REAL-TIME CAPABILITY OF DLR'S BEAMFORMING SYNTHETIC APERTURE RADAR PROCESSING ARCHITECTURE

Maron Schlemon[1], *Martin Schulz*[2], *Rolf Scheiber*[1], *Marc Jäger*[1] *and Joel Amao Oliva*[1]

[1]German Aerospace Center (DLR), Microwaves and Radar Institute, Germany
[2]Technical University of Munich, Chair of Computer Architecture and Parallel Systems, Germany

ABSTRACT

Synthetic Aperture Radar (SAR) enables the generation of realistic and high-resolution 2D or 3D representations of landscapes. Typically, radar instruments are deployed in specially equipped, low-flying aircraft that capture a significant amount of raw data, necessitating image reconstruction processing. However, the aircraft's limited onboard processing capabilities (power, size, weight, cooling, and communication bandwidth to ground stations) and the need to generate multiple SAR products, such as slant-range and geo-coded images during a single flight, require efficient onboard processing and transmission to the ground station. This paper outlines the processing architecture of the digital beamforming SAR (DBFSAR) employed by the German Aerospace Center (DLR) and the specific measures implemented to enable onboard processing. We elucidate the essential software optimizations and their integration into the SAR onboard routines, facilitating (near) real-time capability under certain conditions. Furthermore, we share the insights gained from our work and discuss their applicability to other processing scenarios with limited resource availability.

Index Terms— Real-time SAR; Resource-Constrained Computing; On-Board SAR Processing;

1. INTRODUCTION

SAR generates high-resolution, 2- or 3-dimensional representations of landscapes. The data acquisition is typically conducted using radar instruments on specialized moving platforms, such as aircraft or satellites. SAR systems generate a significant amount of raw data that then must be processed for image reconstruction. Limited resources, such as power consumption, space, weight, cooling, and communication bandwidth to ground stations, restrict onboard processing capabilities. This poses a challenge for real-time on-board processing. Additionally, the hardware is generally fixed and constrained to the equipment available at the initial commissioning of the platforms, which have a much longer lifespan than IT equipment. Airborne hardware cannot be easily replaced or upgraded, as it is subject to rigorous and complex certification procedures. Therefore, any application designed for these environments must be optimized to fully exploit the underlying constrained hardware and efficiently be integrated into the concrete environment and workflows that can be executed. It is worth noting that different hardware might be installed if the system were to be built today. However, the challenges would remain unchanged. Similar constraints exist in many other processing scenarios where high-performance processing must be executed on hardware with severely limited resources.

This paper presents a series of novel optimizations targeting both the computational kernels and the overall SAR imaging processing workflow. In particular, we make the following contributions:

- We enhance the core functions of the imaging algorithm and align them with the given processing system's register sizes, caches, and memory space. This precise alignment results in optimal vectorization levels and, thus, in a high hardware utilization rate.
- We efficiently distribute the fast kernels over many compute cores and processing boards to parallelize the data processing further.
- We optimize the execution environment, data management, and multi-processor setup and show how this enables real-time SAR processing.

To assess our approach, we present its effectiveness of applying it for SAR image formation and geo-coding, which was performed during an airborne flight campaign using DLR's Digital Beamforming Synthetic Aperture Radar (DBFSAR) processing system. The results presented in this paper are representative of edge-based applications, which not only require high performance but also have to operate within the constraints commonly encountered in remote or isolated platforms.

