Real-Time Interferometric SAS Processing with Ultra-Low Power Consumption

Jeremy Dillon
Kraken Robotic Systems Inc.
St. John's, NL, Canada
jdillon@krakenrobotics.com

***Abstract*—In order to fully realize the potential of unmanned systems, vehicles must carry sensors to observe the underwater environment, and sensor data must be processed in real-time for systems to intelligently interpret the environment and adapt their behavior accordingly. Interferometric synthetic aperture sonar (InSAS) is an ideal tool for seabed survey operations because of its superior area coverage rate and resolution compared to conventional sidescan and multibeam sensors. Due to the intensive nature of forming two SAS images per side of the vehicle plus bathymetric processing, InSAS data have traditionally been processed offline in a non-real-time environment after completion of the mission. Kraken Robotic Systems have been performing real-time InSAS processing of seabed imagery and bathymetry for the past five years using a variety of graphics processing units (GPUs). Here, we demonstrate that recent advances in the computational efficiency of InSAS algorithms and the development of ultra-low power GPUs for embedded systems have combined to create the situation where it is now possible to perform the complete InSAS processing chain at full resolution in real-time using extremely low energy consumption.**

***Index Terms*—synthetic aperture sonar, interferometry, real-time, graphics processing unit, low power**

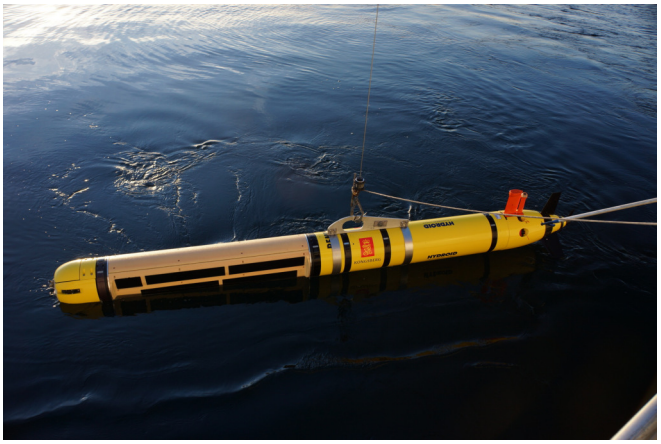

Fig. 1. Kraken AquaPix® InSAS installed and deployed on a REMUS 600 AUV. The forward Y-fin stability and control module, as used for other SAS systems [6], is not required.

I. Introduction

The ocean floor is largely unexplored and unmapped, yet seabed mapping is important for many applications such as oil and gas extraction, pipelines, communications cables, mineral extraction, habitat mapping, and underwater warfare [1]. For example, searching for stealthy mines on the seafloor is a key activity in naval mine countermeasures (MCM). Unmanned underwater vehicles have great potential to improve the efficiency and safety of seabed survey operations. In order to fully realize the potential of unmanned systems, vehicles must carry sensors to observe the underwater environment, and sensor data must be processed in real-time for systems to intelligently interpret the environment and adapt their behavior accordingly. For the MCM example, upon detecting a mine-like object on the seafloor, a vehicle might update its trajectory to obtain additional imagery or relay a message to another inspection and disposal system for further action.

Interferometric synthetic aperture sonar (InSAS) is an ideal tool for seabed survey operations because of its superior area coverage rate and resolution compared to conventional sidescan and multibeam sensors. InSAS provides seabed imagery with high image resolution, which is constant across the entire swath, and the ability to generate highly accurate bathymetry that is perfectly co-registered with the corresponding imagery. An InSAS consists of two vertically separated rows of transducers. A complex SAS image is formed for each row, and the two images are cross-correlated to obtain a high resolution map of relative bathymetry (i.e. the depth of the seabed relative to the vehicle trajectory). Due to the intensive nature of forming two SAS images per side of the vehicle plus bathymetric processing, InSAS data have traditionally been processed offline in a non-real-time environment after completion of the mission. Furthermore, when SAS images are computed onboard the vehicle, they are commonly formed at a degraded resolution for real-time implementation [2]. It is then necessary to reprocess the entire data set offline to obtain the full resolution.

