A DISTRIBUTED SYSTEM FOR ANALYZING TIME VARYING MULTI-RESOLUTION IMAGERY

C. L. Tan
and
W. N. Martin

Computer Science Report No. TR-86-01
January 1986

A DISTRIBUTED SYSTEM FOR ANALYZING TIME VARYING MULTI-RESOLUTION IMAGERY

C. L. TAN

Department of Information Systems and Computer Science
National University of Singapore
Kent Ridge
Republic of Singapore

W. N. MARTIN

Department of Computer Science
Thornton Hall
University of Virginia
Charlottesville, VA 22903

This report has been submitted for publication in a special issue on Computer Vision of the journal, Computer Vision, Graphics and Image Processing.

This research was supported in part by the National Science Foundation through grant ECS-83-07248. Mr. Tan is supported by the National University of Singapore through an overseas graduate scholarship.

ABSTRACT

A distributed system for analyzing time varying, multi-resolution imagery is described in this paper. A pipelined pyramid structure is constructed in the system by continually converging incoming images into successive levels of decreasing resolutions. A set of processes work concurrently and asynchronously on subimages at different levels of the pyramid. These processes initially watch for interesting features in the coarsest resolution rendition of the scene. Processes working on promising areas individually but cooperatively proceed to progressively finer resolution levels. A blackboard structure also exists in the system that permits coordination among these processes, resulting in a unified motion interpretation of the imagery.

TABLE OF CONTENTS

1. INTRODUCTION

High computational cost has long been a problem in computer vision because of the enormous amount of data associated with each image. Multi-resolution image structures, such as pyramids, cones and quad trees [1], have in recent years been introduced to alleviate this problem. Such structures permit hierarchical search and heuristic planning [1] whereby rapid focus-of-attention on promising data may be achieved. Furthermore, these structures because of there hierarchical interconnection are amenable to parallel/serial implementation [2]. In concurrence with these developments, another new area of computer vision emerged in the last decade. Instead of working with static scenes, researchers began experimenting with the capability to perceive time varying imagery. To achieve this capability, time ordered sequences of images have to be processed with the objective of collecting information from the sequence as a whole. Computational cost is thus even higher than the traditional static image analysis, and is especially acute when real-time performance is required.

We have combined the above two areas of developments by allowing time varying imagery to be represented in a dynamic multi-resolution structure. Furthermore, the analysis of these images is distributed among several processors which periodically communicate among themselves through a coordinator.

The workings of our system are motivated by the following scenario. A man walks out of a building and hears the noise of a jet plane. With preconceived features of an airplane, he looks skyward trying to locate a moving object bearing those features. As he scans his field of view, many other objects will also be in sight. These items need to be sorted through in order to locate the airplane. It has been suggested that two phases of motion perception exist in human vision, namely *peripheral* and *attentive* phases [3]. In the initial scan, a set of peripheral processes working in parallel quickly eliminate uninteresting objects but direct

promising areas to attentive processes for more detailed analysis. Successive stages of attentive processing gradually discard further items until the airplane is identified and can be tracked. Meanwhile, the original peripheral processes continue to watch for any new interesting object, such as a second airplane, that may enter the scene.

In our system, the sequence of incoming images are successively converged into varying levels of degrees of resolution. The system is given a set of pre-determined objects of interest that are expected to appear in the scene. Peripheral and attentive processes are allowed to run concurrently with the former processing the lowest resolution rendition of the scene while the latter are being directed to work at progressively higher resolution levels until their individual objects of interest are located.

2. TIME VARYING MULTI-RESOLUTION IMAGERY

The multi-resolution representation that we use in our system is a pyramid structure. Such a structure may be visualized as a sequence of two dimensional images representing a visual scene in less and less detail from the base to the top level of the pyramid [1, 4]. Usually the dimension of the images is reduced by half at each step in the sequence. For instance,

$$P=(S^{512\times512},S^{256\times256},S^{128\times128},\ldots\ldots,S^{1\times1})$$

is a pyramid of 10 image levels. Each element of an image, S, represents a pixel that contains some scene feature information, such as intensity.

Alternatively, a pyramid may be considered as a set P of cells together with a function F that assigns a value v to each cell. Thus

$$P=(\,(v,k,i,j)\mid 0\leqslant k\leqslant L;\ 0\leqslant i,j\leqslant 2^k-1;\ v=F(k,i,j)\,)$$

is a pyramid of $L+1$ levels, whose levels (v,k,i,j) at kth level contains the value $v=F(k,i,j)$. Furthermore, the function F is defined as follows (for $0\leqslant k\leqslant L$, $0\leqslant i,j\leqslant 2^k-1$):

$$F(k,i,j) \equiv \begin{cases} \sum_{p=0}^{p=1}\sum_{q=0}^{q=1} \dfrac{F(k+1,2i+p,2j+q)}{4} & \text{if } 0\leqslant k<L \\ \textit{pixel value of original scene} & \text{if } k=L \end{cases}$$

Suppose that a processor is attached to each array element in the pyramid structure. Furthermore, the $k+1$th level is partitioned into sets of 2×2 adjacent elements with all four elements in each set connected to a single "parent" processor at the kth level. The kth level is further connected in the same manner to the $(k-1)$th level. This is continued until the 0th level is reached. With this construction, the mean intensity of a $2^L\times 2^L$ image can be calculated in L steps. A straight forward raster algorithm to calculate mean intensity needs 2^{2L} steps, while a single level parallel array requires 2^L steps.

However, the reduction of computation steps is not due to the parallelism of the multiple levels of the pyramid. This is because the ancestor processor will be idle while the descendant processors are active. Rather, the strength of the pyramid lies in its non-local communication provided by the parent-child connections. To balance the wastage due to idling processors, a design incorporating heterogeneous and homogeneous processes is necessary [5]. Here, the system contains two components: a large set of simple processors connected in a pyramid structure and a small set of complex processors which have arbitrary access to the pyramid. This design provides the system with the proper resources to implement simple homogeneous processes across the image and complex heterogeneous processes within selected subimages at required resolutions.

The changing nature of the environment must also be considered. Since we are dealing with time varying imagery, while a peripheral process attempts to guide an attentive process into higher resolution levels, the image data will continue to evolve. This leads to the notion of a *pipelined pyramid* [6, 7], which is constructed as follows: At each time interval the pipeline is supplied with a new image which is inserted into the full resolution level (the pyramid base). In parallel to that insertion each parent element in the pyramid obtains its new image value by averaging the values from its immediate descendants.

