

**Tradeoffs Between Quality of Results  
and Resource Consumption in a  
Recognition System**

**Michael D. DeVore, Roger D. Chamberlain,  
George L. Engel, Joseph A. O'Sullivan,  
and Mark A. Franklin**

M.D. DeVore, R.D. Chamberlain, G.L. Engel, J.A. O'Sullivan, and M.A. Franklin, "Tradeoffs Between Quality of Results and Resource Consumption in a Recognition System," in *Proc. of the IEEE International Conference on Application-Specific Systems, Architectures and Processors*, July 2002, pp. 391-402.

Washington University  
and  
Southern Illinois University Edwardsville

# Tradeoffs Between Quality of Results and Resource Consumption in a Recognition System

Michael D. DeVore\*, Roger D. Chamberlain\*, George L. Engel†, Joseph A. O’Sullivan\*, and Mark A. Franklin\*

\*Department of Electrical Engineering

Washington University, St. Louis, Missouri

†Department of Electrical and Computer Engineering

Southern Illinois University Edwardsville, Edwardsville, Illinois

## Abstract

*The implementation of computational systems to perform challenging operations often involves balancing the performance specification, system throughput, and available system resources. For problems of automatic target recognition (ATR), these three quantities of interest are the probability of classification error, the rate at which regions of interest are processed, and the capabilities of the underlying hardware (which is a function of the available computational resources and available power). An understanding of the inter-relationships between these factors can be an aid in making informed choices while exploring competing design possibilities. Combining characterizations of ATR performance, which yield probability of classification error as a function of target model complexity, with analytical models of computational performance, which yield throughput as a function of target model complexity and available resources, we can form a set of parametric curves which relate the quality of the results to the resources consumed.*

## 1. Introduction

In many applications there are tradeoffs between available computational resources and the quality of the results output from the system. This is true for many gaming applications, where the realism of the presented scenes varies widely from one computational platform to another. In distributed games, it is common practice for the system to dynamically adapt the application to take maximum advantage of whatever communication bandwidth is available, lowering the frame rate and information content per frame for slower communication links and raising the frame rate and information content per frame for faster links. That is, the quality of results, in terms of the accuracy of rendered scenes, is dependent upon the available resources and frame throughput requirements. Recognition applications also are faced with these kinds of tradeoffs.

In addition, power consumption is no longer a concern only for battery-powered or portable systems. In his keynote address at the 2001 ISCA Conf. on Parallel and Distributed Computing Systems, David Cohn, head of IBM’s Austin Research Laboratory,

---

<sup>1</sup>This research is supported in part by the US Army Research Office grant DAAH04-95-1-0494, the Office of Naval Research grant N00014-98-1-06-06, the Boeing Foundation, and the DARPA VLSI Photonics Program under grant DAAL01-98-C-0074.

made the point that they consider power consumption to be a primary design issue for high performance systems as well. The power density of modern chips is such that new designs are limited as much or more by power constraints than by area constraints.

An important issue in the choice of which computing platform is suitable for an application is relating the following three quantities: quality of the results, throughput of the system, and available resources. We illustrate a general approach to understanding these relationships in applications where the result is built up incrementally through successive computations. The application of interest is Automatic Target Recognition (ATR) using Synthetic Aperture Radar (SAR) images. The general approach may be applied to applications which, like ATR, yield the result of a function optimization, and the quality of result is dependent upon the fraction of the solution space which has been searched. It may also be applicable to variable-rate encoding of data in which the quality of the resulting representation directly depends upon the resources expended in producing it.

In an ATR system, the quality of results, throughput, and resources translate respectively to the probability of correct classification delivered by the system, the rate at which regions of interest are processed, and the capabilities of the implementation hardware (quantity, speed, and power consumption of the processors, memory, and interconnection network). The classification error rate delivered by an ATR system is directly related to the sophistication of the target models employed. More highly refined target models may be expected to yield lower error rates but generally have larger processing and data storage requirements. These larger requirements translate into slower operation and/or a need for greater computational capability. Knowledge of how the classification error rate depends on target model sophistication can be combined with analytic models for computation hardware, power consumption, and recognition algorithms to determine the relationship between error rate, system throughput, and resource consumption. A quantitative understanding of the tradeoffs involved can be a significant help, enabling system designers to make informed choices when specifying, implementing, and using these systems.

In general, the recognition accuracy can be represented as a function  $A(\alpha; \beta)$ , where  $\alpha$  indexes the fraction of the solution space which has been explored, and  $\beta$  is a parameter vector characterizing the recognition problem being solved (target classes, search methodology, signal model, etc.). Larger values of  $\alpha$  indicate that more of the solution space has been examined, and the accuracy of the solution can be expected to increase. The time required to render a classification for an object in an image can be represented as a function  $T(\alpha; \beta, \gamma)$ , where  $\gamma$  is a parameter vector characterizing the computational hardware (number and speed of processors, network bandwidth, etc.). Larger values of  $\alpha$  imply that more effort has been expended and, for fixed  $\beta$  and  $\gamma$ , imply longer classification times. Finally, the energy consumed while classifying an image can be represented as a function  $E(\alpha; \beta, \gamma)$  which is also increasing in  $\alpha$ . The average power consumed by the recognition system,  $P(\alpha; \beta, \gamma) = E(\alpha; \beta, \gamma)/T(\alpha; \beta, \gamma)$ , is not necessarily an increasing function of  $\alpha$ .