InSAS processing is well suited to hardware acceleration using graphics processing units (GPUs). One attractive feature of GPU processing is the scalability of processing power. One may therefore achieve real-time performance by applying a sufficiently powerful processor. For example, faster than real-time SAS processing is obtained in [3] using 8 server-grade GPUs. However, such an approach is wholly unsuitable for autonomous underwater vehicles (AUVs) where the power consumption of an 8 GPU SAS processor would severely limit the endurance of a battery powered vehicle. Furthermore, results published in [2]–[4] only address the real-time formation of a single SAS image per side with no real-time bathymetry.

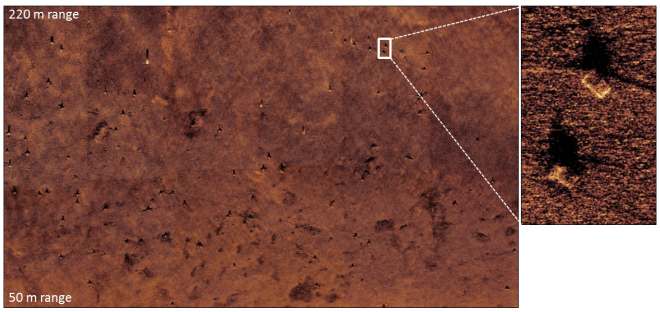

Fig. 2. SAS image mosaic from the target area near Jamestown Island.

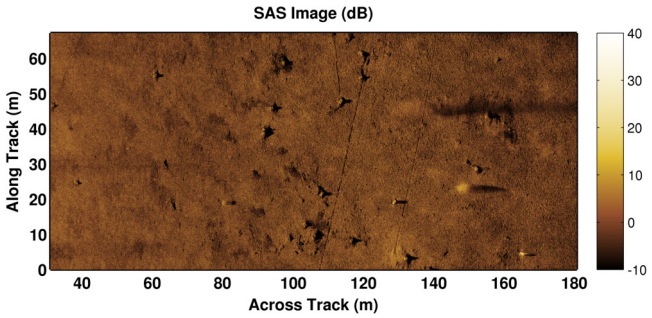

Fig. 3. Seabed image of lobster traps on a sandy seabed.

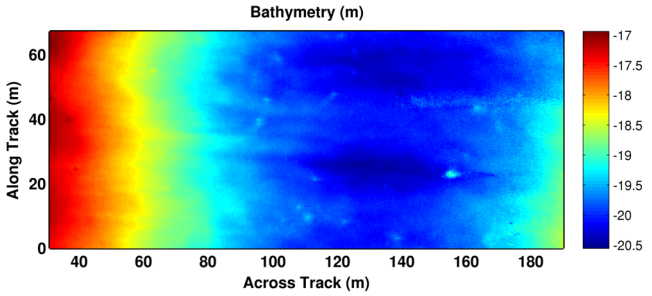

Fig. 4. Co-registered relative bathymetry corresponding to Fig. 3.

Kraken Robotic Systems have been performing real-time InSAS processing (imagery and bathymetry) for the past five years using a variety of embedded GPUs similar to the hardware in high performance notebook computers [5]. In this paper, we demonstrate that recent advances in the computational efficiency of InSAS algorithms and the development of ultra-low power GPUs for embedded systems have combined to create the fortuitous situation where it is now possible to perform the complete InSAS processing chain at the full resolution in real-time using extremely low energy consumption, to the point where the energy "cost" of real-time processing is negligible. Results are presented for a dataset collected from the AquaPix® InSAS installed on a REMUS 600 AUV. This dataset, collected in 2012, has served as a processing benchmark for various iterations of hardware and software development.

Using the sonar matched filter implementation as an example, we illustrate several techniques to optimize the performance of GPU computation including stream processing, concurrent data transfer and processing, data ordering for fast Fourier transforms, and memory alignment. Execution times and output from the NVIDIA Visual Profiler are presented for two types of GPUs, a desktop NVIDIA GTX 670 and a Jetson TX1, to illustrate trade-offs in processing power and memory bandwidth for embedded applications. It is shown that Kraken's INSIGHT software performs faster than real-time full resolution InSAS processing on the ultra-low power Jetson TX1 using a power consumption of only 7.5 watts during processing.

