Optimizing Sensor Network Target Tracking and Localization Algorithms for TMS320C6X

Kamalavasan Srinivasan
Niveditha Sundaram

April 25, 2006

1 Introduction

This project will focus on optimizing two key algorithms from sensor networks domain, namely target tracking and localization for TMS320C6x DSP processor. The optimized algorithms will be profiled using the profiler tools available through *Code Composer Studio* for the target processor. Sensor network applications are becoming commonplace. These nodes are key to many monitoring applications. Surveillance nodes in the field could detect a possible breach of territory by a target that needs to be detected. Localization algorithms are inherent to boundary estimation in sensor networks. Most of the algorithms for target tracking require localization algorithms.

Code optimization is not new for TMS320C6X processor. The chip has been around for over five years. We intend to use the processor for running our target tracking application. Target tracking is a compute intensive task and thus we need to optimize the algorithm to the architecture.

2 Motivation

Target tracking is a common sensor network application that is computationally intensive for processors used in this domain. One of the key components in this algorithm is boundary estimation. This is performed by support vectors drawn along the periphery to estimate the boundary of the target. Our goal is to optimize this section of the code. Target tracking requires real-time guarantees. For providing these guarantees the algorithm must execute in polynomial time. Computationally intensive algorithms need to be optimized in order to meet the assigned deadlines.

The TMS320C6X is a commonly used DSP processor from Texas Instruments. This processor is chosen for our profiling purposes. Profiling tools for processors such as ATmega128L used for ZigBee based sensor nodes are not available. It is reasonable to assume that a DSP processor would be embedded in a sensor node used for military surveillance.

3 Related Work

Code optimization for sensor networks have been performed by many authors [1], [2].

There has been very little work on optimizing target tracking for sensor networks. Our work will have preliminary results based on a DSP processor.

4 Approach

We have been understanding the architecture of TMS320C6x processor. This will help us write optimized code. Embedded programming requires good knowledge of the hardware and software for writing optimized code. Most algorithms can be modularized. Our target tracking algorithm is no exception, we have identified a critical section of the code for boundary estimation. This would be the section we will optimize for the DSP processor. Once this is done we intend to rewrite our Matlab program for target tracking to *C*. This would serve as the baseline version of our project. Subsequent versions of the optimized code would involve hand optimization based on the architecture of the DSP processor.

5 Expected Results

We expect to observe considerable speedup in comparison to the unoptimized version, compiler optimized version and our hand optimized version. Qualitatively, the speedup is expected to improve with our optimized version since we will be performing the techniques discussed in class for rewriting the algorithm. The current Matlab version is unoptimized and exists only for verifying the functionality of target tracking.

6 Task Planning

This section enumerates the various tasks we intend to accomplish in this project

1. Identify critical sections of the Matlab code for target tracking
2. Rewrite the Matlab code to C and profile the code for TMS320C6x. This will serve as our baseline version for comparison.
3. Optimize the C code using various code optimization techniques such as loop unrolling, vectorization, etc.,
4. Profile the optimized code and compare it with the baseline
5. Write a detailed report of the optimization techniques and results

References

[1] Bruce Powell Douglass. *Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

[2] David Gay, Phil Levis, and David Culler. Software design patterns for tinyos. In *LCTES'05: Proceedings of the 2005 ACM SIGPLAN/SIGBED conference on Languages, compilers, and tools for embedded systems*, pages 40–49, New York, NY, USA, 2005. ACM Press.