Optical Flow to Capture Cardiac Motion

Sonia Sharapova

***Abstract*—This project implements a parallel processing system for analyzing cardiac motion in medical imaging sequences using optical flow analysis. The system processes image sequences from cardiac MRI scans and returns them as GIFs with vectors showing motion of the heart contractions overlaid on top of them. Subsequently, a parallel processing technique was implemented which achieved significant performance improvements, with the best-case speedup of 2.31x on large datasets. The system successfully generates visualizations of cardiac motion patterns, providing medical professionals with enhanced tools for analyzing heart function through automated motion detection and visualization.**

I. Introduction

Medical image processing plays a crucial role in modern healthcare, particularly in the analysis of sequential imaging data. Cardiac motion analysis through medical imaging provides essential insights into heart function and potential abnormalities. However, processing large sequences of medical images presents significant computational challenges, particularly when real-time or near-real-time analysis is desired.

This study presents a parallel processing system designed to analyze DICOM image sequences using optical flow analysis. The approach focuses on optimizing the computational efficiency of cardiac motion detection while maintaining accuracy in motion pattern identification. By implementing parallel processing techniques, the hope is to reduce the processing time required for analyzing large medical image datasets.

II. Background

The evolution of cardiac motion analysis through medical imaging has seen several significant developments. Early work in parallel medical imaging focused on data transmission efficiency. Arvanitis et al. developed a pioneering parallel method for medical image transmission using the DICOM protocol, demonstrating significant speedups of 2.2 to 3.5 times for MRI images over local networks [1]. In the domain of optical flow computation, Koch et al. established fundamental parallel algorithms for real-time motion analysis. Their implementation on a Connection Machine supercomputer showed that parallel processing could achieve near-real-time performance while maintaining accuracy comparable to human perception [2]. Current state-of-the-art approaches primarily focus on GPU acceleration and specialized hardware solutions. Christoph et al. recently advanced the field by implementing GPU-accelerated numerical motion tracking for optical mapping of contracting cardiac tissues [3]. Their work demonstrated real-time motion compensation capabilities for videos with successful application across different species and imaging conditions. Although very powerful, these current solutions often require specialized hardware setups, limiting their accessibility in standard clinical environments. This project addresses this limitation by developing a CPU-based parallel implementation written with python that can be deployed in typical clinical settings while still achieving significant performance improvements.

III. Data

Motion detection of heart contractions were computed from publicly available CineMRI data, which is medical imaging data that contains a series of images that capture the heart's motion during a cardiac cycle[6]. These images are taken by repeatedly imaging the heart at a single location during the cardiac cycle.

The CineMRI data used in the project was obtained from publicly available data provided by the Cardiac Atlas Project. The Sunnybrook Cardiac Data (SCD)[4] and the AMRG Cardiac Atlas[5] were downloaded directly from their site. Both datasets provide a complete set of labeled MRI images of a patient's heart in the DICOM image format.

IV. Methodology

The system processes DICOM images through a structured pipeline with five key stages. First, the DICOM Processing Module loads files using the pydicom library and preprocesses them by normalizing pixel intensity values, applying histogram equalization to enhance contrast, and resizing frames to uniform dimensions. Next, the Feature Detection Implementation employs the Shi-Tomasi algorithm with fine-tuned parameters for cardiac imaging, ensuring accurate detection of up to 150 high-quality, noise-free corners per frame.

Subsequently, the Optical Flow Computation step tracks movement between frames using the Lucas-Kanade method. Its configuration, optimized for cardiac motion, balances accuracy and computational efficiency.

To enhance performance, the system employs a parallelization strategy: processing multiple patient datasets simultaneously at the folder level and using a pipelined approach for sequential frame analysis.

Finally, the Visualization Generation module creates animated GIFs to illustrate detected motion. Green vectors, overlaid on the original medical images, represent motion direction and magnitude, providing a clear depiction of cardiac motion patterns. This integrated approach ensures efficient and accurate processing and visualization of DICOM data.