II. Data Collection

A. Sea Trial

In October 2012, the Naval Undersea Warfare Center (NUWC) Division, Newport, RI, and Kraken Robotic Systems entered a cooperative research and development agreement to evaluate the performance of the AquaPix® InSAS deployed from a REMUS 600 AUV [6]. The sonar and AUV are shown in Fig. 1. During 26 missions conducted over six days of testing, NUWC and Kraken collected significant amounts of InSAS data in Narragansett Bay and in Block Island Sound against both deployed targets and targets of opportunity [7]. The deployed target field, west of Jamestown Island and south of the Jamestown Bridge, is in water that is as deep as 28 m. The area contained a number of man-made objects of varying sizes and shapes that were deployed at marked positions specifically for this test, providing ground truth for target detection and identification. The InSAS sensor demonstrated the capability of generating high-resolution imagery to ranges as distant as 220 m. A mosaic with targets highlighted at 200 m range is shown in Fig. 2.

The SAS image for the processing benchmark is shown in Fig. 3. The image consists of numerous lobster traps on a sandy seabed with a maximum range of 180 m corresponding to a vehicle altitude of 18 m. The co-registered bathymetry in Fig. 4 shows a known seabed depression that compares favorably with bottom maps obtained from an independent bathymetric survey. The locations of lobster traps, visible in the bathymetry, also correlate well with the corresponding locations in the SAS imagery.

B. Sonar Description

AquaPix® is a wideband 300 kHz InSAS manufactured by Kraken Robotic Systems featuring a dual-row design for multipath suppression [8]. The InSAS consists of one transmitter module and four receiver modules. Each receiver module has a length of 53.3 cm, which is divided into 16 acoustic elements along track. Included in each module (both transmitter and receivers) are two rows of ceramic elements that are operated in distinct frequency bands. The short range lower row is angled down at a depression angle of 17.5° relative to horizontal, whereas the long range upper row has a depression angle of 5.7°. Each row surveys roughly half of the range swath of the sonar, with the short range row

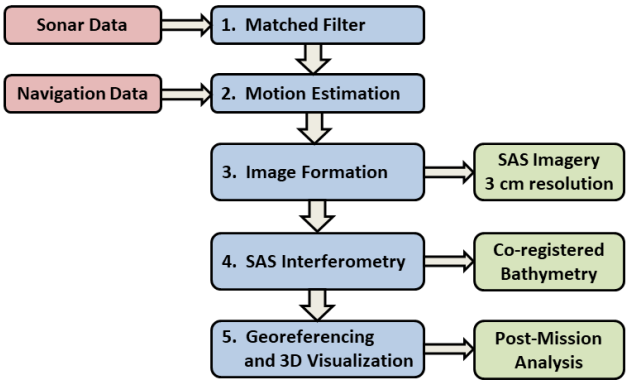

Fig. 5. Processing workflow for Kraken's INSIGHT software.

covering a range of 2 to 5 times altitude and the long range row covering a range of 5 to 10 times altitude. The center frequency and bandwidth for each row are programmable in software. The short and long range rows are operated at center frequencies of 337 and 240 kHz, respectively, to ensure that seabed echoes from each row do not interfere. As shown in Fig. 1, the four receiver modules are arranged in two rows of two modules. The vertically separated modules form an interferometer that provides a precise measurement of the angle of arrival of seabed echoes in the vertical plane. Measurements of seabed relative bathymetry are obtained by cross-correlating the complex SAS images from each row.

III. Data Processing

A. Workflow

Kraken's INSIGHT processing software features a modular architecture as shown in Fig. 5. The core processing steps are numbered 1 – 5 in blue. Inputs consist of acoustic data sampled by the sonar data acquisition system and navigation data from two sources, the embedded rate gyros within the sonar electronics pod and positioning data from the vehicle navigation system. The outputs consist of imagery with a constant resolution of 3 $\times$ 3 cm (range $\times$ cross-range) and co-registered bathymetry. Images from two vertically displaced arrays are used to form an interferometric image, which produces bathymetric maps with a software configurable horizontal resolution as fine as 6 $\times$ 6 cm. Larger bathymetric resolution cells result in improved vertical accuracy as more pixels are combined to obtain a depth measurement. The bathymetric map is therefore typically formed at 25 $\times$ 25 cm resolution, and this is the setting used for the benchmark data in Fig. 4. A key feature of the software architecture is that platform navigation data is only needed to georeference the imagery and bathymetry. The motion estimation for synthetic aperture focusing is fully self-contained, which simplifies the system integration with the host system.