3. SYSTEM DEVELOPMENT

We have developed a system that contains the homogeneous and heterogeneous components described above in a simulated parallel computing environment, known as PISCES, developed at the University of Virginia [8].

PISCES stands for *Parallel Implementation of Scientific Computing EnvironmentS*. It is a virtual system based on the use of MIMD parallel computation to achieve high computation rates for the solutions of large scale scientific and engineering problems. However, a consideration at the inception of the PISCES project was the use of such a system for asynchronous parallel image processing [9]. It was noted that a large class of parallel asynchronous algorithms for image processing could not be adequately handled by SIMD array languages and hence the need for a language that would meet the following requirements: (1) division into similar processes for different parts of a large image structure, (2) parallelism at the level of procedures rather than individual operations, (3) dynamic creation and destruction of processes and their interconnections, (4) closely coupled processing, (5) multiple simultaneous reads of shared data, and (6) sequential writing of shared data.

The PISCES system is organized as a set of "tasks" and "task clusters". The tasks may communicate with each other by passing messages through "handlers",

which are subprograms within tasks. The PISCES design permits five "granularities" of parallelism: (1) parallel execution of clusters, (2) parallel execution of tasks within a cluster, (3) parallel execution of handlers, (4) parallel execution of program segments, and (5) parallel execution of arithmetic operations. Actual levels of parallelism are implementation dependent.

The current implementation runs on a VAX 11/780 under UNIX 4.2, simulating task and cluster level parallelism, and on an Apollo network at the University of Virginia. An implementation on a new FLEX/32 MIMD computer system at NASA Langley is also under way.

4. SYSTEM IMPLEMENTATION

Three different types of tasks are defined for our system within the PISCES environment. The first is a "pyramid" task that simulates the homogeneous component described in Section 2. The "pyramid" task continuously accepts input images and at the same time outputs a pipelined pyramid structure. In doing so, the incoming image forms the base of the new pyramid and each level in the previous pyramid is pushed upwards and converged. The top level of the old pyramid is discarded. The pipelined pyramid structure used in our system contains four levels. The image size and time unit associated with each level are shown in Table 1. Except for the base which is the new incoming image, each level receives and averages image data from the preceding time frame at the immediate descendant level. Figure 1. is a schematic diagram of the four-level pipelined

level	time unit	image size
0	T_{x-3}	64×64
1	T_{x-2}	128×128
2	T_{x-1}	256×256
3	T_x (present time)	512×512

Table 1. A four-level pipelined pyramid.

pyramid containing two moving objects.

A variable number of processes form the heterogeneous component of our system and are simulated by tasks of a second type known as "agents". Each "agent" is essentially an image processor that aims to find corresponding instances of its target object in consecutive images. Moreover, it is intelligent enough to access at each time interval an appropriate subimage at a particular level in the pyramid. As its image analysis proceeds, each "agent" is able to traverse across the pyramid levels in a goal-directed manner, thus simulating the peripheral and attentive phases of motion perception. Each "agent" is also able to distribute its load by spawning new "agents" if additional possible objects of interest are detected. An "agent" may determine that its processing is redundant in relation to other agents and terminate itself. Thus the number of active "agents" varies as the system is running.

The redundancy of "agent" activities is detected by each "agent" through inter-agent communication. This communication is made possible by the third type of task in the system, known as the "scene description model" ("sdm"). The "sdm" constitutes the system's current interpretation of the dynamic scene and performs this function by serving as a repository through which "agents" can communicate status and interpretation information. The "sdm" provides a mechanism for inter-agent cooperation similar to the *blackboard* of HEARSAY-II [10]. In HEARSAY-II, however, the blackboard contains hypothesized data arranged in abstraction levels whereas the data in the "sdm" are actual results obtained from image operations and as yet are not coalesced into abstractions. The data at each level in HEARSAY-II pertain to a one dimensional ordering of various components in an utterance. In the "sdm", on the other hand, the data are essentially two dimensional representing spatial information from images with varying degrees of resolution. Timing information is also explicitly contained in the "sdm" while in HEARSAY-II the temporal ordering is implicitly reflected in the sequencing of its data.

