Confidence-driven Memory Architecture for Real-Time Vision System

Yoshimoto, Hiromasa
Department of Intelligent Systems, Kyushu University

Arita, Daisaku
Department of Intelligent Systems, Kyushu University

Taniguchi, Rin-ichiro
Department of Intelligent Systems, Kyushu University

Confidence-driven Memory architecture for Real-Time Vision System

Hiromasa Yoshimoto, Daisaku Arita and Rin-ichiro Taniguchi

Department of Intelligent Systems, Kyushu University,
6-1 Kasuga-Koen, Kasuga, Fukuoka 816-8580 Japan.

1 Introduction

In this paper, we propose a Confidence-driven architecture to control trade-off between precision and latency of computer vision systems dynamically. It is important for vision systems to achieve data stream interpretation accurately without latency. However, there is a trade-off: a long computation time is required to get accurate information from the input-images; on the other hand, fast computation is desired to get information as soon as possible. The trade-off is influenced by many factors: whether a user assigns high priority to higher precision or to lower latency; how many computational resources such as computation time and bandwidth of the network are available. We have to make difficult trade-off between the precision and the latency in dynamically so that the quality of the system's output keeps the best one.

In many vision-based applications, a lot of implicit trade-offs are considered. To clarify the description, an example of vision-based applications is given, which uses a template matching method to find a target region. Using the template matching, as the resolution of the template images increases, the precision of the search results becomes higher but the computation time becomes longer. We have to make trade-off to determine the resolution of the templates. In another aspect, real-time vision-based applications handle time varying variables, such as the position of a target, and reduction of the redundancies can lead to decrease the computation time. An estimator such as Kalman filter is often efficient to increases the throughput. The estimator provides results faster than the template matching but the precision becomes lower. Trade-off is to determine the ratio of the use of the estimator to that of the template matching.

2 Confidence

It is important that the template matching and the estimator have the same characteristic, that is, as the time increases the accuracy of the output becomes higher monotonically. These monotonic characteristic is a basis of imprecise computation model[1]. If the relation between allowable computation time and the accuracy are known, we can control the trade-off according to the relation. Since, usually, the relation is affected by several factors and is not known precisely in advance, instead we hypothesize a simple relation such as linearity between the time and the accuracy. It is important that there is no generic way to calculate absolute accuracies of vision algorithm. However according to characteristics of the algorithms, we can estimate relative accuracies heuristically. Therefore we have introduced a term "confidence", ranging from 0 to 1. The confidence is an abstraction of the accuracy. The higher the confidence is, the more accurate the estimation is.

We formalize various trade-off problems as a simple producer-consumer problem. To clarify the description, we have introduced several functions. Target information is denoted as a time function $x(t)$. $c(t)$ is the confidence of the $x(t)$. In the system, the producer computes a value of $x(t_n)$ from an input stream. The heavier computation method the producer uses, the higher confidence $c(t_n)$ of $x(t_n)$ becomes. The producer provides a history h of the stream, a set of $\{t_n, x(t_n), c(t_n)\}$. $F_x(h,t)$ is an estimator of $x(t)$, which refers to the history h. When the history has not enough information to estimate $x(t)$, the estimation of $F_x(h,t)$ has less confidence. We have also introduced $F_c(h,t)$, which is an estimator of $c(t)$. In general, the confidence is getting lower as the estimator predicts data in more distant future.

The key idea of our confidence-drive memory architecture is that it is not the data but the confidence which the consumer waits for. The data $x(t)$ is estimated by $F_x(h,t)$ in any time, and the consumer waits until the $F_c(h,t)$ becomes high enough. We call this scheme as "Confidence-driven." Using the Confidence-driven scheme, the trade-off between the precision and the latency is converted to the trade-off between the confidence and the latency.

```
write(Memory m, Value x,
      Confidence c, Time t)
{
  add_history(m, x, c, t);
  send_signal(m);
  if ( c > m.required_cofidence )
    return SATISFIED;
  else
    return NOT_SATISFIED;
}
```

(a) Write Operation

```
read(Memory m, Time t,
     Confidence c, Deadline w)
{
  m.required_cofidence = c;
  while( Fc(m,t) < c ) {
   wait_for_a_signal(m, w);
   if ( timeout(w) ) break;
  }
  return { Fx(m,t), Fc(m,t) };
}
```

(b) Read Operation

Fig. 1. Pseudo-codes of the write and read operations

Confidence-driven memory is a shared memory based on the Confidence-driven scheme. Producer and consumer are implemented as threads and can communicate over the the Confidence-driven memory with three operations, read, write, poll, and two user-defined functions $F_x(h,t)$ and $F_c(h,t)$. Fig.1 shows pseudo-codes of a read operation and a write operation. The consumer task executes the read operation and suspends until the confidence to be high enough. When the consumer is suspended, producer thread computes data to make the confidence bigger. Thus each threads synchronized according to the confidence. In general, there is a certain overhead for the threads to share the data completely. However, in the Confidence-driven scheme, each thread shares estimated data without the overhead while the confidence $F_c(h,t)$ is higher than the consumer requires.

