CONCEPTUAL HIERARCHICAL IMAGE PROCESSOR (CHIP), AN ARCHITECTURE FOR REAL-TIME MACHINE VISION

R W M Smith, C J Radford, P G Ducksbury, G Brown

Introduction When the transputer family of processors was first announced, it promised the availability of massively parallel computing architectures, in which arbitrary processing power might be achieved by interconnecting sufficient transputers. This was of particular interest in image processing (IP), which has always been very hungry for computational power, due to the large data structures (images typically 512*512 bytes), and the high data rates for real time processing (typically 25 to 30 frames per second).

It soon became apparent from experience with architectures like the Machine Vision Transputer Array System (MVTAS) [1], that arrays of transputers were not suitable for implementing low-level image processing primitive operations such as convolution, thresholding and histogramming, as well as many intermediate-level feature extraction operations such as connected component labelling. This is because the transputer's CPU is not sufficiently powerful for one processor to perform these types of operation at pixel rate, and while an array of several devices may be able to achieve the required processing speed, it would be uneconomic, due to the availability of commercial DSP devices which can perform these operations in real-time. Transputer arrays are much more suited to performing the high-level parts of an IP application, such as feature recognition or target tracking. These operations typically require complex programming, using decision-making constructs and abstract data structures which cannot easily be mapped onto dedicated hardware.

The Conceptual Hierarchical Image Processor (CHIP) is an architecture for real time IP applications. It combines a transputer array with an collection of low level processing modules interconnected using a reconfigurable crossbar switch to provide appropriate processing support for both low-level and high-level IP operations. The architecture is flexible and expandable, in that new low-level modules can be added to the architecture as required by each application.

CHIP is designed to be a test-bed for new IP algorithms and new processing module architectures. IP applications can be developed on the CHIP system, and the required low-level and high-level processing resources can be identified. New low-level modules may be prototyped very simply, using commercial off-the-shelf (COTS) devices where available. Once the algorithm and the necessary hardware have been developed and proved, a final target version may then be constructed. This will typically be much simpler than the CHIP, since it will not require the same level of flexibility or reconfigurability. The target may be implemented in an alternative technology, for example, by using ASICs for the processing modules, to meet any size constraints (for example, in airborne or hand-held systems).

Block diagram A block diagram of the CHIP architecture is shown in figure 1. It consists of a group of low-level processing modules, interconnected by a crossbar switch, and a high-level processing engine which contains an array of 20 transputer modules.

Defence Research Agency, St. Andrew's Road, Malvern, WR14 3PS

Video input is from an acquisition module, which accepts a CCIR interlaced monochrome signal. The video is digitised to 512 by 512 pixels, with eight bits per pixel. Video output is on a RGB colour display. Pseudo-colour may be generated on the output by means of a RAMDAC. The display also has the capability to produce overlay graphics on the output image, for annotation of targets for example.

In addition to the video connections shown in figure 1, most of the processing modules contain a transputer of one form or another. The links of the transputers are routed through a IMSC004 routing switch, to allow flexible configuration of the transputer architecture, under software control.

Video crossbar The video crossbar interconnects all of the low-level processing modules, the transputer array, and the acquisition and display module. It is a non-blocking crossbar switch, in which any input may be connected to any one or more outputs. Each input and output is nine bits wide, to accommodate eight pixel data bits and an accompanying synchronisation signal. The synchronisation bit enables processing modules to synchronise to the video data, irrespective of any latency in the data as a result of previous processing modules in a pipeline.

Two versions of the crossbar have been produced, one with 64 inputs and 64 outputs, and the other with 32 inputs and 32 outputs. They are both based on the LSI LOGIC L64270 crossbar switch, in a bit-slice manner, using one per bit for the 64 I/O version, and one per two bits for the 32 I/O version.

The configuration of the crossbar switch is double buffered, and is clocked by the frame-sync pulse. This allows complex configuration changes to be programmed during a whole frame period, and to take effect instantaneously at the next frame-sync. The crossbar is controlled by a transputer, which also conventionally serves as the master transputer within CHIP.