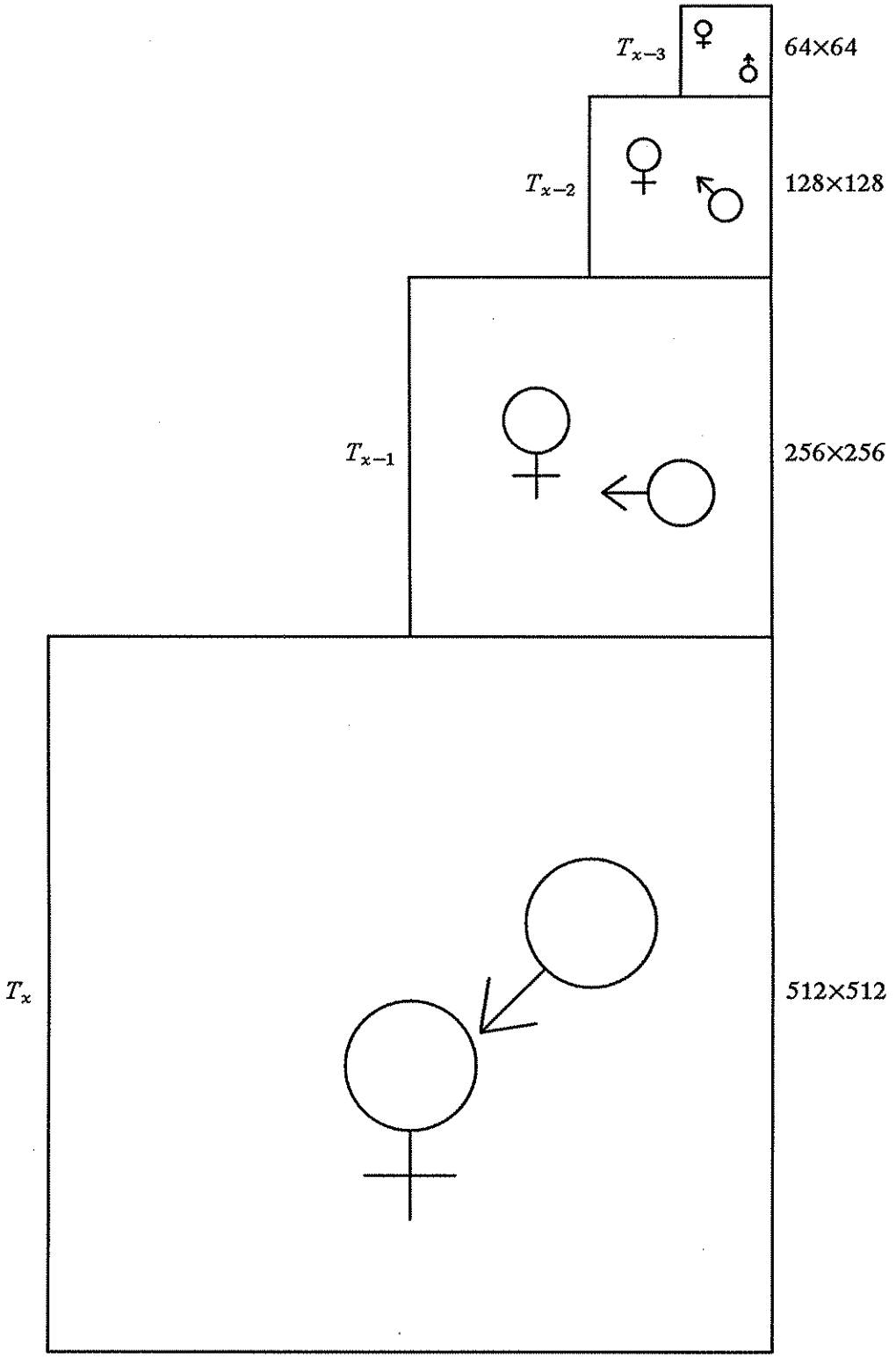

Figure 1. A four-level pipelined pyramid.

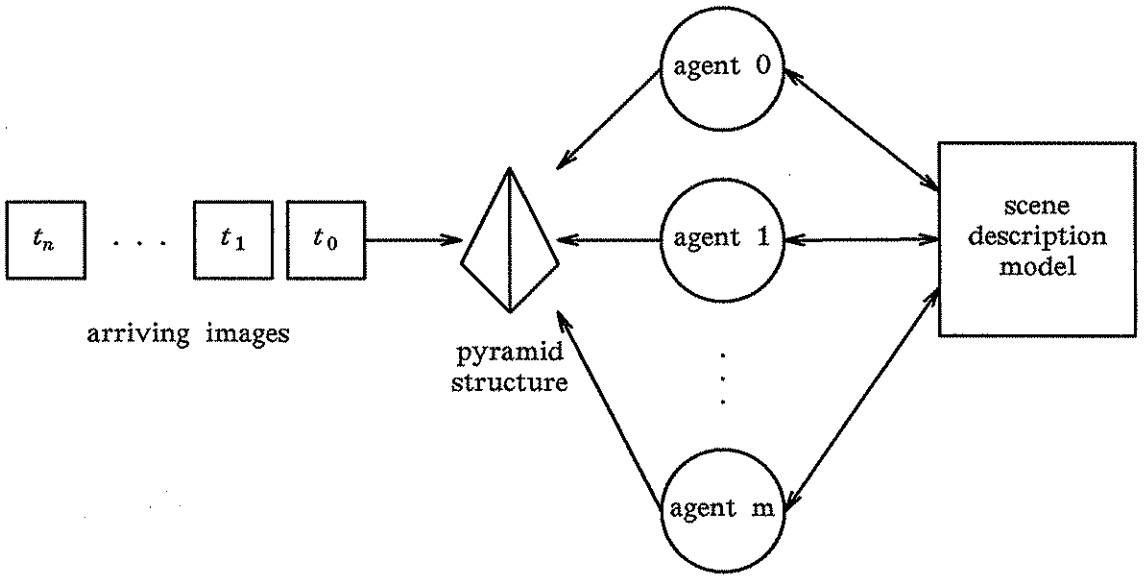

Figure 2. Various tasks in the system: the pyramid, agents, and the scene description model.

Figure 2. shows the interconnection of the three task types in the system. While the "pyramid" continues to updates pyramid images, currently active "agents" concurrently and asynchronously access subimages at various levels of the pyramid. The processing of each subimage by an "agent" is followed by a two-way communication with the "sdm" to update the current interpretation. The system is distributed and yet cooperative with each "agent" being self-directed but sharing information with other "agents". This distributed and cooperative nature is akin to some of the distributed problem solving models proposed by several researchers [11-14].

5. DISTRIBUTED DYNAMIC SCENE ANALYSIS

We will now examine how our system performs distributed analysis of a dynamic scene, similar to the airplane scenario cited earlier. The system first initiates a fixed number of "agents" to look at different pre-determined "home" windows. These windows altogether cover the entire coarsest resolution imagery, i.e., at the top of the pyramid, with marginal overlap between windows. This corresponds to the beginning of the peripheral phase.

Recall that in the airplane scenario, the person tries to locate the object with a preconceived notion of an airplane's features. Of course, there need not be just one type of object of interest. In our system, each "agent" carries with it a *set* of pattern descriptions, one for each type of object of interest. An "agent" that finds a possible instance of any one of the types of objects of interest in the peripheral phase will proceed to the next higher resolution level, thus entering the attentive phase. Subsequent positive identifications of an object will lead to progressively higher resolution levels, emulating the increasing degrees of attention in the human visual system. Any failure during the attentive phase will result in the "agent" in question returning to its "home" window for peripheral processing.

As an "agent" begins its analysis at each pyramid level, it is capable of estimating the position and size of the subimage to be accessed. Past motion data accumulated at each "agent" enables it to perform this function. In processing the subimage, the "agent" attempts a correspondence matching between the pattern descriptions of the objects of interest and any possible objects within the subwindow. There are currently three degrees of match that determine the course of the vertical traversal in the pyramid of an "agent". They are *good*, *fair* and *poor*. An "agent" that finds a good match with an object will proceed to the next higher resolution level for a higher degree of attention. A fair match indicates the necessity for further confirmation of the object and the "agent" will remain at the same level for the next time interval. When a poor match or three consecutive fair matches are obtained, the "agent" aborts its attentive process and returns to the peripheral mode.

The result of each match is first reported to the "sdm" before any new traversal in the pyramid is executed. Reporting to the "sdm" enables checking of any duplicated activities among other "agents" in the system. Thus if an "agent" finds a good match of an object that has already been reported by another "agent", then instead of pursuing that object further, the former will suspend its processing for

a specified time interval and later return to the peripheral phase. Duplication of peripheral activities in the same "home" window will also be detected by referring to the "sdm". In this case, the duplicated "agent" will kill itself rather than resuming the peripheral mode. In case several instances of objects of interest are identified within a window, the "agent" will also spawn an additional number of "agents", each of which is to examine a single object. This is again provided that each additional object is not already attended to by another "agent".