SAS is capable of producing detailed imagery with range-independent resolution suitable for target detection and classification. In practice, operation onboard an AUV is constrained by the processing power available to form the synthetic aperture imagery in real-time and by the fact that vehicle trajectories often deviate significantly from an ideal linear track. The SAS processor must accommodate large translational errors, yaw oscillations, and a non-zero crab angle. Conventional image formation algorithms require a trade-off between motion tolerance, image quality, and processing speed. One of the unique features of Kraken's INSIGHT is that the exact same algorithms and resolution settings are used in real-time as for post-processing. Image formation is implemented efficiently using a proprietary algorithm that achieves a computational speed comparable to that of frequency domain algorithms (e.g. range Doppler and ω-k wavenumber algorithms) while retaining the accuracy of time domain backprojection. For interferometry, another proprietary algorithm is employed to robustly measure the relative bathymetry, i.e. the depth of the seabed relative to the vehicle trajectory. Absolute bathymetry is obtained by adding the vehicle depth to the relative bathymetry and performing standard corrections for tidal effects and refraction due to the sound velocity profile.

B. GPU Acceleration

All of the processing steps shown in Fig. 5 are performed on commercial-off-the-shelf (COTS) embedded PC hardware using NVIDIA GPU acceleration to achieve faster than real-time performance for a double sided interferometric configuration. One advantage of GPU technology is that it is inherently scalable. For example, in a desktop or rack-mount configuration, multiple cards may be interconnected to multiply the processing power. The mobile versions of NVIDIA GPUs are also widely available in embedded form factors with industrial-grade environmental ratings. Use of the CUDA parallel computing architecture ensures that the processing software runs on a wide variety of NVIDIA GPU hardware, allowing a seamless trade-off between power consumption and processing speed.

Although GPUs frequently surpass CPUs in terms of floating point operations per second (FLOPS) and energy efficiency, some caveats must be kept in mind. The most obvious constraint is that GPUs are only well suited for algorithms that are highly parallelizable. GPUs typically contain hundreds or thousands of processing cores. In order to maximize data throughput, it is desirable to have many cores executing similar instructions with simple branching and control logic. Fortunately, synthetic aperture processing is highly parallel in nature, with many identical computational operations performed for each sonar ping or for each channel of the receiver array.

In terms of computer architecture, CPUs retain the advantage of being tightly integrated with random access memory (RAM). Although GPUs have their own onboard RAM with a similarly fast bandwidth, raw data to be processed typically reside in CPU RAM and must be transferred to the GPU over a slower interconnection network such as PCI Express (PCI-E), which introduces latency. Much of the software development effort therefore revolves around structuring the computation

and data transfer to minimize latency while maximizing the utilization of the available GPU processing cores. Further details are discussed below using the matched filter as an illustrative example.

C. Matched Filter

Seabed reflectivity is measured with high range resolution by transmitting a wide bandwidth signal of several milliseconds duration and applying pulse compression upon reception of the seabed echoes. Compression is achieved by applying a filter that is "matched" to the transmitted waveform. In other words, the received signal is correlated with a replica of the basebanded transmit pulse. The resulting filtering operation maximizes the signal-to-noise ratio (SNR) in Gaussian additive noise [9].

For the AquaPix® InSAS, the transmit pulses are linear chirps of 40 kHz bandwidth centered on the carrier frequencies of 240 and 337 kHz. After basebanding, the replica signal $r(t)$ is a complex exponential

$$r(t) = W(t)\ \exp\left(i\ \phi(t)\right) \tag{1}$$

where $W(t)$ is a shading function applied to control sidelobes, and the phase $\phi(t)$ of the linear frequency modulated pulse is given by

$$\phi(t) = \pi\beta\left(\frac{t^2}{T} - t\right). \tag{2}$$

In (2), β is the chirp bandwidth and T is transmit pulse duration.