Low-level processing modules A number of low-level processing modules have been produced so far. These include a pipeline of four A110 convolution processors. Each A110 may operate individually, giving four separate filtering kernels each of up to three rows of seven pixels. Alternatively, they may be combined to give larger filter kernels. The pipeline board has an expansion connector (which does not go to the video crossbar), which allows two boards to be connected together to extend the kernel size still further. Each pipeline of four A110s is programmed by a T225 transputer.

Another processing module that has been implemented is a dual-input dual-output lookup table. This simple device has two video inputs which are concatenated to address a 64 K by 16 bit RAM-based lookup table. The 16 bit output from the LUT forms the two video outputs. By appropriate programming of the LUT, any pair of arbitrary functions of two variables can be performed at pixel rate to a precision of eight bits.

A connected component labeller (CCL) module is in the final stages of testing. CCL is a very common operation in IP algorithms. This hardware module will allow the acceleration of many IP applications which currently implement CCL in software.

Under development is an image warping module, based on the TRW TMC2302 Image Manipulation Sequencer. This can perform bilinear transformation in real-time, and more complex operations at a lower frame rate. A regional histogramming module is also planned. This will generate histograms for defined regions in the image (or for the image as a whole).

In addition to the processing modules, special interfaces have been designed to allow digital image data from two different infra-red staring array imaging systems to be input directly, bypassing the acquisition circuit.

Transputer array A block diagram of the transputer array is shown in figure 2, and a block diagram of one of the transputer modules is shown in figure 3. Each module consists of a T805 transputer, with 4 MByte of DRAM and 2 MB of VRAM. The VRAM is divided into two banks, each of which contains four framestores (256 KBytes each). The serial I/O connections of the VRAM are connected to the video input and output buses that interconnect all modules in the array. Each bank of VRAM is connected to a video input bus and a video output bus. In addition, the video output from each bank is connected to another pair of buses, called the MAP buses (one for each framestore). These buses are used to control the way in which video results distributed through the array are recombined together to form a single image, and their operation is described below. Control of the VRAM and associated DMA operations is performed by a single EPLD on each module.

An input image (from the video crossbar), is distributed on one of the two input buses, A or B, to all transputer modules simultaneously. Each transputer can select whether or not to acquire the image, and to select which of the four framestores in the appropriate bank into which to acquire it. In applications where the transputer array generates a result in the form of an image, the result will be distributed amongst the modules in the array, and must be recombined together to form a single coherent image. To recombine the results together, one of the transputer modules is designated as the array controller. The selection is done either by front panel switches, or by a transputer link adapter. Any module may be chosen as the controller and separate controllers may be used for buses A and B. The selected module has its corresponding map output enabled, and all other modules have the map output disabled. A framestore in the selected VRAM bank on the controller will have been initialised with a map, which defines for each pixel in the result image, which worker module will produce that pixel. The controller and all active transputer modules output simultaneously. The map output from the controller is decoded, to produce one of twenty output enable signals, one for each module in the array. This enables the pixel output from the selected worker module, and the pixel is enabled onto the video output bus. This process is repeated for every pixel in the image.

During video I/O, the transputer modules are free to access both the DRAM and VRAM. The only impact that the video I/O operation has on the transputer is that a single DMA memory cycle must be performed once every four rows of video to transfer from the VRAM serial shift register to the RAM array.

The transputer links are all connected to a set of four C004 link crossbar switches, where any network configuration can be realised. The C004s also allow connections to the main C004, to connect the transputer array to the rest of the CHIP link network.

Programming model The CHIP system can be programmed using standard transputer toolsets, such as the INMOS OCCAM or ANSI C toolsets, and can be hosted by any computer that supports these tools. To date, the system has been hosted by SUN, VAX and IBM PC computers.

To the programmer, CHIP appears as an array of transputers, of various types. The links between the transputers may be set up in an arbitrary configuration, prior to running the main application code. Transputers on modules which have specific functions, control the operation of the hardware by means of memory mapped registers. Framestores in the transputer array and elsewhere are accessed as two-dimensional memory mapped arrays. To abstract the applications programmer from specific hardware details, system software libraries have been produced for each module, so that the hardware may be programmed by means of procedure calls.

Interprocessor synchronisation is achieved by two means. The simplest mechanism is by transfer of messages between transputers, implicitly synchronising the processors at the point where the message is passed. In addition, all transputers have their EVENT input connected to the FRAME

SYNC signal, and a high priority process which runs on every processor waits on the EVENT, allowing processor to synchronise explicitly to the video frame sync.

