of new objects.

How relevant is what we have been discussing? Much of the above, while plausible and, indeed, arguably necessary, is not yet at the point where it can be easily exploited on specific applications. Arguably, future systems should evolve in this manner but, for current practical purposes, *ad hoc* solutions may

Figure 5.6: The power of top-down processing in vision.

Figure 5.7: A partially-complete 'labyrinthine' visual architecture.

suffice.

Bibliography

[1] A. Sloman, 'On designing a visual system: towards a Gibsonian computational model of vision', *Journal of Experimental and Theoretical A.I.*, **1(4)**, 289–337 (1989).

[2] J. Fodor, *The modularity of Mind*, MIT Press, Cambridge, Mass. (1983).

[3] J. J. Gibson, *The ecological approach to visual perception*, Lawrence Earlbaum Associates, (1979).

[4] H. G. Barrow and J. M. Tenenbaum, 'Recovering intrinsic scene characteristics from images', in *Computer Vision Systems*, A. R. Hanson and E. M. Riseman (Eds.), Academic Press, New York (1978).

Chapter 6

Computer Vision — Craft, Engineering, and Science

K. Dawson, D. Furlong, M. Jones, N. Murphy, and D. Vernon

6.1 Introduction.

We come now to the final chapter. The emphasis throughout the book so far has been on the issues which influence the design of vision systems and the considerations which affect the evolution of the paradigms upon which vision systems are based. This chapter will continue in this vein, drawing out the issues which appear to play a pivotal role in the development of computer vision sytems. We will begin with the most general, and indeed the most fundamental, issue: the paradigms for vision which underpin the way we conceive of a vision system and the role we see it playing: as a truly autonomous system or as a utilitarian control system. Leading on from there, we will proceed to discuss the possible taxonomies of vision sytems, whether it is more, or less, useful to classify them on a functional basis (what they do) or on the basis of their constituent algorithms (how they do it). The use of knowledge, explicit or implicit, *a priori* or learned, raises its head here too, for the chosen taxonomy dictates not only the methodology for the design of vision systems but also the manner in which knowledge is incorporated into the system.

Both of these discussions lead us naturally to contemplate the state of play in the field of computer vision and its maturity as a discipline. It is in this context that the title of this book arose — *Computer Vision; Craft, Engineering, and Science* — for it reflects the mutually-relevant but distinct evolutionary stages of development of all disciplines.

Finally, we conclude the chapter , and the book, by asking two questions which are simple to pose, but difficult to answer: What is the role of a vision system and what is the best way to build one? Definitive responses would surely ring hollow but, in the light of the material presented here, we will venture some tentative answers. Their verity is something that only experience will decide.

6.2 The Paradigms for Vision

One of the purposes of this book is to assess and examine the accepted paradigms which are inherent in the field of vision system design. To initiate proceedings it is therefore appropriate to query how vision systems are constructed. Are they input/output systems in the normal engineering black box or transfer function schema? Are they autonomous systems, i.e. systems embodying closure in the form of a number of sophisticated interacting feedback loops? Or are they fundamentally process systems, either representational or connectionist? We need to examine the schema which we inherently adopt in approaching vision system design.

As we saw in chapter 2, an exciting area of development has been the advent of *active vision* systems which support interaction and necessarily require a view of the vision system as an on-going process involving a nexus of competing sub-systems rather than as an interpretative system, i.e. as a process of getting as much information as possible from a static image or a series of static images. Such an approach demands that we formally deal with attention, which issue does not even arise when dealing with interpretative vision systems. Indeed, the more traditional approaches to vision system design have focussed almost entirely on computational problems in the sense of seeking to provide interpretation of static scenes. When we impose real-time constraints, the *real* problem of the purpose of a vision system emerges. It could be argued that Marr, for example, completely avoided such questions because his focus was on computational efficacy. However, when freed of the shackles of the computational Formula 1 Grand Prix, vision systems can be seen in a different light. Vision as a process is not just the effective recovery of the shape of a giraffe from a given sequence of scenes, for example, but becomes a problem of *dynamic purpose tracking*, where the purpose of the system has become maleable. This kind of purposive dynamism raises questions of representation, i.e. if our vision system is to dynamically interact with a given environment then how does it maintain its representation of that scene? Does it involve some kind of representational updating? And if so what is the form of the representation? Is it a geometric or symbolic representation? Symbolic storage seems to be required in that there is a richness and freedom allowed by dealing with a more abstract level than geometric representation. However, this in turn raises the issue as to whether or not symbolic storage is hallucination or representation — a road map or the road. If it is the former then it might be true to say that vision as