When an "agent" finally pursues an object to the base level, thereby establishing a full identification of the object, it generates a new "agent" to *track* the object. The original "agent" returns to the peripheral mode if its "home" window is not being "watched" by another "agent". The new "agent" will also move to the coarsest resolution level of the pyramid at which the object can be confidently tracked. This resembles the ease with which the person follows the airplane's movement

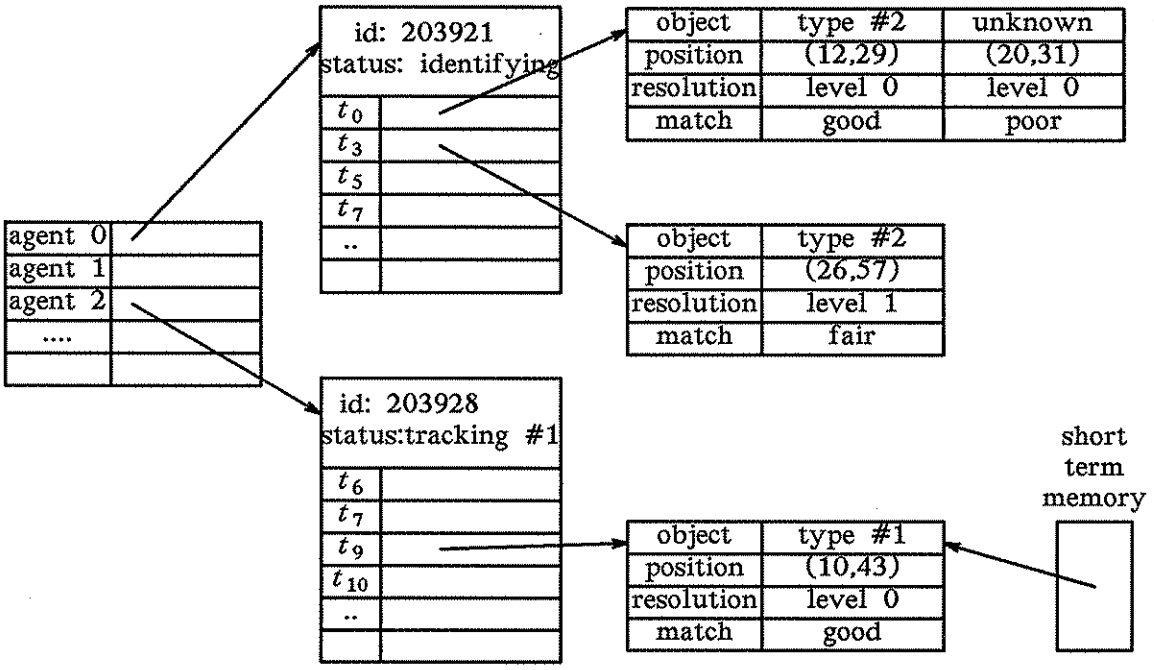

Figure 3. Data organization in the sdm's blackboard.

once the plane is located. The new "agent" thus attains a "tracking" status in contrast to the "identifying" status of its parent. The "tracking agent" also estimates its window size and position for each time unit and reports its tracking result to the "sdm". The shared information in the blackboard again prevents any other "agent" from identifying an object that is already being tracked by a "tracking agent".

Communication between each "agent" and the "sdm" is asynchronous with each "agent" reporting its own data and picking up the most current information available in the blackboard. The blackboard, as shown in Figure 3., enables an "agent" to "see" what others are doing and prevents multiple "agents" from simultaneously processing the same object. A short-term-memory also resides in the "sdm" and will be described in Section 7. Thus, we have a distributed system containing multiple "agents", variously engaged in peripheral, attentive, or tracking activities. The imagery analyzed may contain multiple moving objects, with each "agent" essentially attending to a single object. Complicated scenes, however, could give rise to ambiguous or occlusion situations that are handled in the manner described below.

6. AMBIGUITY RESOLUTION

Ambiguity occurs when an object that is being tracked moves into close proximity of other objects. The "agent" that has been tracking the object may suddenly find two or more objects in its window. Determining the correct correspondences may not be possible if the current objects are of the same type, or appear at low resolution to be similar and each of about the same distance from the last observed locations of the objects.

Disambiguation of this multiple objects case can be facilitated through information available from other "agents" which are also analyzing these objects. The