Physical implementation The CHIP system is currently implemented as a 19" double Eurocard rack. The low-level modules each occupy a single or double Eurocard space. There is no backplane for this part of the system; the DIN41612 96-way connectors have wire wrap tails, to which all video, transputer and other connections are connected using header sockets. This makes changes and additions very easy to implement.

The transputer array consists of twenty identical single Eurocard modules, arranged on two backplanes of ten modules each, interconnected by double Eurocards which contain the input buffer, output buffer and link crossbar. The video crossbar is a board approximately 25 cm square, mounted behind the rack on pillars. All video connections to and from the video crossbar are done using ribbon cable. Video clocks are distributed from a small buffer board using twisted pair ribbon cable. All transputer links within the low-level modules, are implemented with twisted pair ribbon cable.

Power is from a 100 A power supply, mounted on the rear of the case. The transputer array consumes approximately 40 A when all nodes are continuously accessing DRAM so the power supply is sufficient for all foreseeable additions to the system.

Applications - Sobel convolution To illustrate the ease with which low level processing modules can be interconnected, a Sobel edge enhancement convolution has been implemented. A block diagram of the configuration is shown in figure 4. It uses two of the four A110s, and to convert the vertical and horizontal edge intensity components to magnitude and angle components, the dual LUT was used. The LUT is initialised using a transputer from the array, but once this is done, all processing is carried out by the low-level modules at pixel rate.

Applications - Pearl-Bayes Network texture segmentation A Pearl-Bayes Network (PBN) is essentially a directed acyclic graph in which the vertices (nodes) represent events and the arcs (links) represent relationships between events. The technique is a bayesian approach for evidential reasoning which is used here to combine evidence from several sources in order to obtain the belief of some event. The technique was assessed in two applications, the first being the location of urban regions in airborne infrared reconnaissance imagery, and the second the location of driveable regions in autonomous land vehicle imagery.

The geometric approach to parallelism was used and an image was split into non overlapping regions (typically 16x16 pixels in size) for which a set of statistics were computed (edges, maxima/minima, grey level distribution type) prior to being converted into evidence for the network. A multilevel approach was developed in which the evidence from several different resolutions of an image were combined into a belief of an event, the reason for this being to introduce some contextual knowledge into the system (when attempting to classify a region of an image it makes sense to look at neighbouring regions). References [2] and [3] contain a description of the algorithm.

The approach is ideally suited for parallel architectures as it is deterministic and the speedup is linear with the number of transputers processing the image.

Figure 5 shows output for the urban region location with an outline drawn around the areas which have the highest probability of being urban, whilst figure 6 shows a mask over an area which is considered to be driveable in an image taken from a forward-looking camera mounted on a road vehicle.

Applications - Probabilistic Relaxation (ridge detection) Probabilistic Relaxation is concerned with assigning labels to image objects. In the example discussed here, the aim was to detect ridge

features (for example, roads) within airborne reconnaissance images. The objects in this case are image pixels and the labels assigned to them denote possible classifications, namely that the pixel belongs to a vertical ridge, a horizontal ridge or no ridge.

After smoothing the image using a 2-dimensional Gaussian filter, the image is then filtered in two further convolutions both of which use a 1-dimensional ridge kernel. The first of these is applied in the vertical direction whilst the second is applied in the horizontal direction. The filtered output from the convolutions is then used to create a probabilistic representation for each of the ridge classification labels, these are then applied as the input to the probabilistic relaxation framework.

The relaxation is achieved by recursively combining evidence for the different label categories. The evidence combining procedure uses a dictionary to improve the connectivity of the resulting edge map, the dictionary consists of a set of rules which are the most consistent edge labellings over a 3x3 neighbourhood. Full details of the approach can be found in reference [4], whilst reference [5] gives more detail on the parallel implementation.

The algorithm is non-deterministic, unlike the PBN application discussed above. Geometric parallelism was used to divide the image, and dynamic load balancing in the form of a work-farm was employed to ensure all processors were occupied as much as possible. Because of the iterative filtering nature of the algorithm, the resulting image shrinks at the boundaries with each iteration. Boundary problems were dealt with in this case by use of overlapping regions. Linear speedup is therefore not possible with this algorithm and with the initial Gaussian filtering being computed using A110 convolvers, the algorithm required 89 seconds using 1 controller and 19 worker transputers arranged in a ring structure. Indications were that if the algorithm could be expressed using 8-bit arithmetic then a hardware implementation may be possible.