The matched filter is applied by zero padding the replica signal and performing the correlation in the frequency domain using the fast Fourier transform (FFT). Since the replica signal is common to all pings, the zero padding, Fourier transform, and complex conjugate operations are performed once. The result is then stored in GPU memory. For each ping, the filter must be applied to all 64 channels of the receiver array (32 channels for each of the upper and lower rows of the InSAS). The sequence of operations is as follows:

1) Transfer one ping of data from CPU RAM to GPU RAM
2) Convert 16-bit integer data to 32-bit floating point
3) Apply FFT to each channel of data
4) Multiply each channel by the replica conjugate FFT
5) Apply inverse FFT to each channel of data
6) Copy the result to a block of GPU RAM

The above steps are repeated for each ping of data that forms the SAS image (typically 100 to 200 pings depending on vehicle speed). Note that in the last step, matched filtered data remains on the GPU for further processing rather than being transferred to CPU RAM.

D. Implementation Details

The sonar data acquisition system records basebanded data as 16-bit integer in-phase and quadrature components, whereas the FFT algorithm expects floating point data. While the conversion could take place on either the CPU or the GPU, performing the conversion on the GPU is more efficient because it minimizes the quantity of data to be transferred across the relatively slow PCI-E interconnection. Nevertheless, since each InSAS image requires approximately 1 GB of raw 16-bit data, a significant delay would occur if the transfer was performed all at once. Fortunately, NVIDIA GPUs are capable of concurrent data transfer and processing when transfers occur between CPU and GPU RAM [10]. Concurrency is achieved in software by creating two processing streams that operate on alternate pings. Thus, data transfer in one stream occurs simultaneously with data processing in the other stream. In this manner, the latency of data transfer is effectively hidden, as demonstrated in Section IV.

It is well known that the FFT algorithm is most efficient when the transform size is a power of two, which can be obtained by zero padding the data and replica signal as necessary. A significant implementation detail is the ordering of data in GPU RAM upon which the FFT and inverse FFT are applied. Each ping of data is a two dimensional matrix of complex values with one dimension representing time (or range from the sonar) and the other dimension corresponding to the array channel number. The CUDA FFT library supports strided processing, which means that the FFT can be applied across either rows or columns of the matrix by specifying the offset, or stride, between consecutive data samples. Despite this flexibility, the GPU-accelerated FFT is most efficient when real and imaginary components are interleaved and the stride length is set to one, i.e. when consecutive samples in time are stored consecutively in GPU RAM.

Finally, the efficiency of the CUDA FFT is sensitive to the alignment of data to internal GPU memory boundaries that are powers of two in size. Therefore, zero padding each ping of data to a power of two in length has the additional benefit of aligning the transform with the memory structure of the GPU. The consumption of GPU RAM is also reduced by utilizing the CUDA FFT option to perform the transform "in place" overwriting original data with transformed values.

IV. Results

Processing times and output from the NVIDIA Visual Profiler are presented for two types of GPUs, a desktop GTX 670 and a Jetson TX1, to illustrate trade-offs in processing power and memory bandwidth for embedded applications. In Figs. 6 and 7, the blue and purple bars labeled "Compute" represent the utilization of GPU processing cores during matched filtering of the long range sonar data. Steps 2 to 5 of Section III-C appear as six processing blocks instead of four because the CUDA runtime environment implements the size 16384 FFT and inverse FFT operations using cascades of length 128 ($16384 = 128 \times 128$).

Above the compute section, gold bars indicate two types of memory operations. The longer bars labeled "HtoD" (Host to Device) represent transfers of sonar data from CPU RAM to GPU RAM over the PCI-E bus (Step 1). The shorter bars labeled "DtoD" (Device to Device) represent copies of filtered data within GPU RAM (Step 6). Toward the bottom of each figure, processing operations and data transfers are also shown

