UAV Video Image Stabilization on the SRC MAP® Processor

William Turri, University of Dayton Research Institute (UDRI)
William.Turri@udri.udayton.edu

David Pointer, SRC Computers, LLC
dpointer@srccomputers.com

Introduction

Compensating for airframe motion in Unmanned Aerial Vehicle (UAV) imagery is crucial to mission success. Image stabilization is traditionally accomplished by wireless video transmission to ground-based computers, which process the image frames before presentation to the UAV operator. However, usable wireless bandwidth is not keeping pace with the amount of data produced by today's high-resolution image sensors, which is forcing designers to move many image processing functions from the ground to on-board the UAV. These functions, which could include target tracking, feature extraction, temporal image compression and others, require stabilized video input.

Our goal is to stabilize 11-megapixel grayscale video at a sustained rate of three frames-per-second (fps). Meeting this computational demand along with airframe size, weight and power (SWaP) requirements is challenging. Published results utilizing the SRC MAP Processor for Synthetic Aperture Radar (SAR) image processing [1], the Tactical Reconnaissance and Counter Concealment Enabled Radar (TRACER) project [2] and airborne image processing in general [3] indicates the SRC-7 MAP Processor provides the performance required by typical airborne image processing applications. In addition, several fielded SRC-7 systems in small to mid-sized UAVs show the MAP Processor delivers this compute capability within each program's SWaP specifications. For example, a single CPU/MAP system requires 165 cubic inches of space, weighs less than 6 pounds and consumes less than 100 watts of power.

UAV Video Image Stabilization

Video image stabilization (Figure 1) for each image frame is computed by selecting "good" features in the previous frame, locating (tracking) these features in the current frame, estimating the tracked feature motion vectors between frames, then removing the estimated motion for each pixel in the current frame.

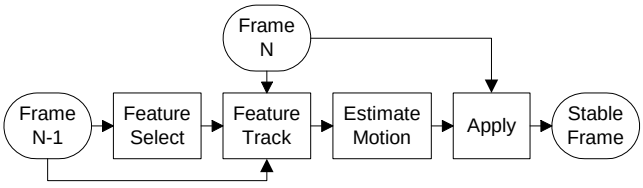

Figure 1: Video Image Stabilization Processing Flow

Appropriate algorithm selection for each of these processing blocks depends upon the nature of the undesired video motion due to airframe flight, random airframe motion and ambient visual conditions. Which algorithms are "best" for UAV video image stabilization is currently the subject of much research. This implementation is derived from one [4] of several papers in this area.

Implementation

From the results presented in [4], this implementation uses the Harris corner detector to select 5x5 pixel templates of features from the previous image frame. Each 5x5 feature template from the previous frame is located in the current frame by utilizing the sum of absolute differences (SAD) template matching technique. The affine transform model provides reasonably accurate motion estimation results for the majority of parameters in the UAV standard six-degrees-of-freedom flight model [4], and so the iterative least squares method is used on the feature motion vectors to obtain estimates for the affine model's scaling, rotation and translation parameters. These affine model parameters are used to remove the estimated motion from each pixel in the current image frame.

These algorithms were initially implemented in Matlab for a developmental proof-of-concept. This Matlab code was then translated into ANSI C programs for the CPU and the MAP Processor.

Application performance profiling results on the CPU naturally lead to the system design shown in Figure 2. Image frames are directly input to the MAP Processor where features are selected and tracked between frames, resulting in a set of image feature coordinates for two frames. The current frame and these coordinates are input to the CPU, where the final motion estimation and image stabilization take place. The data transfer to the CPU is overlapped with the MAP execution, and so does not add to the overall processing time.