3 Evaluation

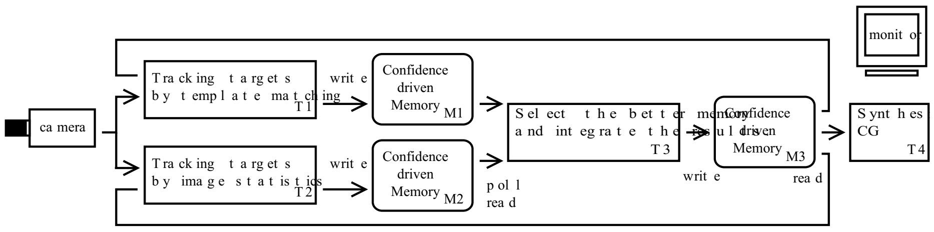

Fig. 2. Application overview

To evaluate the Confidence-driven memory, we have constructed a vision application, which estimates human's hand and face positions and displays its analysis result on the monitor (see Fig.2). The application consists of four threads, T_1, T_2, T_3 and T_4. These threads can communicate over three Confidence-driven memories, M_1, M_2 and M_3. T_1 and T_2 are image processing threads for searching the positions of human's face and hand from an image sequence. T_1 searches for the target position based on a template matching algorithm, which finds a region similar to given templates of a face and a hand. T_1 needs about 240msec/frame under Pentium III 1GHz. On the other hand, T_2 searches the centroids of skin color regions and estimates the most suitable regions as face and hand region. T_2 needs only 18$msec$, but the error would be bigger than T_1, because hand and face regions have similar skin colors. According to the confidence, T_3 selects more suitable memory. When accurate results is required, T_3 selects the results of T_1 more frequently. T_4 reads estimated positions of the target from M_3 and displays them.

A stream of input images are processed by these four threads. The threads communicate one another by lazy transaction over the Confidence-driven mem-

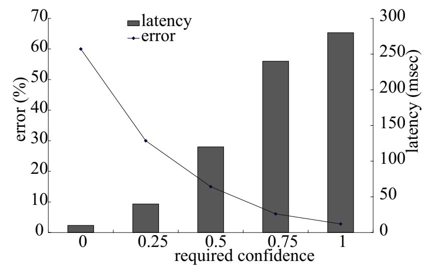

Fig. 3. The latency and the error per the required-confidence

ories. Therefore, we can control the quality of the system output by tuning the required-confidence. Fig.3 shows the relation among the latency, the error and the required-confidence. It shows that when the required-confidence is set to be large the estimation accuracy becomes high but the latency becomes large. On the other hand, when the required-confidence is set to be small the latency becomes small but the accuracy becomes low. Thus, the tradeoff between latency reduction and the accuracy can be smoothly controlled by the required-confidence.

4 Conclusion

In this paper, we have proposed a Confidence-driven Memory architecture, an efficient real-time communication mechanism for distributed vision processing. The architecture has similar features to Imprecise Computation Model and Dynamic Memory[2]. The important point is that it provides a mechanism of synchronization with respect to confidence, which can increase the system execution efficiency in terms of throughput and latency. Because it is a practical technique to use predicated values, we believe, the Confidence-driven Memory architecture can be applied to a number of applications. By using this Confidence-driven Memory architecture, programmers can describe real-time computer vision applications, without taking care of the problem of deadlock and latency of communication.

References

1. J. Liu, W. Shih, K. Lin, R. Bettati and J. Chung. Imprecise Computation, *Proc. IEEE*, Vol.82, No.1, pp.83-94, 1994.
2. T. Matsuyama, S, Hiura, T. Wada, K. Murase and A. Yoshioka. Dynamic Memory: Architecture for Real Time Integration of Visual Perception, Camera Action, and Network Communication, *Proc. of Computer Vision and Pattern Recognition*, pp.728-735, 2000.

This article was processed using the LaTeX macro package with LLNCS style