Ideally, all tests would be run on the midway cluster, but due to the sizes of the files this analysis was performed on a local machine and averaged over three runs.

V. Results

The system successfully generated animated GIFs showing cardiac motion patterns with overlaid motion vectors. These visualizations effectively highlight areas of cardiac motion, the direction and magnitude of the movement, and temporal patterns in cardiac cycles. Figure 1. shows selected frames from the AMRGAtlas output, where the green arrows represent the motion vectors found by the algorithm.

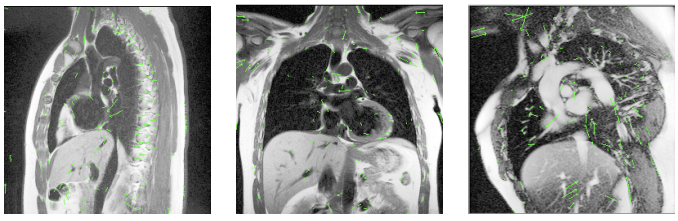

Fig. 1. Frames showing motion vectors in AMRGAtlas data.

Performance Results

The system was tested on multiple datasets of various sizes. The most significant performance improvement was observed with the largest available dataset – SCD_IMAGES_01 from Sunnybrook at 1.31GB. Figure 2. shows the performance of the parallelized computation across three separate runs.

Benchmark Summary for SCD_IMAGES_01:

- Average Original Time: 122.16 seconds
- Average Parallel Time: 52.95 seconds
- Average Speedup: 2.31x

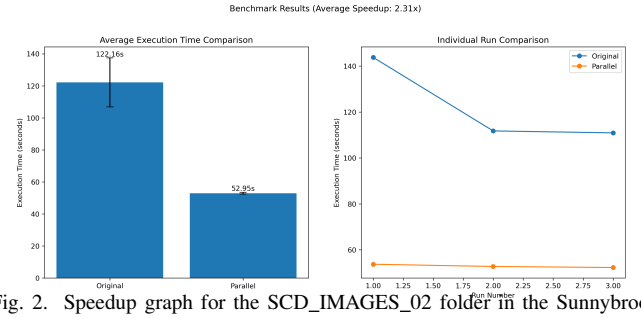

Fig. 2. Speedup graph for the SCD_IMAGES_02 folder in the Sunnybrook Dataset

Speedup Comparisons

Other datasets showed varying degrees of improvement:

- SCD_IMAGES_02: 1.84x speedup
- SCD_IMAGES_03: 2.09x speedup
- AMRGAtlas: 1.33x speedup

Larger datasets took a longer time to compute, but showed significant improvement with parallelization. This is expected as there is more work distributed.

Running on Midway Cluster

Due to the large size of the datasets, only the AMRG Atlas Dataset was processed on the midway cluster. On midway, the program was executed on 8 CPUs per task. Compared against local machine performance, the average parallel speedup increased from 1.33x to 1.5x.

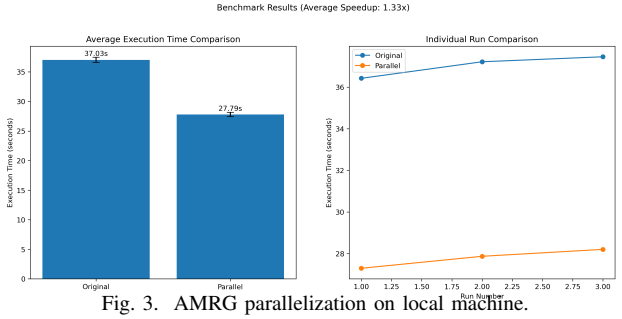

Fig. 3. AMRG parallelization on local machine.

This shows that parallelization performance improves given isolated execution and bigger computational power.