Figure 6.1: A spectrum of vision paradigms.

process is a question of controlled hallucination rather than updated representation, and that the ability of the vision system to expect or predict would be based on internal modelling or simulation rather than scene interpretation and representational modification. As such, a symbolic representation must involve *controlled hallucination.* Getting robust invariant image descriptions is far from easy and requires much work for we cannot as yet say whether they can be general purpose in nature, or whether they must be application-specific grouping algorithms. Only through laboratory experiment is this question likely to be answered, i.e., the proof will be in the doing.

This view of the vision system as a controlled hallucinatory environment is a move away from a representational paradigm toward a constructionist stance where, rather than adopting a paradigm of visual input/output, the system is viewed as dealing with sensor surface perturbations from which the symbolic representation is manufactured. But, in such constructionist systems how does it know what to do? From where does its purpose derive from? Typically, the answer involves a rule-based supervisor of some sort who selects perceptual tasks based on a pre-defined menu of options. Only through moving more towards autonomous systems does the possibility of evicting the homunculus agent become available. Thus, a continuous spectrum of vision system paradigms begins to emerge: from representational systems, through constructionist systems, to autonomous systems, where many hybrid shades are feasible (see figure 6.1).

At the representational system end of the spectrum, there is the inherent requirement of isomorphic mapping of some external world; at the autonomous end, this isomorpism is specifically denied and consequently the issue of isomorphic mapping does not even arise. In the middle ground, constructionist systems can be viewed as *almost* autonomous agents which are given purpose by some pre-determined rule-based system which will use sensor pertubations as a ground for its decisions but which will require isomorphic mapping. Are all these vision system paradigm options equally valid? Just what constitutes a *real* vision system? Is an array of photo sensors a vision system? Or would tacking on an expert system transform such an array into a vision system? It would appear that the answer depends very much on the context and that therefore *the choice of vision system paradigm is application dependent.* A vision system

can mean different users. It may well be that there is no one all-embracing paradigm which characterises a vision system.

For many applications (e.g. inspection systems) relatively simple vision systems are pragmatic, but if the quest is *vision understanding* then we are of necessity required to consider issues of autonomy and intelligence. In this respect, the move toward vision as process is opening up new test-beds in that the functionality of the vision system is being relegated to a subservient layer with the focus being put on the development of intelligent homuncular agents. However, although this approach can be seen as a move along the spectrum of vision systems toward autonomy, these systems are not truly autonomous in that there is still reliance on pre-programmed, rule-based task allocation. If a vision system cannot define its own goals, can we call it truly intelligent? Does such *homuncular autonomy* advance matters much? Developments in the field are of the nature of application specific solutions and are certainly not proving to be a panacea for general-purpose vision systems. To expand the relevance of our relevance of our research efforts requires that we expand our contextual perspective with the attendent that we specifically address issues of true autonomy. In this light, a very fundamental observation seems to be that there is not, as yet, any well developed *science* of autonomy. Much work remains in the definition and development of such a science which, from a research point of view, is good news indeed.

It might well be asked if effort expended in the development of solutions for specific problems in intelligent vision system design is energy well spent, for when we view such efforts from a more removed perspective it seems that we are attempting to erect magnificently ornate structures on very shaky foundations. If the same efforts were to be invested in the investigation of the principles of a science of autonomy then the long term rewards might be of far greater significance. There is an apparent reluctance on the part of vision system designers to invest in autonomy and, with regard to vision systems, the related topics of perception and intelligence. This is not frequently made explicit as advances in the form of pragmatic results (computational efficiency increases, or superior edge detection algorithms, for example) are far more gratifying and more publishable than any acknowledgement of profound ignorance when faced with problems of autonomy, perception and intelligence. In effect, it would appear that the vision system research community is phase-locked to a constricted domain by the mutual lack of admission of ignorance on the part of those involved.