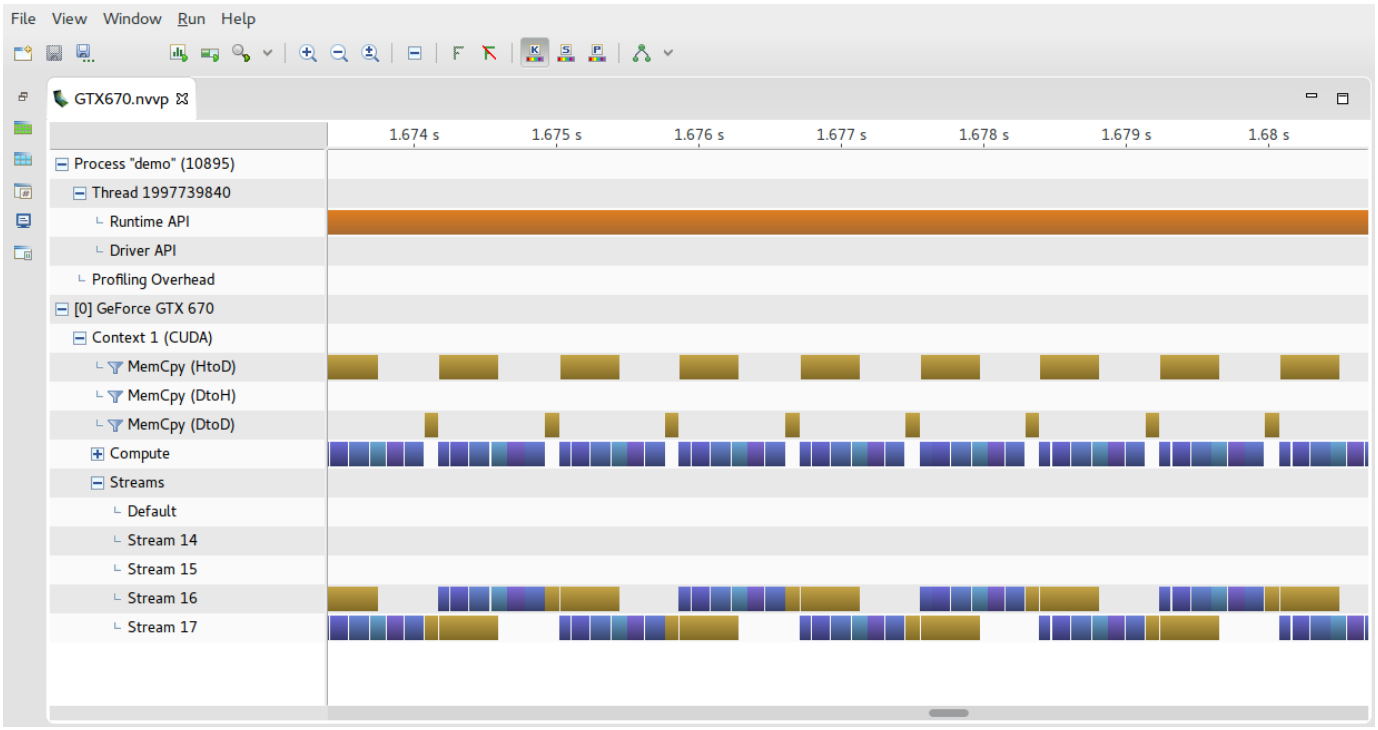

Fig. 6. NVIDIA Visual Profiler output for matched filtering using a GTX 670 GPU.

as they occur in their respective streams. The first two inactive streams in each figure correspond to the processing of short range sonar data, which is not shown.

A. Desktop GPU

Results for an Intel Core i7 desktop PC with an NVIDIA GeForce GTX 670 GPU are presented in Fig. 6. The GTX 670 features 1344 CUDA cores, 2 GB of GDDR5 memory onboard the GPU, and a memory bandwidth of 192.2 GB/s. The matched filter processing time is approximately 1 ms per ping. The entire InSAS processing chain (matched filter, motion estimation, two SAS images per row, and bathymetric processing) is completed in 2.41 s for a single side, compared to 42 s of real-time for the vehicle to travel the along track distance of 68 m in Fig. 3. Processing on the GTX 670 is therefore 17.4 times faster than real-time, or 8.7 times faster for a double sided system.

Fig. 6 demonstrates that processing data on a ping-by-ping basis with two streams effectively hides the latency of the CPU to GPU data transfer while maintaining a high level of GPU utilization. The matched filter speed is limited by the processing capacity of the GPU rather than by a memory or data transfer bottleneck. Therefore, adding more cores, for example by choosing the GTX 680 over the 670, would improve the overall performance. However, beyond a processing power increase of roughly 80% the computation would be limited by memory bandwidth.