In solving a given recognition problem,  $\alpha$  may not be allowed to reach its maximum value of 1 if time and resource constraints (which may vary dynamically) do not permit the entire solution space to be searched before a decision must be rendered. In these cases, the functions  $A(\alpha; \beta)$ ,  $T(\alpha; \beta, \gamma)$ , and  $P(\alpha; \beta, \gamma)$  trace a parametric curve in three-space, and their relationships are of key importance. For a given problem  $\beta$  and point in the design space  $\gamma$ , they can be used to determine the elapsed time required to achieve a given correct classification percentage, the power consumption required to maintain a given throughput, and the correct classification percentage that will result from a given power budget. If the

design parameters in  $\gamma$  are allowed to vary within some allowable region  $\Gamma$  of the design space, the design space can be searched for the point which provides the most desirable relationship between  $A$ ,  $T$ , and  $P$ .

If the functions  $A(\alpha; \beta)$ ,  $T(\alpha; \beta, \gamma)$ , and  $P(\alpha; \beta, \gamma)$  can be approximated early in the system design phase, these ideas can be used to guide design and implementation decisions. Recognition accuracy can be approximated on the basis of empirical studies, and expressions for processing time and power can be formulated on the basis of performance models for the underlying hardware technology. We have previously developed analytic models for ATR from SAR imagery which relate recognition accuracy to the classification time through the number and speed of processors [6] and to the bandwidth of the interconnection network [5]. In this paper, we extend these results to directly include power consumption as a first-level consideration, to allow a variable amount of processor local memory, and to extend the experimental results from a four- to a ten-class case.

Section 2 provides a high-level overview of the ATR problem that is used in this analysis. Section 3 describes the dependence of processing time and power consumption on the hardware architecture based on parametric models for the hardware and ATR algorithm. In Section 4 we present quantitative examples of the use of this methodology in exploring the design space of ATR systems. Section 5 summarizes our results and concludes the paper.

## 2. Application description

In many automatic target recognition (ATR) problems, an image of a scene is collected with a sensor platform and regions of interest within the image are located with target detection algorithms. These regions are extracted to form sub-images, often called target chips, which are then used as input to ATR systems. These systems then report the most likely explanation for each chip in terms of the class, sub-class, pose, and articulation of the target represented in the chip.

ATR can be implemented as a search over a database of target descriptions, often called templates, which characterize the statistical properties of patches on the target surface when imaged through the sensor platform [7]. A template can be thought of as the value of a vector-valued function defined over a multi-dimensional parameter space which includes target class, sub-class, pose, state of articulation, etc. A search is conducted for the template parameters which maximize the likelihood of the observed chip.

The collection of templates for all ranges of poses and articulations for the target classes of interest constitute a database through which the ATR system must search. Because the possible combinations of pose and articulation states are uncountably infinite, the database is quantized and an efficient representation of the data it holds is desirable. The portion of the database within each target class can be arranged hierarchically, with each successively lower level representing a finer quantization of the parameter space [4]. Such an arrangement supports recognition performed by searching over coarsely quantized parameters at high levels of the hierarchy and then gradually descending to finer quantization levels. At any given time, the classification decision represents the most likely object class found up to that time. In this way, the accuracy of a classification depends upon the extent of the model database searched prior to making a decision. Decisions can be made incrementally, with an initial classification successively refined as the search proceeds.

For such systems, it is desirable to organize the recognition algorithm in such a way that

the probability of classification error decreases rapidly at the beginning of the search. Let  $i$  denote the number of templates per target class for which likelihood values are computed. Through experimentation, we can determine samples of the probability of classification error,  $P_e(i)$ , as a function of the extent of the search. In a practical implementation it is also desirable that the probability of error decrease rapidly as system resources, such as CPU cycles and network bandwidth, are consumed. Resource consumption can be predicted from models of the computation hardware and will be dependent in part upon application defined quantities such as the average number of floating point values in a typical template and the number of bits involved in the communication of those floating point values.

For the ATR problem addressed in this paper, we consider classifying into one of ten possible classes an unknown military vehicle in a  $128 \times 128$  pixel, complex-valued SAR image from the Moving and Stationary Target Acquisition and Recognition (MSTAR) program conducted at Wright Laboratory under DARPA funding. While the azimuth angle is unknown, the vehicle is assumed to be on level ground, unarticulated, and imaged from a known depression angle. The dimensionality of the search space increases with each additional pose parameter that is not assumed to be known. The location of the vehicle is assumed known to within 5 feet, which corresponds to  $\pm 5$  pixels given the 1 foot resolution of the imagery. Templates for each of the ten possible classes were estimated from images of the corresponding vehicles, sampled over  $360^\circ$  of target azimuth angle. Data on classification accuracy as a function of the number of templates considered were determined empirically from test imagery that was independent of that used for template estimation.