Herein lies the crux of the matter, for if we cannot apprehend (and, worse, cannot even admit that we cannot apprehend) what natural vision is, then it is most unlikely that we can develop artificial vision systems. In effect, all we are doing is groping in the dark and gauging our efforts by the effectiveness of the results achieved. This might indeed be a pragmatic approach but is hardly intelligent, for the number of interacting variables involved and the range of strategies possible are so large as to mitigate against the achievement of general purpose results within several lifetimes, if at all. However, if we were to get

the foundation right, and in the context of vision systems this means grasping what vision is, then the efforts to develop general purpose vision understanding systems are likely to be more efficient and to yield results of greater value and of more universal application than those which have been achieved to date. To extend the analogy, it might be said that many of the pre-fabricated functional elements — the cladding — that might go into the constitution of general purpose vision understanding systems are in existence, but that what is lacking is an approach to stable foundation and framework design onto which these functional elements might be attached. There is nothing to suggest that these fundamental issues are beyond the scope of the ingenuity of those working in vision system design. It is simply a matter of lack of attention to date. Therefore, the responsibility now lies firmly with vision system designers to address these foundational issues - what is it to see, what is it to be intelligent, what is it to be autonomous.

6.3 Taxonomies of vision systems

There are two chief conclusions arising from the arguments in the previous section: first, that the long-term goal of robust general-purpose computer vision will necessarily involve a well-founded understanding of autonomy; and second, that much can be achieved, in the short term, by adopting pragmatic, application-specific, approaches to vision which exploit what is known and what has been established about visual sensing.

Let us now set aside the elusive, if important, goal of understanding autonomy and concentrate on the pragmetic issues; the issues which will facilitate short-term benefit, if not long-term truth. We begin by considering the manner in which we delimit, or classify, pragmatic vision systems. We do this in the expectation that, if we choose the appropriate classification criteria, and hence the attendant taxonomy, we might have greater success in designing vision systems which match well the chosen application.

The table of contents of most computer vision textbooks provides the reader with a list of algorithms and by deploying these algorithms, an expert is supposed to be able to construct a vision system. This sounds plausible. However, something is clearly missing: this is the knowledge and experience which is required to select and combine the algorithms in a manner which is most appropriate for a given task. Alternatively, one might address vision systems on the basis of their functionality; for example, by describing how one might go about addressing tasks such as gauging, pose estimation, free-space mapping, or terrain modelling. As a functional taxonomy is a higher level of abstraction than an algorithmic one, it seems that a mapping from the functional to the algorithmic taxonomy is required. The difficulty is that there can be considerable choice in the selection of the algorithms which best suit the task and it is here that vision becomes somewhat more *ad hoc* since there is no formal and well-grounded

method for the selection of the algorithms. On the other hand, algorithmic problems are typically well specified and are mapped directly into computational solutions. Functional problems in vision, though, are often poorly specified (usually just through the provision of 'typical' example instances) and at present little has been done to formally relate functional problems to particular algorithmic solutions. Nonetheless, a functional taxomony seems to be the more useful of the two, although it necessitates the existence of the mapping between functionality on the one hand and algorithms and representations on the other.

Almost all vision systems use knowledge to a lesser or greater extent. There appear to be, at least, two forms of knowledge: there is domain knowledge and there is the knowledge base, which is accessible to the vision system, and which is abstracted, or filtered, from that domain knowledge. This knowledge base can comprise procedural knowledge or declarative knowledge, with implicit or explicit representations of information. The decision as to which knowledge is 'important' from the entirety of the domain knowledge is what determines the processing model which is adopted at the algorithmic level of the vision system. It seems that, quite often, when a vision system is being designed, you don't initially decide on a particular processing model, i.e. a particular type of algorithm, but instead you look at what knowledge is important, and based on that you decide on the processing model. It is the knowledge which is important rather than an initial decision about which algorithm to choose. Of course, these two are to an extent mutually-dependent, but the issue is the order in which the choice is made. In essence, there is first the transformation from the application domain to the knowledge base and this, then, implicitly defines what type of techniques can be used. For example, the industrial inspection system which was described in the first chapter uses correlation-based rules to make the necessary quality assurance decisions, but the signatures which represent the 'useful' information about the application are what are primary and were developed first — the correlation paradigm followed. This observation about the ordering of choices in the design of a computer vision system — first knowledge, then algorithms — seems to suggest that indeed a functional taxonomy would be the most appropriate. However, the critical mapping from the functional and the knowledge to the algorithms still remains to be made explicit and formalized.