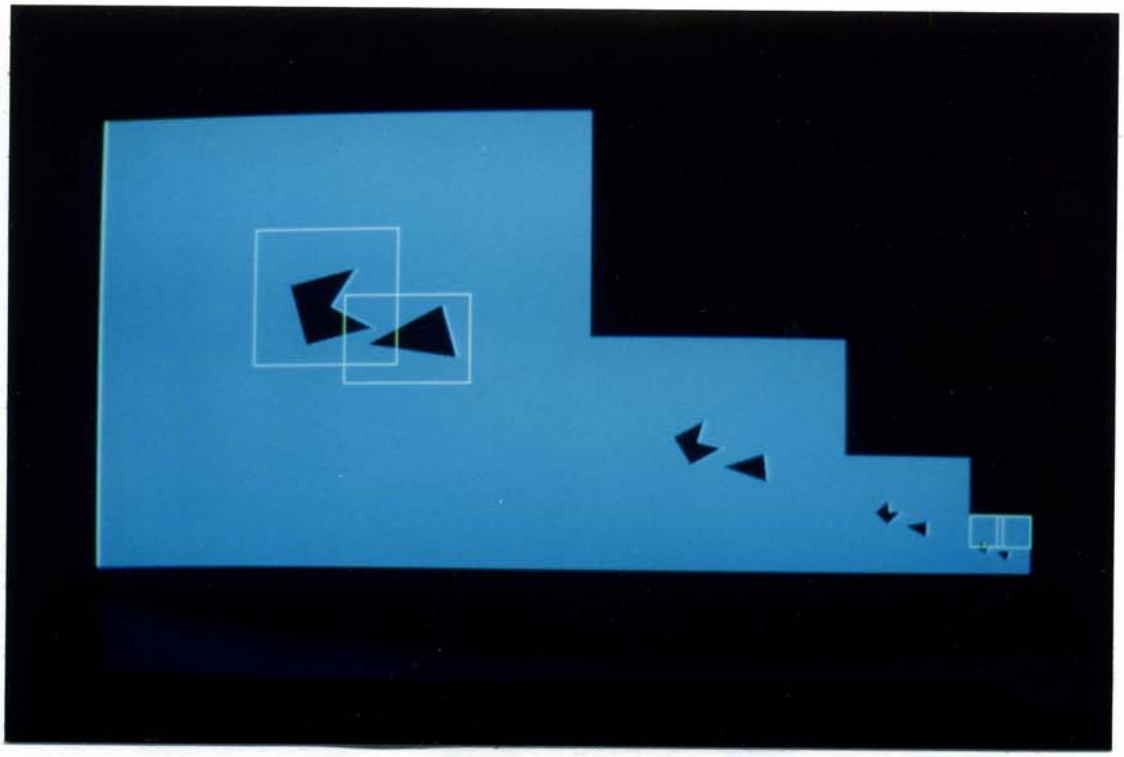

Figure 4(a). Image pyramid with two objects identified at level 3.

asynchronous communication discussed above, however, does not guarantee that an "agent" will get the best information for resolving the ambiguity because other useful data may still be on the way. The solution calls for a partial synchronization scheme in that "agents" which are involved in an ambiguous situation are temporarily synchronized in order to cooperate to resolve their ambiguities. Other "agents" access the "sdm" asynchronously as usual. A "resolver" in the "sdm" is called to perform the disambiguation function when all "agents" have at least reported once after the first report of an ambiguous state. The "resolver" first makes use of unambiguous data in the blackboard to fully or partly resolve the ambiguity through a process of elimination. The remaining ambiguity is resolved among the synchronized "agents" based on the shortest overall distances between the current positions and the last observed locations of the objects.

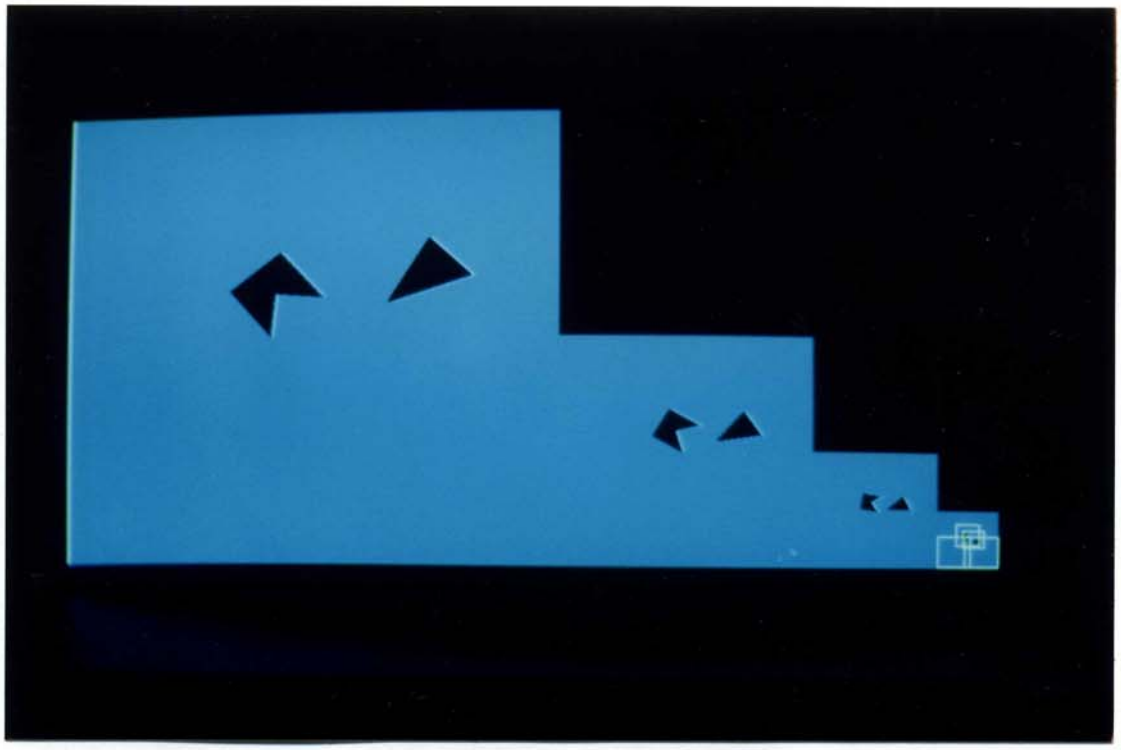

Figure 4(b). Image pyramid involving tracking with ambiguity.

Figures 4(a). to 4(c). show three instances of pyramid images involving two objects approaching each other and later moving apart. The boxes in these figures represent the windows of different "agents" in the course of identification or tracking. Figure 4(a). shows two "agents" that have fully identified the two objects at level 3 and two other peripheral "agents" at level 0. In Figure 4(b)., two tracking "agents" are involved in an ambiguous situation as their individual windows encompass images of both objects which appear identical to either "agent" at low resolution. The system is able to resolve this ambiguity based on the partial synchronization scheme described above. Figure 4(c). is a later instance in the time sequence at which the ambiguity has been resolved. Normal tracking resumes in this instance. In both Figures 4(b). and 4(c)., the tracking "agents" have a smaller window than the "home" window of each peripheral "agent". Some peripheral "agents"

are suspended at different times as result of redundancy and their windows thus do not appear during suspension periods.

7. OCCLUSION ANALYSIS

Occlusion is often a difficult problem in motion analysis [15-19]. Due to the simple nature of the pattern descriptions that we are currently assuming, even partial occlusion will render the objects involved unrecognizable. Notice, however, that for any data driven tracking system there is some level of occlusion for which the objects must be considered lost and tracking processes terminated.