### 3. Architecture model

The architecture model follows the development in [6] in which a network of  $N_P$  processors share a template database and a source of SAR image chips to be classified. Each processor has its own local memory of a size sufficient to store all pixels of an image chip and all templates for a single target class at some level  $N_L$  in the quantization hierarchy. The relationship between  $N_L$  and the memory size in bytes is given in Figure 1 for the ATR problem described in the previous section. The hierarchical relationship between templates can be exploited to allow the efficient computation of some templates from only incremental data from the database given that a template from a higher level of the hierarchy is already in processor local memory. Beyond level  $N_L$  in the hierarchy, this relationship cannot be exploited for all templates and some will need to be communicated in their entirety.

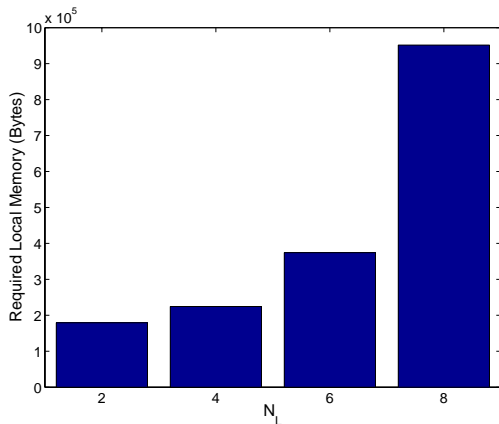
Our interest lies in the average throughput of the classification system and the average power consumed. We do not focus on the latency nor the energy consumed in processing any one SAR chip. To this end, we assume that incoming images to be classified arrive at a rate sufficient to allow all processors to stay utilized and suppose that a scheduling mechanism exists to coordinate processor activity. Because of the sequential nature of the operations, the elapsed time and average power required to classify an image chip includes distributing the chip to all processors, retrieving templates from the database, fetching values from local memory, and performing the relevant computations. We model these as interlaced but nonoverlapping stages of computation, so both the total computation time and power can be expressed as a sum over each stage. If portions of each can overlap in a given implementation, then elapsed time can be taken to represent effective times, equal to the total time multiplied by some nonoverlap percentage.

The architecture model and algorithm properties are used to derive the number of pro-

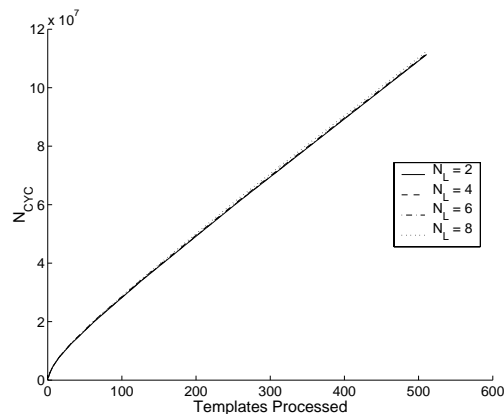
cessor cycles used in computation,  $N_{CYC}(i)$ , number of local memory accesses used in computation  $N_{MEM}(i)$ , and number of bits communicated from the database,  $N_{BITS}(i)$ . These are functions of the number of templates processed per target class prior to rendering a classification. The derivation of these quantities is given in [6] and will not be repeated here but are plotted in Figures 2 through 4. Each of these quantities can be related to a power consumption within the system, and we correspondingly decompose power into that supporting algorithm computation, local memory, and communication. These quantities depend upon the scope of the ATR problem such as the number of target classes considered and number of pose variables which are unknown.

### 3.1. Computation

The total amount of time spent in computation is proportional to the number of clock cycles  $N_{CYC}(i)$  required by the algorithm and inversely proportional to the number of processors  $N_P$  and the processor frequency  $f_P$ . The number of clock cycles required is linear in the number of cycles per instruction (CPI) of the processor and is a slowly varying function of  $N_L$  as shown in Figure 2.



**Figure 1. Local memory as a function of  $N_L$ .**



**Figure 2.  $N_{CYC}$  as a function of  $N_L$  assuming a CPI of 0.5.**

For computation, the classic equation for dynamic power consumption in VLSI circuits is  $P_{COMP} = \alpha C f_P V_P^2$ , where  $\alpha$  indicates the number of signals that transition per clock cycle,  $C$  is the average capacitive load on each gate output,  $f_P$  is the processor clock frequency, and  $V_P$  is the processor supply voltage.