Figure 7 shows the result of the relaxation to locate the road like structures in the image. Note that this image is heavily urbanised and therefore the processing time required probably represents an upper bound.

Applications - Target detection and tracking in infra-red imagery The detection of ground targets from IR staring array imagery requires several algorithms for reliable results. For this application, a local variance thresholding filter and a local mean thresholding algorithm were used. Both techniques pass a window over the image to determine the local variance and mean of the IR image. The combined results from these filters was passed to a connected component labeller for target extraction.

A nearest neighbour correlation tracker labels the detected targets as they manoeuvre within the field of view. Gating regions and a predictor filter resolve the multi-target, multi-track association problem and scenarios such as crossing targets have been simulated with no misassociation.

Figure 8 shows how the target detection and tracking algorithms are segmented for CHIP. The computationally intensive variance and mean filters are split across four transputer modules each. The tracking association and prediction are performed in parallel on two transputer modules. The detection and track co-ordinates are passed to the overlay graphics module for display on the original IR image. The application has so far been implemented using only the transputer array, but a further development would be to build low-level modules to perform the variance and mean thresholding operations, and to perform the CCL operation with the hardware currently under development. This would allow near-frame rate operation.

Conclusions The CHIP system is a flexible and expandable architecture for real-time image processing. It provides a hierarchical selection of hardware processing modules, which support low-level, pixel intensive operations, intermediate level feature extraction, and complex high-level operations.

New low- and intermediate level processing modules may be added to the system with relative ease, due to the use of a simple interprocessor video protocol and a low pixel clock speed. An array of 20 transputer image processing modules provides a processing engine for high-level operations, with special video buses to input and output images directly to and from the transputers' memory with minimal impact on the transputers' available processing time.

A number of low-level processing modules have already been constructed, and more are in development. Further modules will be produced as required by the applications investigations which are being pursued.

Several applications have already been implemented on the CHIP system, ranging from a simple Sobel edge enhancement, to complex image segmentation algorithms and target tracking applications.

Work is now progressing in a number of directions. Further applications are being implemented to demonstrate the abilities of the architecture. In addition, the limitations of the architecture are being addressed. The main limitations are twofold. First, the computational power offered by the array of transputers is inadequate for many of the algorithms that are planned. Increasing the size of the array would be one possible solution to this problem, but in many cases, the overheads associated with adapting an algorithm for parallel processing mean that this may become counter-productive. An alternative solution is to select a more powerful processor as the basis of the array. A number of processors are currently being benchmarked for use as the basis of a new high-level processing array, including the T9000 transputer, the TMS320C40, the i860, the ALPHA and the PowerPC. The latter three devices do not directly support multi-processing, but a hybrid module based on a transputer for interprocessor communications, with a shared memory interface to the chosen microprocessor could be used to overcome this limitation.

The second limitation that is becoming apparent is that many modern commercial and military sensors have a data bandwidth much higher than can currently be supported by CHIP. For example the Thermal Imaging Common Module (TICM) has a bandwidth of 15.2 M pixels per second and HDTV has a bandwidth in excess of 100 M pixels per second. To address this limitation, the next version of CHIP will use a different interprocessor communications protocol that will have a higher bandwidth.

References

[1] Barrett D.J. "Image processing and video rate I/O on transputer arrays," IEE Colloquium on Transputers for Image Processing Applications, 13th Feb. 1989, 4/1-4/7

[2] Ducksbury P.G., `Driveable region detection using a Pearl Bayes Network', IEE Colloquium on Image Processing for Transport Applications, London, 9 Dec 1993.

[3] Ducksbury P.G., `Parallel Texture Region Segmentation using a Pearl Bayes Network', British Machine Vision Conference 93, University of Surrey, Guildford, UK, 21-23 Sept, 1993.

[4] Hancock E.R., Kittler J., `Edge-Labeling using Dictionary-Based Relaxation', IEEE PAMI, vol 12, no 2, Feb 90.

[5] Ducksbury P.G., `Parallel Processing for the Vision by Associative Reasoning Project : Final Report', DTI IED-1936 Vision by Associative Reasoning project, July 1993.