To attack this problem, a short-term memory is incorporated within the blackboard of the "sdm". The short-term memory maintains the trajectories of

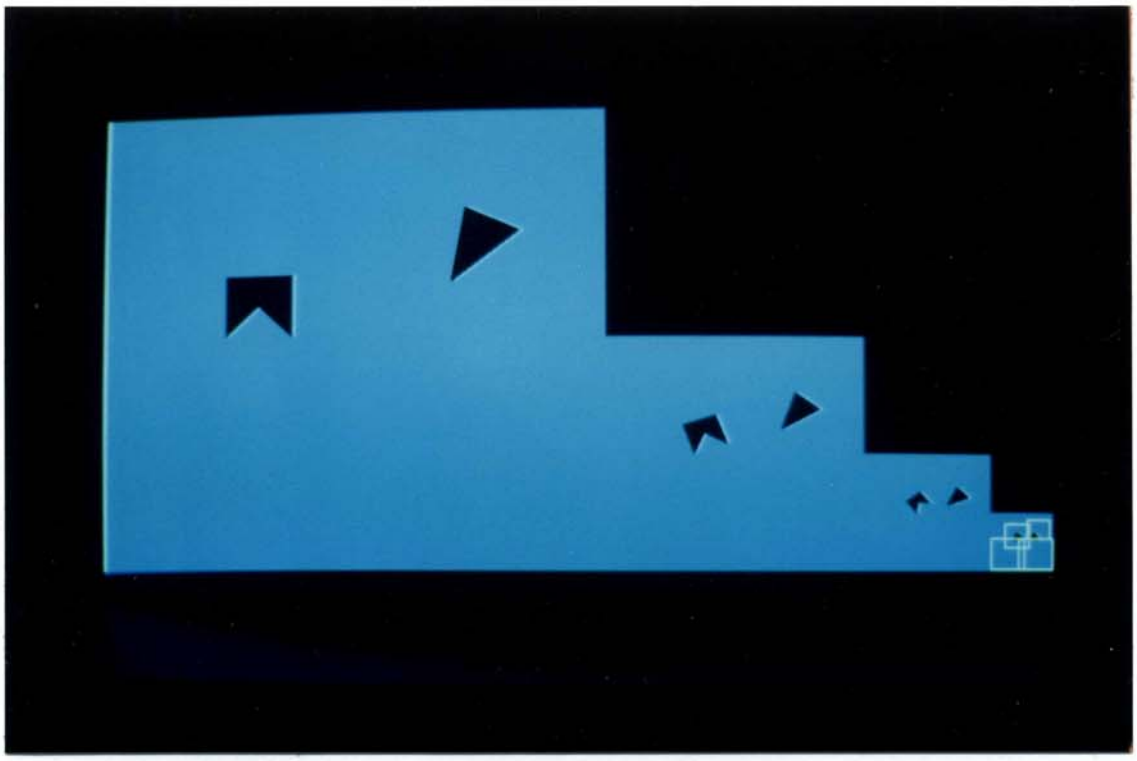

Figure 4(c). Image pyramid involving tracking with no ambiguity.

objects previously analyzed and then lost. Any newly discovered object is correlated to the information in this memory in an attempt to trace it to an object that had been tracked earlier and lost, possibly due to occlusion. Statistical correlation and curve fitting techniques are used in sorting through data in the short-term memory. It is apparent that objects that were tracked but lost too long ago will have little effect in correlating with newly found objects. Thus items in the short-term memory are made to "age" with the passage of time and those older than a certain age are expelled from the short-term memory.

Figure 5. shows an object moving so as to become occluded by a stationary object. The trajectory of the former is retained in the short-term memory during the process of tracking.

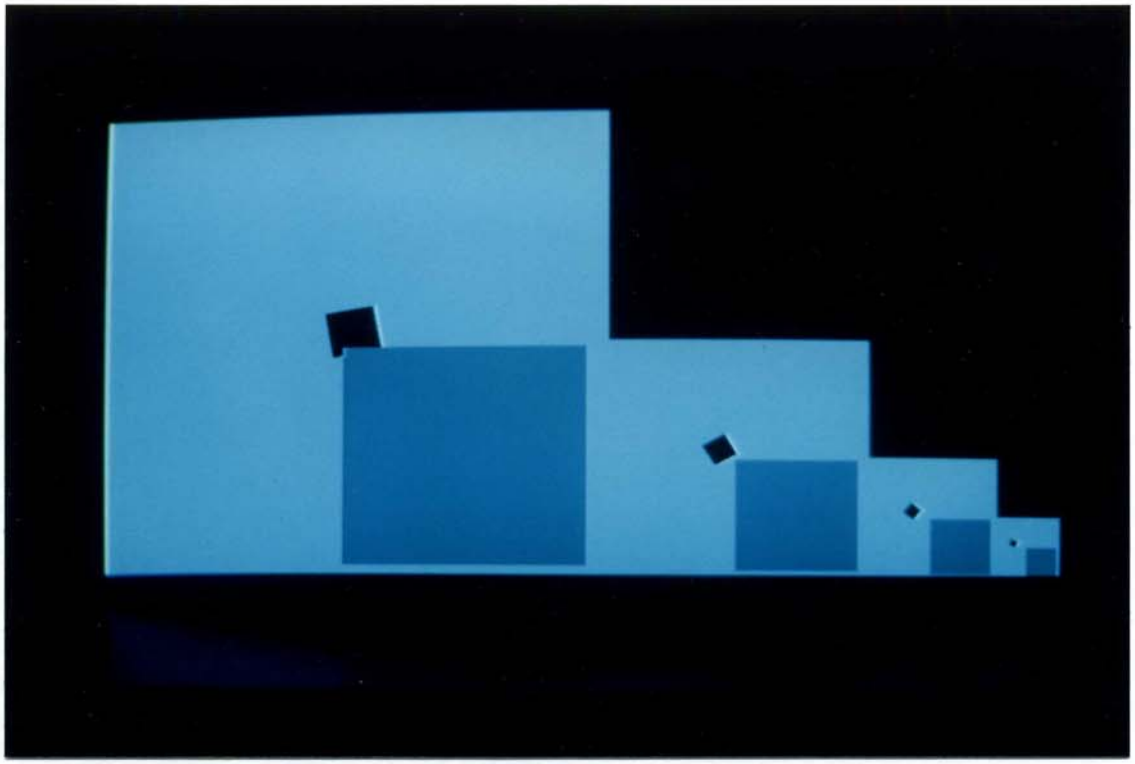

Figure 5. Image pyramid involving occlusion.

8. CONTINUOUS IMAGE ARRIVALS

In our initial phase of implementation, image arrivals are partially synchronized to "agent" activity to allow us to study the system's behavior without the concern of an "agent" missing image data. In this partially synchronized case, the "pyramid" supplies new images only after all "agents" have completed a cycle on the current images. This synchronization is provided by the "sdm" which signals the "pyramid" to update new images at the end of each cycle. Such synchronization appears rather unnatural when in biological visual systems images arrive continuously at the retinal cells regardless of whether the brain is able to process them. The capability of tracking with continuous image arrivals is also vital if real-time performance is required.