If  $f_P$  is varied,  $V_P$  can be varied also, referred to as dynamic frequency and voltage scaling. The form of this relationship varies with implementation technology. As an example, Figure 5 plots the frequency of a ring oscillator as a function of supply voltage in the AMI 0.5  $\mu\text{m}$  process. We model this as a linear relationship,  $V_P = k_1 f_P + V_0$ , with hard limits on the range of  $V_P$ . This model is shown as the solid line on the figure. The nonlinear relationship with power dissipation is illustrated in Figure 6, which plots the power dissipation in the ring oscillator over the same range of supply voltage. In this example, we are analyzing a 33 stage ring oscillator designed using the Mentor ADK publicly available cell library [11].

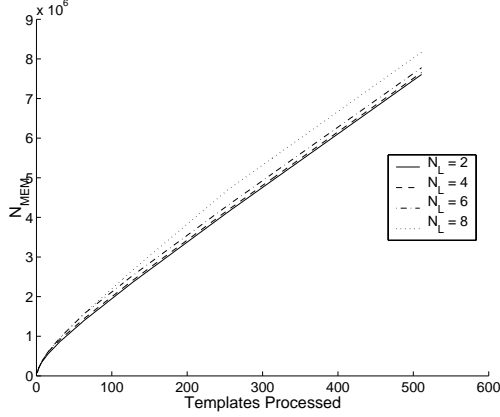


Figure 3.  $N_{MEM}$  as a function of  $N_L$ .

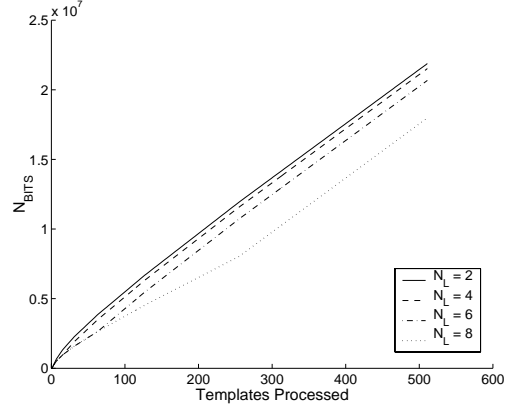


Figure 4.  $N_{BITS}$  as a function of  $N_L$ .

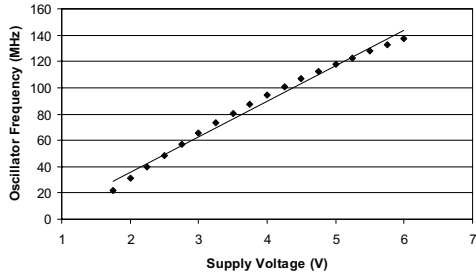


Figure 5. Ring oscillator frequency as function of supply voltage.

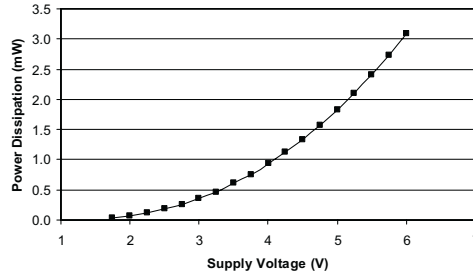


Figure 6. Power dissipation of ring oscillator as function of supply voltage.

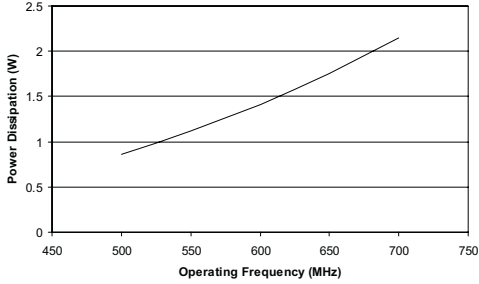
The Crusoe processor from Transmeta is designed for frequency and voltage scaling. Using published data points for the TM5600 [9], we have calibrated the above processor power model and present the predicted power versus operating frequency in Figure 7. Over the frequency range of 500 to 700 MHz, the operating voltage varies from 1.2 to 1.6 V. Burd [1, 2, 3] reports on an experimental frequency and voltage scaled processor designed for low-end applications, called the Dynamic Voltage Scaled (DVS) processor. The power versus frequency curve for this processor is given in Figure 8.

The number of clock cycles during computation is equal to the clock frequency times the elapsed time of computation. The average power consumed by computation is then

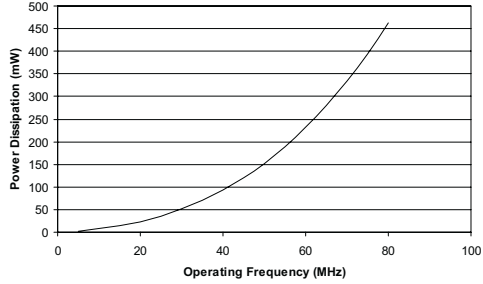
$$P_{COMP} = \alpha C (k_1 f_P + V_0)^2 \frac{N_{CYC}(i)}{T_{CHIP}(i)}. \quad (1)$$

### 3.2. Memory access

The amount of time spent in retrieving values from local memory is proportional to the number of bits accessed from memory,  $N_{MEM}(i)$ , and is inversely proportional to the number of processors and the clock frequency. The quantity  $N_{MEM}(i)$  is a function of  $N_L$  as shown in Figure 3 and accounts for variation in memory usage if templates are



**Figure 7. Crusoe power dissipation as a function of operating frequency.**



**Figure 8. DVS power dissipation as a function of operating frequency.**

communicated in their entirety or incrementally based on their hierarchical relationship.