2. DLR'S DBFSAR SYSTEM OVERVIEW

The DBFSAR system, described in [1], is an advanced high-resolution airborne SAR with digital beamforming capabilities. It is operated at X-band and features 12 simultaneous

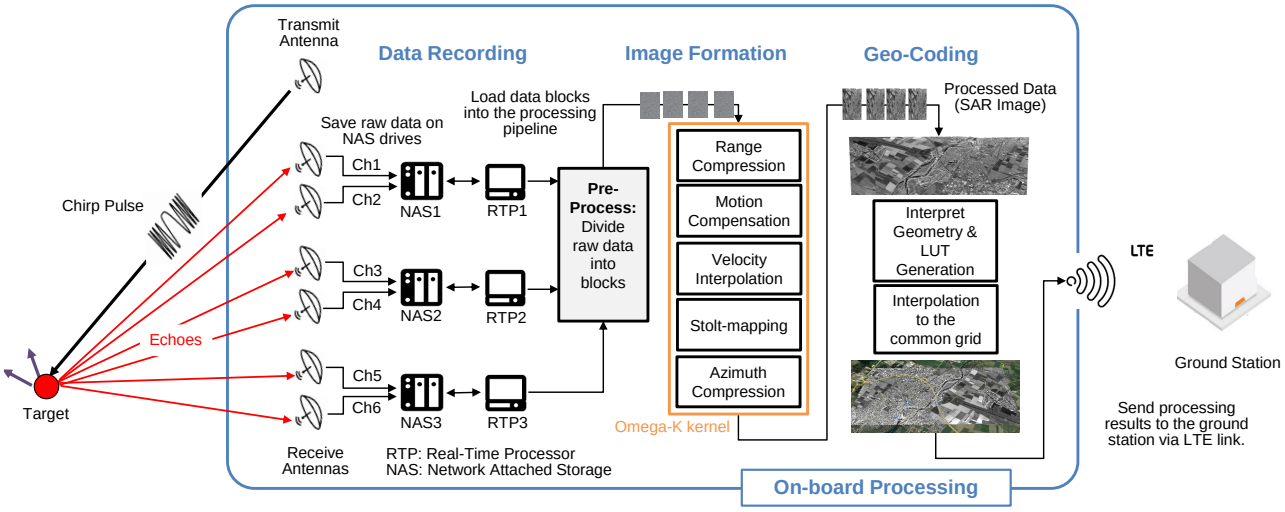

Fig. 1. On-board image formation processing flow

receive and four sequential transmit channels, each with a bandwidth of 1.8 GHz, flexible digital beamforming antenna configurations, and a high-precision navigation and positioning unit. Furthermore, it has a dedicated processing system that includes three NAS drives, three Intel quad-core CPUs (RTP1-RTP3), a total of 48 GB of RAM, and an LTE modem to downlink the on-board processed data products to the ground station, as illustrated in Fig. 1. The onboard processing system can perform various SAR applications, such as Moving Target Indication (MTI), SAR image formation, and geo-coding. Table 1 shows the system details. In the context of MTI, as outlined in [2], a specialized antenna configuration composed of six receive antennas is employed. It acquires multi-channel SAR data, which is stored on the NAS drives. Finally, the RTPs initiate the processing by executing a pre-processing step, which divides the raw data into blocks and transfers them to the processing pipeline.

Table 1. Real-time Processing Hardware

Processor	
Name	Intel® Core™ i7-3610QE
Code Name	Ivy Bridge
Cores	4
Hyper threading	Off
Base frequency	2.3 GHz
Supported instructions	SSE4.2, AVX
Memory	
Type	DDR3
Capacity	16384 (2x8192) GB
Operation	Dual Channel
Frequency	1333 MHz
Operating System	
Distribution	Linux / openSUSE

3. SAR ONBOARD PROCESSING PROCEDURE

Fig. 1 also represents the procedure of the SAR image formation pipeline, which is based on the Omega-k (EOK) algorithm, as well as the geo-coding procedure, in which the resulting focused image is interpolated into the corresponding geographic geometry using a digital elevation model (DEM). To achieve this, look-up tables (LUT) are generated that associate each pixel in the radar geometry to a given latitude and longitude of a common geographical grid, with the last step of the algorithm being the interpolation from the radar geometry to this new geometry using said LUTs. The EOK algorithm is based on physical optics and wave scattering theory. It is less susceptible to the platform's motion and can generate high-resolution images. Regarding computational demands, the Omega-K algorithm requires numerous complex mathematical calculations and a substantial amount of computational resources [3]. To make the pipeline real-time capable, we first analyzed our Python-based model of the EOK algorithm. We identified its performance bottlenecks using sophisticated profilers, transferred them to C, and optimized them using low-level software optimizations, such as Single Instruction Multiple Data (SIMD) and Open Multi-Processing (OpenMP) [4], [5]. SIMD and OpenMP are techniques applied to optimize the run-time performance of code by exploiting the par-

allel processing capabilities of modern CPUs. A detailed explanation of our optimizations is provided in [7], [8], and [9].