This constraint on image arrivals has been removed in our present design. The "pyramid" now acts as a front end sensory system and has no knowledge of the processing activities among the "agents" and the "sdm", apart from sending subimages to individual "agents". The "pyramid" will update the pyramid structure whenever a new image arrives. As an "agent" may be delayed by a lengthy computation in the analysis of a requested subimage, two consecutive requests from that "agent" may not necessarily receive two images that are immediately adjacent in the temporal sequence. Each "agent" will now have to be robust in continuing its control on pyramid traversal and window selection despite occasional missing data in the "sdm". This is made possible by the fact that each image frame now carries an explicit time stamp. Moreover, because "agents" may miss image frames, there may be substantial temporal gaps between instances of an object reported by different "agents". An "agent" may thus report an instance of an object appearing at a much later time stamp than those reported earlier by other "agents" which have yet to report again because of computational delays. In order to determine whether the current "agent" is redundant, i.e. analyzing an object already attended to by another "agent", the current "agent" will have to reconcile its reports with

others in the blackboard. In doing so, each "agent" normalizes blackboard data with respect to its own time frame. As shown in Figure 3., time units reported by individual "agents" are not necessarily consecutive. In addition, overall motion interpretation from the blackboard has to take into account these temporal gaps.

The continuous image arrival design has been tested with the same sets of imagery as that in the original design including those involving ambiguity and occlusion. The "agents" tolerate occasionally missing several frames. The tolerance depends on the the assumed maximum displacement between consecutive image accesses assumed by the window selection algorithm of an "agent". If an "agent" is extensively delayed then it may lose the object it was attempting to track. Alternatively, the object may partially extend beyond the selected window resulting in only a fair match, thus slowing the identification process. The system fails to detect objects that are moving faster than a certain limit, while it continues working reasonably well with missed frames involving slower objects. This performance degradation also reflects the confusion or optical illusions in human vision with complicated and fast changing scenes.

9. DISCUSSION

It has long been recognized that motion perception plays a central role in biological visual systems [20]. For example, lower animals, such as frogs, perceive only prey that is moving [21]. Our experience in detecting with ease a camouflaged object in its momentary movement is another example. Inspired by biological and psychological research, computer analysis of time varying imagery has found wide applications despite its relatively short history. Examples of areas of applications include medicine, communication, manufacturing, military, etc. [22-24]. In the course of these developments, many ingenious methods have been devised to provide motion detection and measurement [3, 20, 25]. Basically, these techniques fall into two different schemes, namely, intensity-based schemes and token-matching

schemes.

In intensity-based schemes, motion measurements are based on the local changes in light intensity values. In some early studies, changes were detected by cross-correlation. Leese, *et al.* [26], and Smith and Phillips [27], for instance, measure cloud motion from satellite images based on gray-level cross-correlation. Temporal-spatial gradient techniques are other methods for detecting change in intensity at an image point over both time and space to estimate the rate of movement of the underlying surface. Various methods using gradient techniques have been proposed by several authors [28-30].

In token-matching schemes, identifiable features or tokens are located and then matched over time. For instance, Chow and Aggarwal [16], in analyzing the motion of planar figures, match figures on the basis of pattern descriptions, such as area and principle axes. The use of tokens that only represent partial objects are found in several studies. For example, feature points that correspond to distinguishing parts of objects, such as corners, are extracted for matching as used by Barnard and Thompson [31], and Lawton [32]. In the work by Martin and Aggarwal [17], Jacobus, *et al.* [33], and Tsuji, *et al.* [34], matching is established between small fragments of the objects' boundaries.

After motion has been detected and measured it may be used by higher level processes to make interpretations of the observed scene. An example of this is in the recovery of three-dimensional shape [20]. Another example is in achieving even higher abstractions from the motion measurements to give semantic interpretation for scenes. These semantic abstractions require a knowledge base and an inference engine to achieve meaningful motion understanding [35-37]. An actual implementation of such semantic abstractions has been realized with a system called ALVEN [35, 38]. This system measures the human left ventricle wall motion from x-ray cineangiograms and then determines the heart muscle competence.

In our system, there is a motion detection and measurement mechanism in each "agent". The method used in each "agent" is of a token-matching scheme based on a set of pattern descriptions similar to that developed by Chow and Aggarwal [16]. One reason for using a token-matching scheme is that token-matching schemes generally achieve a higher degree of accuracy than intensity-based schemes but at the price of more extensive processing [20]. However, the problem of extensive processing required for token-matching schemes is not as acute in our system due to the multiprocessor design and the capability of each "agent" to restrict its processing window.

As various motion measurements are channeled from different "agents" to the "sdm", the system attempts a simple motion interpretation of the whole scene by providing the motion parameters of all the objects that appear in the scene. This is made possible by the data aggregated in the blackboard in the "sdm". The blackboard fits into the general definition of the blackboard architecture given by Hayes-Roth [39]. It serves as a global data base for scene interpretation as well as an intelligent control for the system's functions. Currently, the system performs only a direct abstraction from the blackboard for a simple scene interpretation without semantic analysis. The system is thus basically a tracking system, with "agents" tracking *individual* objects. It may not be suitable for imagery involving general *global variations* such as in cineangiography and agricultural land aerial picture sequences. The tracking ability of the system, however, will find applications in scenes containing localized movements. Traffic monitoring, missile tracking, and biomedical studies of organism and animal motion are some examples of such applications.

The tracking performance of each "agent" depends on the motion detection and measurement technique it uses. The design of our system, however, is not dependent on the motion analysis capability of each "agent". The modular design of the system permits, for instance, "agents" to adopt different motion estimation

techniques. The "sdm" may also incorporate various knowledge representations and an inferencing structure thereby allowing a high-level semantic interpretation of tracking data.

REFERENCES

[1] S. Tanimoto and A. Klinger, eds., *Structured Computer Vision*, Academic Press, 1980.

[2] L. Uhr, "Psychological motivation and underlying concepts," in *Structure Computer Vision*, ed. S. Tanimoto and A. Klinger, pp.1-30, 1980.

[3] W.N. Martin and J.K. Aggarwal, "Survey: Dynamic scene analysis," Computer Graphics and Image Processing, Vol. 7, pp. 356-374, 1978.

[4] P.R. Cohen and E.A. Feigenbaum, eds., *The Handbook of Artificial Intelligence*, Vol. III, Chapter XIII Vision, Section E1. Pyramids and Quad Trees, pp.279-286, 1982.