The above processor power models are inclusive of on-chip cache, but do not include the power associated with the main memory at each node. This memory, typically DRAM technology, can be modeled as having two components to its power consumption [8]. The first is linear in the number of bits stored, representing the idle power required to refresh the dynamic storage cells. The second is linear in the number of bits accessed, representing the power required for driving the sense amplifiers and (more importantly) delivering the signals off-chip, across the bus, to the processor. The resulting model is, therefore,

$$P_{MEM} = k_2 \cdot g(N_L) \cdot N_P + k_3 \cdot \frac{N_{MEM}(i)}{T_{CHIP}(i)}, \quad (2)$$

where  $g(N_L)$  represents the size of the local memory needed to store the image chip and all templates for a single target class at level  $N_L$  as shown in Figure 1. In the results that follow, we have calibrated the memory model using data from Micron DDR RAM [8].

### 3.3. Communication

The total amount of time spent in communication includes the amount of time spent distributing an incoming image to all processors and the amount spent retrieving templates from the database. Assuming the architecture does not support generalized data broadcasting, the image chip to be classified must be independently communicated to each processor, and with a switched interconnection network this can be accomplished in  $\lceil \log_2(N_P + 1) \rceil$  steps of  $S_C$  bits over a network supporting  $BW$  bits per second. The number of bits  $N_{BITS}(i)$  to communicate templates from the database is a function of  $N_L$  as shown in Figure 4 and accounts for variation if templates are communicated incrementally or in their entirety.

The power required for communication can be divided into two components. The first is the power required to actually deliver the bits across the communications link and is inherently a linear function  $k_4 BW$  of the data rate. The second is the power required by the processor to manage the communications task (this includes data copies, protocol processing, etc.). This can be expressed as a linear function of the number of bits communicated, assuming the number of instructions and memory accesses required is proportional to the data volume. This component of communication power includes both processor and memory access power consumption. The power consumed in communication is not constant

over the classification period. The average power over the classification period is

$$P_{COMM} = k_4 BW \frac{T_{COMM}(i)}{T_{CHIP}(i)} + \left( \alpha C (k_1 f_P + V_0)^2 \cdot \text{CIC} \cdot \text{CPI} + k_3 \right) \frac{N_P S_C + N_{BITS}(i)}{T_{CHIP}(i)}, \quad (3)$$

where  $T_{COMM}(i)$  is the total time elapsed in communication as a function of the number of templates considered and CIC is the communication instruction count per bit.

As an example, 100 Mb/s Ethernet systems have a voltage swing of 1 V. Assuming a cable capacitance of 20 pF, that gives a value for  $k_4$  of 20 pJ/bit. The processing component is the sum of the compute power expression in Section 3.1 and memory access power expression in Section 3.2, with number of cycles and bits depending on the quantity communicated. In [10] Wolf and Franklin measure the computational complexity of a number of header-processing applications in *Commbench*, a telecommunications benchmark set. The mean value for the computational complexity is reported as 6.1 instructions per byte.

### 3.4. Aggregate system

With these assumptions, the architecture model yields a chip processing time in seconds of

$$T_{CHIP}(i) = \frac{S_C}{BW} [\log_2(N_P + 1)] + \frac{N_{BITS}(i; N_L)}{BW} + \frac{N_{MEM}(i; N_L) T_{MEM}}{N_P} + \frac{N_{CYC}(i; N_L, \text{CPI})}{N_P f_P} \quad (4)$$

where  $f_P$  is the clock frequency, BW is the interconnection bit rate, and  $T_{MEM}$  is the memory access time per bit. Given the predictability of ATR processing we assume that a 64 bit value can be read on each clock cycle with memory prefetch, and  $T_{MEM} = 64/f_P$ .

Finally, for any system there is a baseline power dissipation that is not a function of how the system is operated. This value simply provides an offset to the above power consumption model and will not be considered further. The average power consumed by the system is thus

$$P_{TOT} = \alpha C (k_1 f_P + V_0)^2 \frac{N_{CYC}(i)}{T_{CHIP}(i)} + k_2 \cdot g(N_L) \cdot N_P + k_3 \cdot \frac{N_{MEM}(i)}{T_{CHIP}} + k_4 BW \frac{T_{COMM}(i)}{T_{CHIP}(i)} + \left( \alpha C (k_1 f_P + V_0)^2 \cdot \text{CIC} \cdot \text{CPI} + k_3 \right) \frac{N_P S_C + N_{BITS}(i)}{T_{CHIP}(i)} \quad (5)$$

The architecture parameters are included in the summary provided in Table 1.