4. IMPLEMENTATION AND COMPUTE KERNEL OPTIMIZATIONS

One set of optimizations directly targets the execution of the compute-intensive kernels identified in the last section. In particular, we focus on using SIMD instructions, which can potentially enable a high degree of parallelism. Single instruction multiple data (SIMD) instructions are implemented on Intel's general-purpose processors to enhance performance. Only a single instruction is fetched and applied to a data set (vectors). Intel's Intrinsic Instruction Set Library provides a convenient solution to implement SIMD instructions [4, 5, 6]. When the data is stored sequentially, Intel's SIMD instruction sets can be applied efficiently and thus create fast single-core kernels (components). Therefore, we used it, for example, to vectorize the interpolation required repeatedly in the processing pipeline [9].

5. EVALUATION IN RECENT AIRBORNE CAMPAIGN

To evaluate the effectiveness of our optimizations towards achieving real-time capability, we have established the following real-time condition: the system is real-time capable if the processing of a SAR image can be completed within the typical $t_{rt} = 90s$ data acquisition time frame. This criterion was used as a reference during our recent flight campaign, where we successfully tested our developments for various image resolutions.

As explained earlier, we have tailored the SAR algorithms to the architecture to leverage the parallelism available on the target platform. By distributing the processing kernel across the cores of the three RTPs, we effectively exploit all the available parallelism. Figure 2 illustrates the results of our tests for the multi-core implementation at three different example resolutions. When implementing the algorithm using only a single core, we observe that real-time capability cannot be achieved, as the processing times for all resolutions exceed $t_{rt_1} = 90s$. However, utilizing two processing cores, we reach a threshold where real-time capability is attained. This occurs when the resolution is larger than (1.0, 1.0), as depicted in Figure 2, which also displays the corresponding speedup. In the case of two cores, the speedup approaches the ideal, effectively reducing the overall run time by a factor of two. This improvement holds even when utilizing four cores, enabling real-time capability for all resolutions, as the run times are shorter than t_{rt}. Thus, as defined in our study, we achieve real-time capability for all cases.

Regarding the geo-coding procedure, data is split into blocks in range and azimuth to be processed sequentially, in order to preserve memory. Future implementations will adapt

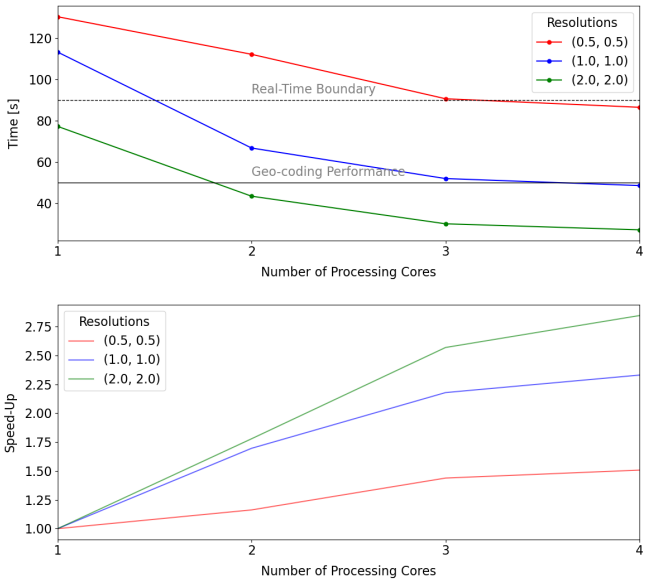

Fig. 2. Performance comparison: runtime (top) and speed-up (bottom)

the concurrent processing of each geo-coding block. The computational time of the geo-coding blocks does not depend on the processed range and azimuth resolutions, since the resolution of the common grid is a result of the input DEM used during geo-coding.