TABLE I
JETSON TX1 INSAS POWER CONSUMPTION

State	Power (W)
Idle	2.5
Processing Peak	10.2
Processing Average	7.5

B. Ultra-Low Power Embedded GPU

Results for a Jetson TX1 module are shown in Fig. 7. The TX1 features a quad-core ARM Cortex A57 CPU, 256 CUDA cores, 4 GB of LPDDR4 memory that is divided between the CPU and GPU, and a memory bandwidth of 25.6 GB/s. The matched filter processing time is approximately 7 ms per ping. The InSAS processing chain for a single side is completed in 9.55 s. Processing on the TX1 is therefore 4.4 times faster than real-time, or 2.2 times faster for a double sided system. In Fig. 7, the TX1 matched filter speed is also limited by GPU processing capacity rather than memory throughput, which has recently been doubled for the newer Jetson TX2 module.

In Table I, power measurements were performed during InSAS processing with an AC electrical load meter. The measurements include power consumed by the AC to DC power supply for the Jetson TX1 board. One attractive feature of the TX1 is that upon completion of processing, the power consumption returns to the idle level of 2.5 watts almost instantaneously. The average measurement is the average value during InSAS processing, not counting time spent in the idle state. For a double sided InSAS system, the TX1 would spend

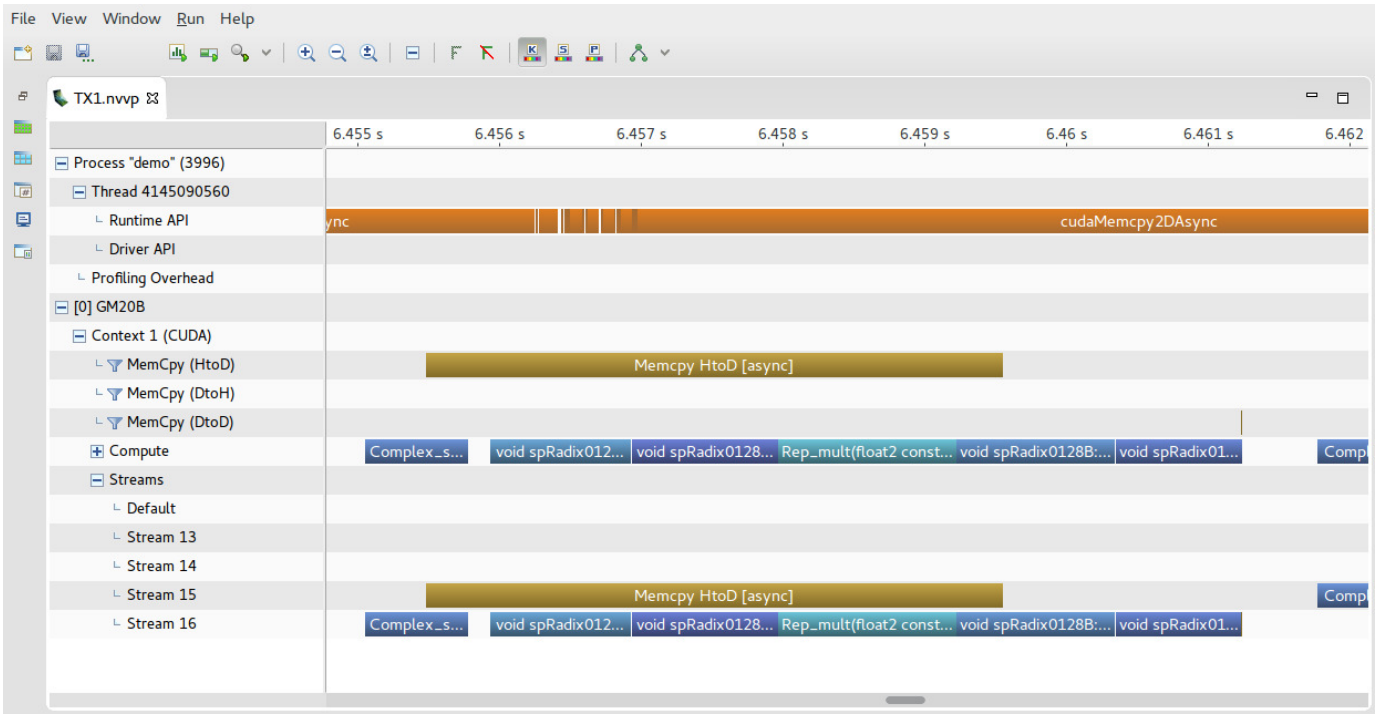

Fig. 7. NVIDIA Visual Profiler output for matched filtering using a Jetson TX1 GPU.