## 4. Model use

Given the model development above, we now have the ability to relate the quality of results (i.e., percentage error in the classification of targets) to the available computational resources (i.e., power, etc.) for our application of interest. The model defines the relationship between these quantities, yet there are many choices left to us with respect to how we might take advantage of this relationship. In what follows, we illustrate a number of potential uses and discuss options for other uses.

In our first use of the model, a number of design parameters are fixed, and we are exploring the remaining degrees of freedom in the design problem. Here, we are assuming the use of  $N_P = 4$  Crusoe processors, each operating at  $f_P = 500$  MHz (with an estimated

**Table 1. Model variables (units are stated where relevant).**

Variable	Meaning	Units
$N_P$	number of processors	
$f_P$	processor frequency	Hz
$V_P$	processor supply voltage	V
CPI	mean clocks per instruction, ignoring memory effects	
CIC	communication instruction count	inst./bit
$T_{MEM}$	mean memory access time per bit	s
$N_L$	largest quantization level fitting in local memory	
BW	bandwidth of the interconnection network	bits/s
$S_C$	size of a SAR image chip	bits
$N_{CYC}(i)$	number of clock cycles to compute through template $i$	
$N_{MEM}(i)$	number of memory bits accessed through template $i$	bits
$N_{BITS}(i)$	number of bits from database through template $i$	bits
$P_{COMP}$	computational power	W
$P_{MEM}$	memory power	W
$P_{COMM}$	communications power	W

CPI of 0.5), sufficient local memory to store  $N_L = 6$  hierarchical levels in the template database, and a relatively slow ( $BW = 10$  Mb/s) communications infrastructure. Figure 9 shows four relationships for this set of design constraints. In the upper-left plot, the percent classification error is plotted versus the time spent on classifying each chip. Here we see the quantified relationship between quality of results and throughput of the system. The upper-right plot extends this to include quality of results versus energy input (per classification). In both plots, we see the improvement in quality (decrease in error probability) as the time available or energy input is increased.

What varies along the two curves in the top plots (not shown explicitly) is the number of templates that are searched in the database ( $i$  in the model development). This represents the extent of the database that is searched, and is shown explicitly in the bottom two plots of the figure. The bottom-left plot shows the total power consumption of the system as a function of the extent of the search. In this plot, the power clearly shows a minimum at low search extents, which turns out to be quite near the knee of the classification error curve. That points to the existence of a good design point with minimal power consumption and very good quality of results.

The power is decomposed in the bottom-right plot to show the relative contributions of computation, memory, and communications. At low search extents, the communications power is dominating (due to the need to distribute the chips). At higher search extents, the memory power is dominating, suggesting that a smaller memory might be a design worthy of consideration.

Figure 10 repeats the above curves with one change, the local memory has been decreased to only store  $N_L = 2$  hierarchical levels in the template database. Since we are not communications limited in throughput, the upper-left plot is unaltered. Note, however, the significant change in the remaining three plots. In the upper-right, we see that the energy required for a similar quality of classification has decreased by better than half. In the lower-left, we see that the minimum power point has moved to the right (implying a greater quality of result) and is much flatter near the minimum (implying less sensitivity