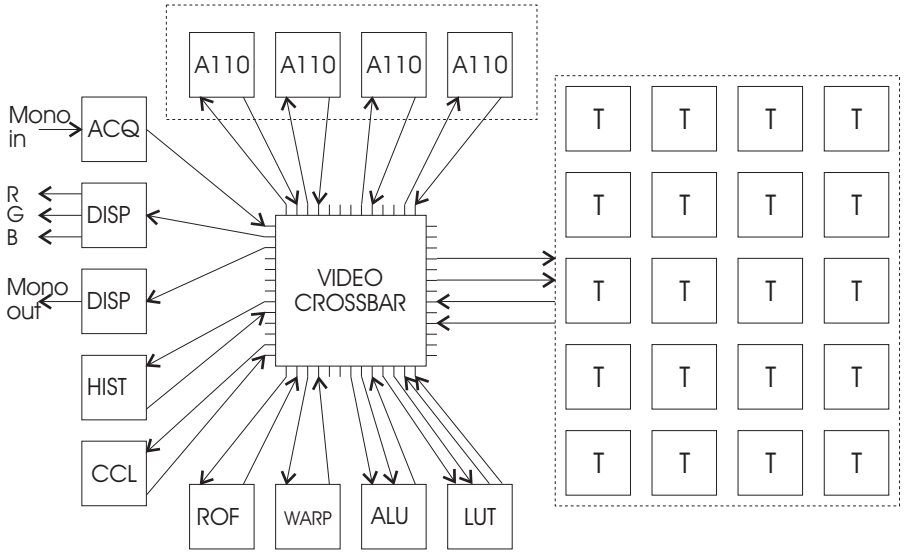

Figure 1. Block diagram of the CHIP system.

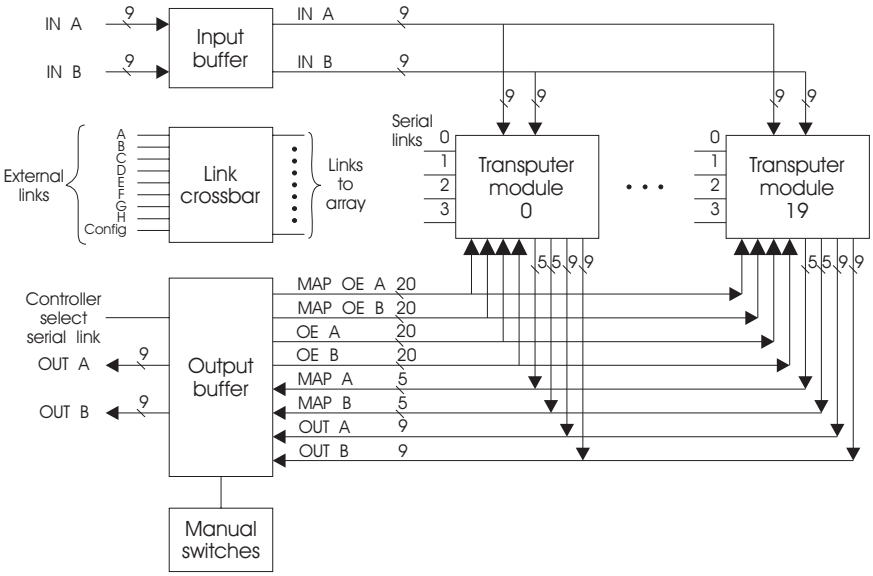

Figure 2. Block diagram of the transputer array.