[5] B.H. Thomas and W.N. Martin, "Heterogeneous and homogeneous processes in multiresolution dynamic scene analysis," Eighteen Annual International Conference on System Sciences, Honolulu, Hawaii, Jan.2-4, 1985.

[6] P.J. Burt, C. Yen and X. Xu, "Multi-resolution flow-through motion analysis," IEEE Conf. on Computer Vision and Pattern Recognition, Washington D.C., pp.246-280, 1983.

[7] B.H. Thomas and W.N. Martin, "Pyramid structures in dynamic scenes," Proceedings ICRP, Montreal, Canada, pp. 1008-1010, 1984.

[8] T.W. Pratt, *PISCES User's Manual*, Version 3, University of Virginia, August 1984.

[9] R.L. Douglass, "The requirements of a language for asynchronous parallel image processing," in Proceedings of the 1980 International Conference on Parallel Processing, IEEE, pp.147-148, 1980.

[10] L.D. Erman, F. Hayes-Roth, V.R. Lesser and D.R. Reddy, "The HEARSAY-II speech understanding system: Integrating knowledge to resolve uncertainty," Computing Surveys, Vol. 12, No. 2, pp. 213-253, 1980.

[11] S. Cammarata, D. McArthur and R. Steeb, "Strategies of cooperation in distributed problem solving," Proceedings IJCAI-83, pp.767-770, 1983.

[12] V.R. Lesser and D.D. Corkill, "Functionally accurate, cooperative distributed systems," IEEE Trans. Systems, Man and Cybernetics, Vol. SMC-11, No.1,

pp.81-96, 1981.

[13] V.R. Lesser and D.D. Corkill, "The application of artificial intelligence techniques to cooperative distributed processing," Proceedings IJCAI-79, pp.537-540, 1979.

[14] R.G. Smith and R. Davis, "Frameworks for cooperation in distributed problem solving," IEEE Trans. Systems, Man and Cybernetics, Vol. SMC-11, No.1, pp.61-70, 1981.

[15] J.K. Aggarwal and R.O. Duda, "Computer analysis of moving polygonal images," IEEE Trans. Computers, Vol.C-24, pp.966-976, Oct. 1975.

[16] W.K. Chow and J.K. Aggarwal, "Computer analysis of planar curvilinear moving images," IEEE Trans. Computers, Vol.C-26, pp.179-185, Feb. 1977.

[17] W.N. Martin and J.K. Aggarwal, "Computer analysis of dynamic scenes containing curvilinear figures," Pattern Recognition, Vol.11, pp.169-178, 1979.

[18] R. Peterman, "Computer analysis of planar motion of polygons," Master Thesis, University of Austin, Jan. 1975.

[19] M.O. Ward and Y.T. Chien, "Occlusion analysis in time-varying imagery," IEEE Conf. on Pattern Recognition and Image Processing, Dallas, Texas, pp.504-507, 1981.

[20] S. Ullman, "Analysis of visual motion by biological and computer systems," Computer, Vol.14, No.8, pp.57-69, 1981.

[21] G. Johansson, "Visual motion perception," Scientific American, Vol.232, pp.76-89, 1975.

[22] H.-H. Nagel, "Image sequence analysis: what can we learn from applications?" in *Image Sequence Analysis*, ed. T.S. Huang, Springer-Verlag, pp.19-141, 1981.

[23] H.-H. Nagel, "Overview on image sequence analysis," in *Image Sequence Processing and Dynamic Scene Analysis*, Springer-Verlag, ed. T.S. Huang, pp.2-39, 1983.

[24] B. Radig and H.-H. Nagel, "Evaluation of image sequences: A look beyond applications," in *Digital Image Processing*, eds. J.C. Simon and R.M. Haralick, pp.525-560, 1980.

[25] W.B. Thompson and S.T. Barnard, "Lower-level estimation and interpretation of visual motion," Computer, Vol.14, No.8, pp.20-28, 1981.

[26] J.A. Leese, C.S. Novak and V.R. Taylor, "The determination of cloud pattern motions from geosynchronous satellite image data," Pattern Recognition, Vol.2, pp.279-292, 1971.

[27] E.A. Smith and D.R. Phillips, "Automated cloud tracking using precisely aligned digital ATS pictures," IEEE Trans. Computers, Vol.21, pp.715-729, 1972.

[28] C.L. Fennema and W.B. Thompson, "Velocity determination in scenes containing several moving objects," Computer Graphics and Image Processing, Vol.9, pp.301-315, 1979.

[29] J.O. Limb and J.A. Murphy, "Estimating the velocity of moving objects in television signals," Computer Graphics and Image Processing, Vol.4, pp.311-327, 1975.

[30] B.K.P. Horn and B.G. Schunck, "Determining optical flow," Artificial Intelligence, Vol.11, pp.185-203, 1981.

[31] S.T. Barnard and W.B. Thompson, "Disparity analyses of images," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No.4, pp.333-340, 1980.

[32] D.T. Lawton, "Processing translational motion sequences," Computer Vision, Graphics and Image Processing, Vol.22, pp.116-144, 1983.

[33] C.J. Jacobus, R.T. Chien and J.M. Selander, "Motion detection and analysis by matching graphs of intermediate-level primitives," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol.2, No.6, pp.465-510, 1980.

[34] S. Tsuji, M. Osada and M. Yachida, "Tracking and segmentation of moving objects in dynamic line images," IEEE Trans. Pattern Analysis & Machine Intelligence, Vol. PAMI-2, No.6, pp.516-542, 1980.

[35] J.K. Tsotsos, J. Mylopoulos, H.D. Covvey and S.W. Zucker, "A framework for visual motion understanding," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No.6, pp.563-573, 1980.

[36] J.K. Tsotsos, "Knowledge and the visual process: content, form and use," Pattern Recognition, Vol.17, No.1, pp.13-27, 1984.

[37] J.K. Tsotsos, "The scope of motion research: from image intensity changes to semantic abstractions," Computer Graphics, Vol.18, No.1, pp.7-9, Jan. 1984.

[38] J.K. Tsotsos, "Knowledge organization and its role in representation and interpretation for time-varying data: the ALVEN system," Comput. Intell., Vol.1, pp.16-32, 1985.

[39] B. Hayes-Roth, "The blackboard architecture: A general framework for problem solving?" Heuristic Programming Project Report, No. HPP-83-30, Stanford University, 1983.