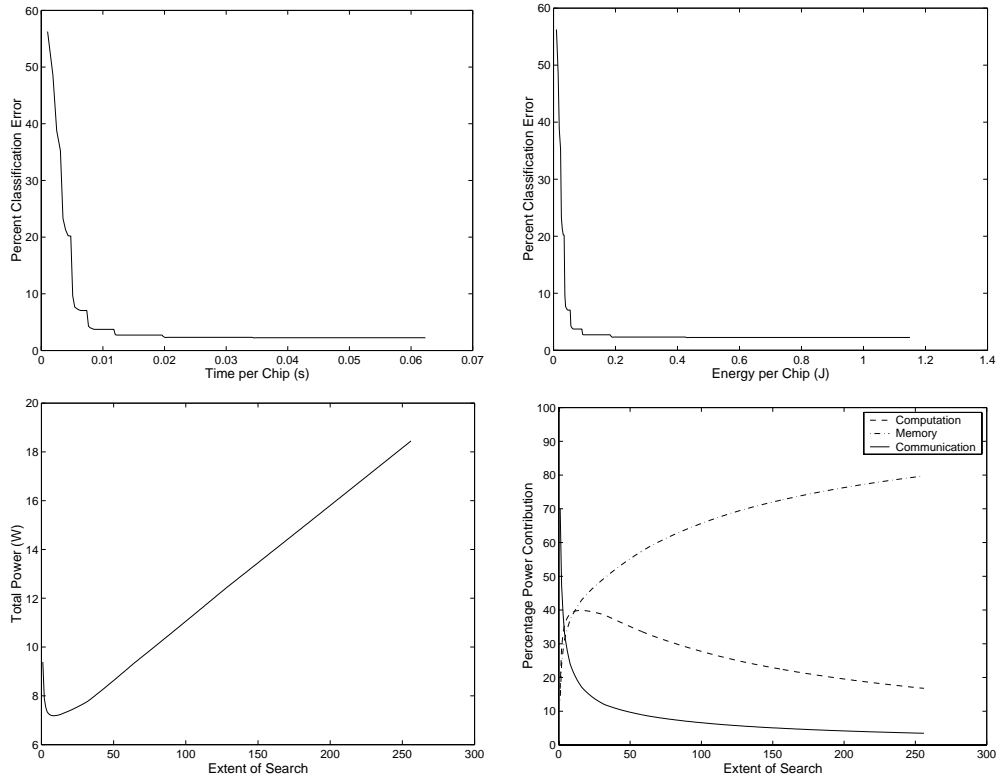


Figure 9. Model results for  $N_L = 6$ .

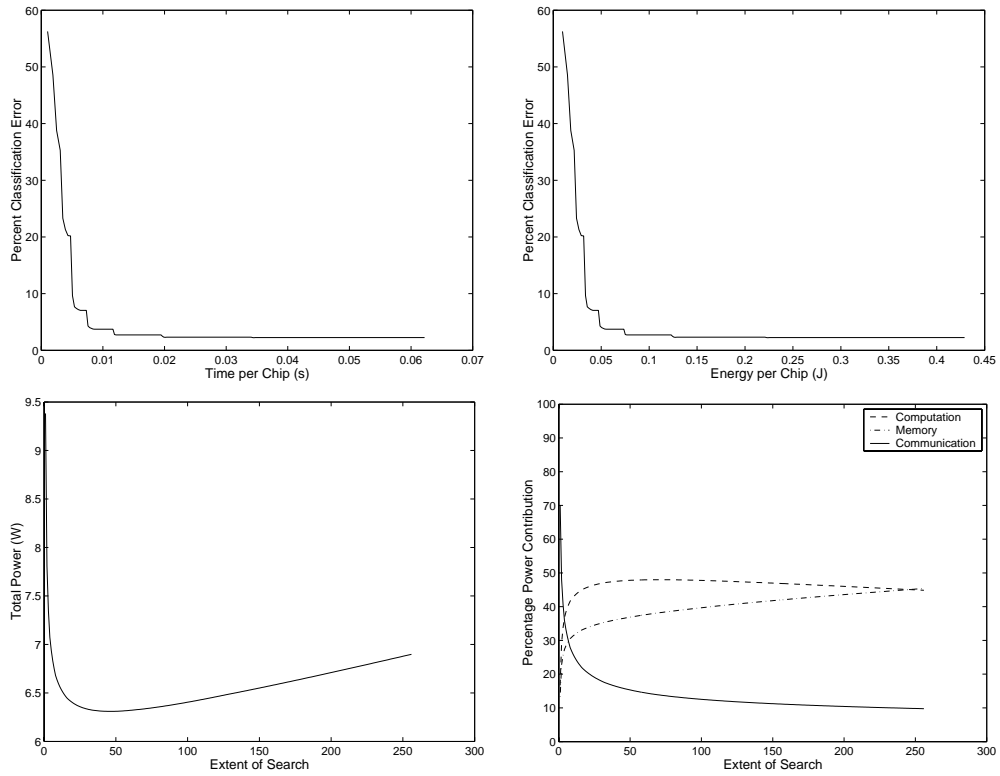
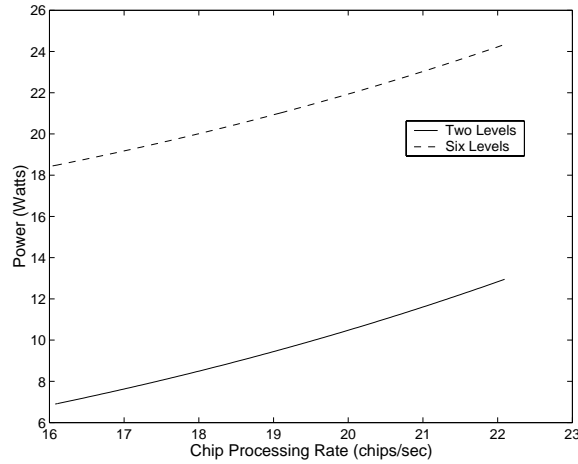


Figure 10. Model results for  $N_L = 2$ .



**Figure 11. Power dissipation versus throughput while scaling processor frequency and voltage.**

to minor perturbations in the operating characteristics). In the lower-right, we see a more even match between the power consumption of the computation versus the memory, in effect giving us a more power-balanced system.