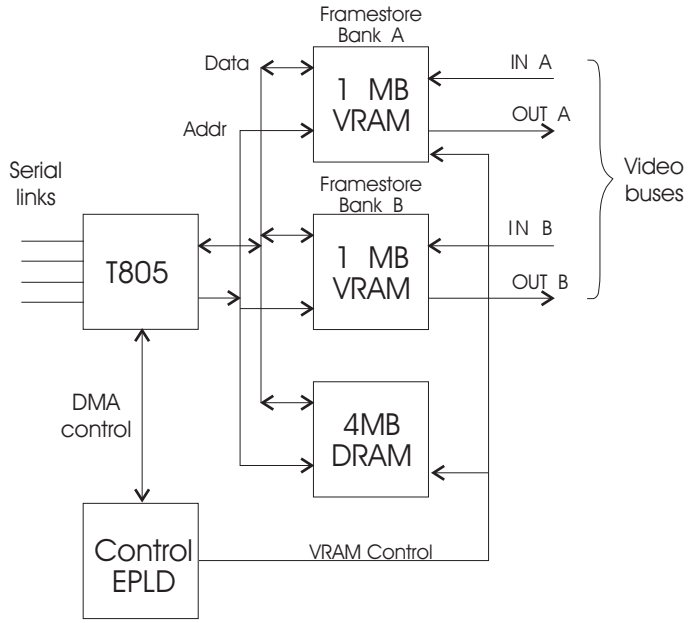

Figure 3. Block dagram of one of the transputer modules.

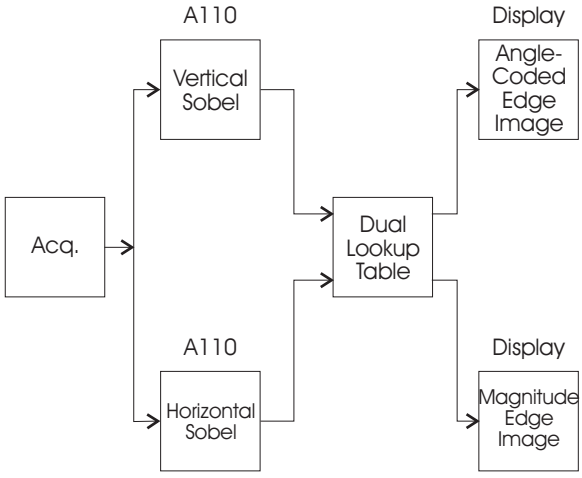

Figure 4. Block diagram of the Sobel application.

Figure 5. Output from the PBN image segmentation algorithm, showing the segmentation of a reconnaissance image into urban and non-urban regions.

Figure 6. Output from the PBN algorithm, identifying the driveable region of an image from a forward-looking camera on a vehicle.

Figure 7. Results of the Probabilistic Relaxation algorithm when applied to the image in figure 5. The ridge features (roads) have been detected in the image.

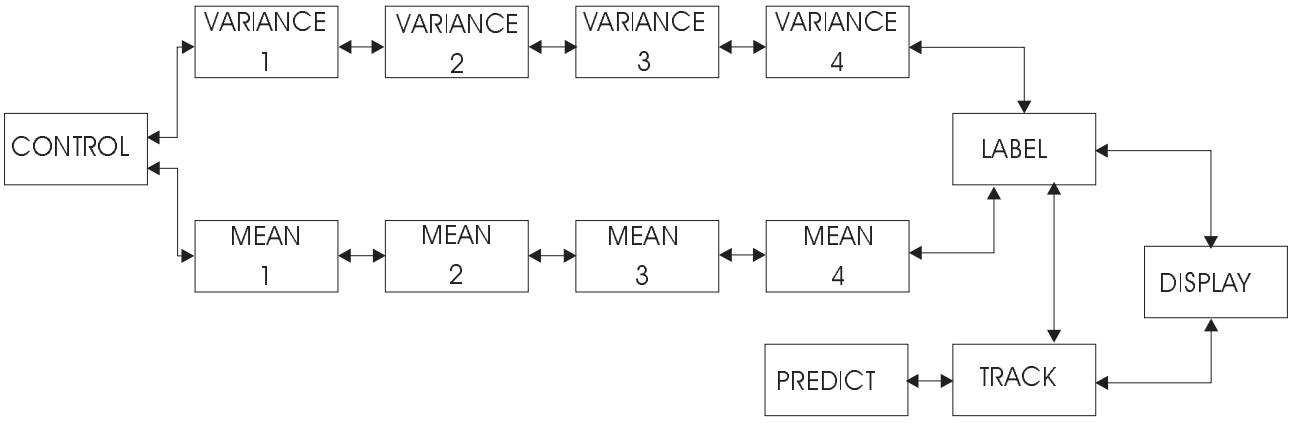

Figure 8. Multi-transputer configuration for target detection and tracking. In this prototype version, video is distributed to the mean and variance processors using the video buses. Results are then collected and passed to the labeller using transputer links.