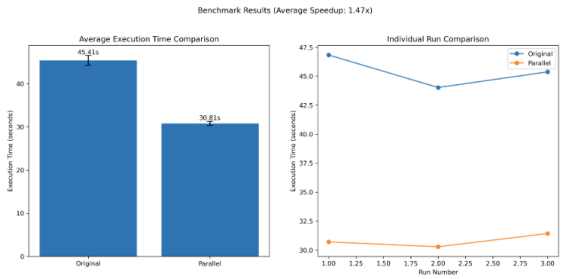

Fig. 4. AMRG parallelization on midway cluster.

VI. Discussion

The parallel implementation of optical flow analysis for cardiac motion detection demonstrates significant potential for advancing medical image processing in clinical settings. The achieved speedup of 2.31x on larger datasets (1.31GB) compares favorably with previous CPU-based implementations, though it falls short of specialized GPU implementations that have reported speedups of 3-5x [3]. However, this system is fully implemented in python and is compatible with CPU-based systems, giving it an advantage in its accessibility and deployability as it operates on standard clinical hardware without requiring specialized GPU infrastructure. This makes it a more practical solution for widespread adoption in medical facilities. The correlation between dataset size and speedup efficiency suggests particular utility for processing the increasingly large imaging datasets common in modern medical practice. With additional development time and resources, several enhancements could further improve the system's capabilities. Integration of GPU acceleration when available could provide optional performance boosts while maintaining CPU compatibility as a fallback. Development of real-time processing capabilities and direct integration with medical imaging workflows could streamline clinical applications. Additionally, implementing machine learning techniques for feature detection could potentially improve motion tracking accuracy in low-contrast regions. The implementation demonstrated in this paper provides a combination of significant speedup and broad compatibility provides a valuable contribution to the field, particularly for facilities without access to specialized computing resources. This balance of performance and accessibility helps bridge the gap between advanced motion analysis capabilities and practical clinical implementation.

VII. Conclusion

The parallel processing implementation presented in this paper successfully reduced the computational time required for cardiac motion analysis while maintaining accuracy in motion detection. The system demonstrates particular efficiency with larger datasets, achieving up to 2.31x speedup. These improvements in processing efficiency, combined with effective motion visualization, contribute to the advancement of cardiac imaging analysis tools.

References

[1] Maani, Rouzbeh et al. “A parallel method to improve medical image transmission.” Journal of digital imaging vol. 25,1 (2012): 101-9. doi:10.1007/s10278-011-9387-9

[2] Bülthoff, H et al. “A parallel algorithm for real-time computation of optical flow.” Nature vol. 337,6207 (1989): 549-55. doi:10.1038/337549a0

[3] Lebert, Jan et al. “Real-Time Optical Mapping of Contracting Cardiac Tissues With GPU-Accelerated Numerical Motion Tracking.” Frontiers in cardiovascular medicine vol. 9 787627. 24 May. 2022, doi:10.3389/fcvm.2022.787627

[4] Radau P, Lu Y, Connelly K, Paul G, Dick AJ, Wright GA. “Evaluation Framework for Algorithms Segmenting Short Axis Cardiac MRI.” The MIDAS Journal – Cardiac MR Left Ventricle Segmentation Challenge, http://hdl.handle.net/10380/3070 download

[5] AMRG Cardiac Atlas, Auckland MRI Research Group, Auckland, New Zealand. Available online http://www.cardiacatlas.org/studies/amrg-cardiac-atlas/. download

[6] Kellman, Peter et al. “High spatial and temporal resolution cardiac cine MRI from retrospective reconstruction of data acquired in real time using motion correction and resorting.” Magnetic resonance in medicine vol. 62,6 (2009): 1557-64. doi:10.1002/mrm.22153

Code References

[7] Motion Estimation with Optical Flow: Optical Flow Information

[8] Optical Flow in OpenCV: OpenCV

[9] Feature Detection Using Shi-Tomasi Corner Detection: OpenCV Informational