TABLE II
INSAS PROCESSING TIME FOR BENCHMARK DATASET

GPU	Elapsed Time (s)	Faster Than Real-Time Ratio
GTX 670	2.41	17.4
Jetson TX1	9.55	4.4

roughly half the time with the GPU at idle, resulting in an overall rate of energy consumption potentially as low as 5 watts depending on the nature of other tasks assigned to the processor such as recording data, communicating with the sonar data acquisition system, georeferencing, and interacting with the vehicle control system. The processing times for both GPUs are summarized in Table II for the single sided InSAS benchmark dataset.

V. CONCLUSION

The combination of computationally efficient InSAS algorithms and the development of ultra-low power GPUs make it possible to perform the complete InSAS processing chain at full resolution in real-time using extremely low energy consumption. Several techniques were illustrated to optimize the performance of GPU computation including stream processing, concurrent data transfer and processing, data ordering for fast Fourier transforms, and memory alignment. Execution times and output from the NVIDIA Visual Profiler were presented to illustrate trade-offs in processing power and memory bandwidth for embedded applications. It was shown that Kraken's INSIGHT software performs faster than real-time full resolution InSAS processing on the Jetson TX1 using an average power consumption of only 7.5 watts during processing. Ultra-low power real-time InSAS processing is therefore an enabling technology for the development of intelligent and persistent AUVs that sense their environment and adapt their behavior accordingly.

REFERENCES

[1] H. J. Callow, P. E. Hagen, R. E. Hansen, T. O. Sæbø, and R. B. Pedersen, "A new approach to high-resolution seafloor mapping," J. Ocean Technology, vol. 7, no. 2, pp. 13–22, 2012.

[2] F. Baralli, M. Couillard, J. Ortiz, and D. G. Caldwell, "GPU-based real-time synthetic aperture sonar processing on-board autonomous underwater vehicles," in Proc. MTS/IEEE OCEANS Conference, Bergen, Norway, 2013.

[3] D. P. Campell and D. A. Cook, "Using graphics processors to accelerate synthetic aperture sonar imaging via backpropagation," in Proc. High Performance Embedded Computing Workshop (HPEC'10), MIT Lincoln Laboratory, Lexington, MA, USA, 2010.

[4] H. J. Callow, "Comparison of SAS processing strategies for crabbing collection geometries," in Proc. MTS/IEEE OCEANS Conference, Seattle, WA, USA, 2010.

[5] D. Shea, D. Dawe, J. Dillon, and S. Chapman, "Real-time SAS processing for high-Arctic AUV surveys," in Proc. IEEE/OES Autonomous Underwater Vehicles (AUV 2014), Oxford, MS, USA, 2014.

[6] R. P. Stokey et al., "Development of the REMUS 600 autonomous underwater vehicle," in Proc. MTS/IEEE OCEANS Conference, Washington, DC, USA, 2005.

[7] A. N. Mirkin, J. DiCecco, and T. A. Merchant, "Operation of a synthetic aperture sonar from a REMUS 600 autonomous underwater vehicle," Naval Undersea Warfare Center Division, Newport, RI, Tech. Rep. NUWC-NPT 13-013, 2013.

[8] M. Pinto, "Interferometric synthetic aperture sonar design optimized for high area coverage shallow water bathymetric survey," in Proc. Underwater Acoustic Measurements, Kos, Greece, 2011.

[9] I. G. Cumming and F. H. Wong, Digital Processing of Synthetic Aperture Radar Data. Artech House, 2005.

[10] J. Cheng, M. Grossman, and T. McKercher, Professional CUDA C Programming. John Wiley & Sons, 2014.