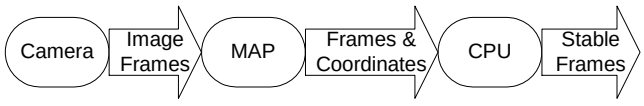

Figure 2: Image Processing Flow

Results

In Table 1, the CPU is a 2.67 GHz quadcore Intel Nehalem system with 12 GBytes of system memory running Fedora Core 10 Linux. The MAP Processor is an SRC-7 Series H MAP with two 150 MHz Altera EP2S180 FPGAs, 64 MBytes of On-Board Memory, and two 512 MByte banks

of Common Memory. The frames are 11 megapixel (4008x2672) 8-bit grayscale images from an Illunis XMV digital camera. 220 features are tracked between each pair of video frames. The camera connects directly into a Camera Link card on the MAP Processor's GPIOX interface. The CPU-only timing measurements assume the use of a high-performance PCI-Express frame grabber card capable of sustained data transfer between the camera and CPU system memory at the full Camera Link data rate of 900 MBytes per second.

Algorithm	CPU (s)	MAP (s)	Speedup	CPU + MAP (s)
Feature Select	5.803	0.072	81x	0.072
Feature Track	72.724	0.987	74x	0.987
Estimate Motion	0.000045	0.000051	0.9x	0.000045
Apply	0.093	0.072	1.3x	0.093
Totals	78.527	1.130	70x	1.151

Table 1: Application Performance

The CPU alone stabilizes an 11 megapixel image frame with 220 tracked features once every 78 seconds. Virtually 100% of that time is spent in executing the feature select and tracking algorithms. The MAP Processor executes the feature select algorithm 81 times faster and the feature tracking algorithm 74 times faster than the CPU for the same image frames and feature count. In other words, the single MAP Processor does the work of over 70 Nehalem CPUs, replacing several conventional servers. Working together as shown in Figure 2, this heterogeneous system achieves an image stabilization frame rate of nearly 1 fps.

Summary

Profile-driven partitioning of the application tasks across the system CPU and the MAP processor achieves a performance level of 70x. The MAP processor could execute all four algorithms internally without a CPU, but the MAP's execution of the motion estimation and final image stabilization algorithms are roughly the same as the CPU's execution time. Moving these functions from the MAP to the CPU freed up MAP resources which were then used to implement additional parallelism in the feature tracking algorithm, which increased its performance on the MAP from an initial 10x to its current 74x over the CPU.

Table 2 summarizes the MAP processor architectural features that significantly contributed to the implementation's performance results. The GPIOX interface enabled sustained delivery of image frames directly into the MAP Processor at full Camera Link data rates. The Common Memory acted as a frame buffer for the incoming frame as well as holding the previous and current frames locally for processing. Streaming allowed concurrent processing within the MAP for a large performance gain over the CPU's serial processing. Lastly, streaming DMA between the Common Memory and the MAP FPGAs and between the MAP and the CPU system memory enabled data to be transferred in parallel with processing, and so the data transfer time did not add to the MAP Processor's execution time.

Work continues at UDRI to improve the current 0.9 fps processing rate using the heterogeneous system to 3 fps.

Architectural Feature	Implementation Benefits
GPIOX Interface	Direct full Camera Link video input into the MAP processor.
Common Memory	Local frame buffer storage.
Streams	Stream data as it is computed in one compute loop and consume it in a subsequent compute loop, enabling concurrent processing without data pooling in memory.
Streaming DMAs	Stream the input data directly into compute loops without needing to store data in local memories and similarly for computed output data.

Table 2: MAP Processor Benefits to Implementation

References

[1] P. Buxa. et al., "Mapping of a 2D SAR Backprojection Algorithm to an SRC Reconfigurable Computing MAP Processor", *Proceedings of the High-Performance Embedded Computing Workshop 2005* (HPEC'05), MIT Lincoln Laboratories, Lexington, MA, September 20-22, 2005.

[2] J. Isenman, et al., "Signal/Data Processor Implementation and Algorithms for Realtime Wide-Angle Ultra-Wideband SAR Image Formation", submitted to the *High-Performance Embedded Computing Workshop 2009* (HPEC'09), MIT Lincoln Laboratories, Lexington, MA, September 22-24, 2009.

[3] *SRC Computers' MAP® Processors for Airborne Intelligence, Reconnaissance, and Surveillance Applications*, MKT-043-00, SRC Computers LLC, Colorado Springs, CO, January 6, 2009.

[4] D. Johansen, *Video Stabilization and Target Localization Using Feature Tracking With Small UAV Video*, M.S. Thesis, Department of Electrical and Computer Engineering, Brigham Young University, December 2006.