6.4 Vision in a Broader Context.

Consideration of the possible means of classifying vision systems and the way in which they are designed leads to a broader issue; that of the state of play of vision in general. There are three different manners in which vision (or in fact most problems) can be viewed; as a *craft*, as an *engineering* discipline, or as *science*.

The craft approach encourages the solution of particular problems by any

means at our disposal through the judicious use of previous experience and intuitive understanding of the problem, even though those means do not necessarily have any formal basis. Craft vision takes a problem and solves it by hammering what is known about vision and the particular application into (the required) shape. This is exactly how the visual inspection system described in chapter one was designed: by looking to see what reliable information could be easily extracted from the images of the scene, and by identifying the information which best characterized the the information. Having decided on the information, we then come to decide on the most effective way to extract, process, and analyse the information (in this instance, by thresholding, signature formation, and correlation).

Science may be regarded as a means of formally addressing problems in strict, rigorous, well-founded, and mathematical way. But it may also be viewed as a means of formalising the methods which are found, by experience, to work. For example, consider both the theory of invariance in differential geometry for extracting curves in images and the simple *ad hoc* techniques for performing the same operation.

On the other hand, engineering is the application of science to particular problems using some reasonably well understood and systematic methodology. Engineering constitutes the middle ground, the methodological and well-grounded application of theories in the context of specific application constraints.

The relationship between craft, engineering, and science is not, however, a simple linear spectrum with craft at one end, engineering in the middle, and science at the other end. For, while science and craft may appear to be poles apart, there is nonetheless a subtle relationship. Craft supplies the intuition (and experience) which is essential to the formulation of scientific principles and science rationalizes, *a posteriori*, the intuitive and heuristic by elucidating the (hidden) theory on which it rests. The relationship then is a circular one in which each of the three terms feed off one another, contributing, in total, to the advancement of the area (see figure 6.2).

It would appear that the majority of application-oriented vision systems have been developed by the craft of the vision engineer. Our goal is, or should be, the development of the scientific principles and engineering methodology underlying vision so that vision problems may all be addressed in a systematic and formal fashion (i.e. moving from craft to engineering under the guidance of science).

6.5 Conclusions

It is fitting that we should conclude a book such as this, which has attempted to look under the surface of the popularly-accepted mantle of computer vision, with two bold, if not brash, questions: What is the role of a vision system and

Figure 6.2: The evolution of the discipline of computer vision.

what is the right way to build one?

As we have noted above, we must stipulate the type of system which is under consideration before answering such questions. For application-specific vision systems, the role must be to meet the demands required for that specific application. For more general ones, e.g. autonomous vision systems, the role is seen to be one of survival of the perceiver. Between these two extremes, many other sub-roles can be identified, e.g., to obtain useful information from the environment or to make decisions about the state of that environment. Given the evolutionary model of Craft, Engineering, and Science of computer vision as a discipline, there was an acceptance that our current stage of evolution corresponds mostly to the Craft stage, with notable developments at both the Engineering and Science stages. Consequently, it was agreed that, while it would be ideal to be able to build general vision systems, for now the vision community should adopt, in part at least, a bottom-up approach to building application- and domain-specific vision systems in an effort to maximise the benefit of Craft approaches while developing the engineering and the scientific: The vision community cannot and should not wait until there is a perfectly complete and self-consistent theory of vision before they try to progress. The formation of taxonomies of current techniques and their extension was considered to be one way of making the transition from the craft to the engineering phase.

This is sound advice indeed. But it would be wrong to conclude on such a pragmatic and, from the research point of view, such a limiting note, for in no sense was the tenor of the discussion at workshop one which reflected a lack of ambition. On the contrary, the strong feeling was one of challenge: challenge in promoting the integrity of the vision community and challenge in advancing the state-of-the-art of computer vision. But it is a well-understood sense of challenge: knowing what is to be addressed — the autonomous, the constructionist, the representational — and why and how it is to be addressed — for the ad-

vancement of the science of vision, for the promotion of principled engineering of vision sytems, and for the exploitation of the craft-oriented experience we have in applying our current understanding of computer vision.