6. LESSONS LEARNED

If we put these findings into the context of the on-board processing where a total of nine cores (three per CPU) are available, we can state that the DBFSAR system is per our definition real-time capable for the specified image resolutions. We achieved this by understanding the problem entirely and thus being able to align performance-critical parameters, such as block size, resolution, accuracy, etc., with the system's architecture. Regarding algorithmic performance, a gain can be realized by hardware-oriented development. It has turned out that SIMD instructions provide an effective way of optimization concerning computation time. This, though, requires an increased programming effort and does not ensure a performance improvement regarding SAR image quality. However, the benefit is for the achievable resolution within the limited time frame imposed by the real-time requirement.

The findings will be taken into account for upcoming projects. Care will be taken during the planning phase to ensure that the CPU supports the latest instruction sets and that sufficient cores are available. In addition, the dimensions are selected so block processing can run without a performance loss.

Fig. 3. Geo-coded SAR image processed onboard (Kaufbeuren, Germany 2022)

7. OUTLOOK AND FUTURE WORK

It has been shown that the hardware-oriented optimizations, along with aligning SAR parameters (block size, resolution, etc.), accelerate SAR routines and thus the overall SAR application. Future work consists of improving the algorithms mainly by utilizing all hardware resources optimally. From the research that has been carried out, it is possible to conclude that DLR's DBFSAR system can process SAR data in real-time when using all the resources effectively.

The developed hardware accelerations will also benefit the onboard processing of future drone-based SAR systems. They will equally well be used within the ground segment servers for offline processing.

8. REFERENCES

[1] A. Reigber et al: The High-Resolution Digital-Beamforming Airborne SAR System DBFSAR. MDPI Remote Sensing. vol. 12, no. 11, pg. 1710, 2020

[2] Baumgartner, Stefan V. und Joshi, Sushil Kumar (2021) Onboard Processing Concept for Maritime Surveillance Demonstrated with DLR's Airborne Radar Sensors F-SAR and DBFSAR. In: Proceedings of the European Conference on Synthetic Aperture Radar, EUSAR. European Conference on Synthetic Aperture Radar (EUSAR), 2021-03-29 - 2021-04-01, Leipzig, Germany.

[3] A. Reigber, E. Alivizatos, A. Potsis and A. Moreira, "Extended wavenumber-domain synthetic aperture radar focusing with integrated motion compensation," in IEE Proceedings - Radar, Sonar and Navigation, vol. 153, no. 3, pp. 301-310, June 2006

[4] Agner Fog, "Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD, and VIA CPUs," Technical University of Denmark

[5] Intel Corporation, "Intel® 64 and IA-32 Architectures-Software Developer's Manual,"

[6] Intel Corporation, "Intrinsics Guide," https://www.intel.com/content/www/us/en/docs/intrinsics-guide/index.html

[7] M. Schlemon and J. Naghmouchi, "FFT Optimizations and Performance Assessment Targeted towards Satellite and Airborne Radar Processing," 2020 IEEE 32nd International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), Porto, Portugal, 2020, pp. 313-320, doi: 10.1109/SBAC-PAD49847.2020.00050

[8] M. Schlemon, R. Scheiber, S. Baumgartner, S. K. Joshi, M. Jaeger and S. Pasch, "On-board Processing Architecture of DLR's DBFSAR / V-SAR System," EUSAR 2022; 14th European Conference on Synthetic Aperture Radar, Leipzig, Germany, 2022, pp. 1-5.

[9] M. Schlemon, M. Schulz and R. Scheiber, "Resource-Constrained Optimizations For Synthetic Aperture Radar On-Board Image Processing," 2022 IEEE High Performance Extreme Computing Conference (HPEC), Waltham, MA, USA, 2022, pp. 1-8, doi: 10.1109/HPEC55821.2022.9926327