To illustrate variation across another design dimension, Figure 11 plots the total power dissipation versus throughput rate as we vary the operating frequency of the processors from 500 MHz to 700 MHz. The processor voltage is scaled with the frequency from 1.2 V to 1.6 V. Curves are shown at a fixed probability of classification error ( $P_e = 2.3\%$ ) and for both memory sizes described above ( $N_L = 6$  and  $N_L = 2$ ). This type of plot enables the designer to understand the application-level implications of scaling frequency and voltage. It also illustrates the significant power savings realized by the smaller (per processor) memory.

In addition to the uses illustrated above, there are several other ways in which the model can potentially be used. The intent of the modeling effort is to provide an evaluation tool than enables a more informed exploration of the potential design space than would otherwise be available. To this end, the model enables quantitative comparisons of a number of design tradeoffs, including number and performance of processors, memory quantity, speed of the interconnection network, and quality of the overall results.

These tradeoffs need not be limited to design-time choices. The model can also provide guidance in the configuration of an adaptive system, quantifying the inherent tradeoffs between quality of results, throughput, and power consumption. Consider the case of an airborne system that must adapt to varying environmental conditions. In a rural environment, the throughput rate might be relatively low, reflecting a lower rate of interesting objects on the ground that require classification. Under these conditions, the system might dedicate more resources to each classification, giving a lower probability of classification error. The circumstances in an urban environment are potentially significantly different, with a much larger set of objects on the ground requiring classification. The higher throughput rate necessary will result in a higher classification error rate. The above description essentially involves the choice of an operating point on the upper-left curve of, say, Figures 9 or 10. While the shape of the curve itself is determined at design time, the point on the curve at which the system operates can be a run time decision, enabling the system to

adapt to changing external conditions. Knowledge of how the system will operate over a range of parameters enables both better initial design choices and more intelligent usage over its entire lifecycle.

## 5. Summary and conclusions

This paper has presented a model that relates quality of results to resource consumption for a specific embedded system application. More generally, we relate the three quantities: probability of classification error, system throughput, and available resources. The quantification of available resources includes computation (number, speed, and power of processors), memory (quantity and power), and communications (data rate and power of interconnection network). This type of modeling enables the system designer to quantitatively understand the implications of design choices in terms that are meaningful at the application level.

The use of the model is illustrated through a simple exploration of the design space along two dimensions (memory per processor and processor operating frequency). A more general use of the model would be to guide a design optimization where one is given a set of requirements and the minimum power configuration that meets those requirements is automatically derived. The solution of this type of optimization problem represents our current efforts and will be reported in future work.

## References

- [1] Thomas D. Burd. *Energy-Efficient Processor System Design*. PhD thesis, University of California at Berkeley, 2001.
- [2] Thomas D. Burd and Robert W. Brodersen. Design issues for dynamic voltage scaling. In *Proc of Int'l Symp. on Low Power Electronics and Design*, pages 9–14, 2000.
- [3] Thomas D. Burd, Trevor Pering, Anthony Stratakos, and Robert W. Brodersen. A dynamic voltage scaled microprocessor system. In *Proc. of Int'l Solid State Circuits Conf.*, February 2000.
- [4] Michael D. DeVore, Joseph A. O'Sullivan, Sushil Anand, and Natalia A. Schmid. Probabilistic approach to model extraction from training data. In Edmund G. Zelnio, editor, *Algorithms for Synthetic Aperture Radar Imagery VIII, Proc. of SPIE*, 2001.
- [5] Michael D. DeVore, Joseph A. O'Sullivan, Roger D. Chamberlain, and Mark A. Franklin. Dependence of recognition accuracy on available network bandwidth. In *Proceedings of the Fifth Annual High Performance Embedded Computing Workshop*. Lincoln Laboratory, Massachusetts Institute of Technology, 2001.
- [6] Michael D. DeVore, Joseph A. O'Sullivan, Roger D. Chamberlain, and Mark A. Franklin. Relationships between computational system performance and recognition system performance. In Firooz A. Sadjadi, editor, *Automatic Target Recognition XI, Proc. of SPIE*, volume 4379, 2001.
- [7] Joseph A. O'Sullivan, Michael D. DeVore, Vikas Kedia, and Michael I. Miller. Automatic target recognition performance for SAR imagery using a conditionally Gaussian model. *IEEE Transactions on Aerospace and Electronic Systems*, 37(1):91–108, January 2001.
- [8] Micron Technology. Calculating Memory System Power for DDR. Technical Report TN-43-03, Micron Technology, Inc., May 2001.
- [9] Transmeta Corporation. *Crusoe Processor Model TM5600*, August 2000.
- [10] Tilman Wolf and Mark Franklin. CommBench - A telecommunications benchmark for network processors. In *Proc of Int'l Symp. on Performance Analysis of Systems and Software*, pages 154–162, April 2000.
- [11] David M. Zar. Standard Cell Library Design with IC Station. In *Proc. of Mentor Graphics Users' Group International Conference*, 1999.