Journal of Scientech Research and Development
Volume 5, Issue 2, December 2023
P-ISSN: 2715-6974 E-ISSN: 2715-5846
Open Access at: https://idm.or.id/JSCR/index.php/JSCR

STITCHING IMAGE WITH SCALE INVARIANT FEATURE TRANSFORM USING PARALLEL PROGRAMMING MPI (MESSAGE PASSING INTERFACE)

STITCHING IMAGE DENGAN SCALE INVARIANT FEATURE TRANSFORM MENGGUNAKAN PARALEL PROGRAMMING MPI (*MESSAGE PASSING INTERFACE*)

Dian Kartika Sari
Jurusan Kesehatan Politeknik Negeri Jember
E-mail: dian@polije.ac.id

ARTICLE INFO

Correspondent:

Dian Kartika Sari
dian@polije.ac.id

Key words:
educational games, GDLC, introduction to household appliances

Website:
https://idm.or.id/JSCR/index.php/JSCR

Page: **1006 - 1015.**

ABSTRACT

MPI (Message-Passing Interface) is a communications protocol for parallel programming of computers. A feature of MPI is that the data stored on each computer is completely separate from that stored on other computers. If one computer needs data from another, or wants to send certain data to all other computers, the appropriate library must explicitly request data transfer. In this discussion, virtual machines are used as a substitute for using computers to carry out parallel programming simulations. The virtual machine used is a virtualbox with a connected NAT network. In testing the performance of Parallel Programming with MPI (Message Passing Interface), the image stitching process was carried out using feature extraction with Scale Invariant Feature Transform. The results of the first test produced a processing speed (stitching image) of 5.79 seconds with an efficiency of 0.132 on the master node, a processing speed of 5.85 seconds on the slave1 node with an efficiency of 0.087 and a processing speed of 5.87 seconds on the slave2 node with an efficiency of 0.065.

INFO ARTIKEL

Koresponden

Dian Kartika Sari
dian@polije.ac.id

Kata kunci:

MPI *(Message-Passing Interface), Stitching Image,* Virtual Mesin

Website: https://idm.or.id/JSCR/index.php/JSCR

Hal: 1006 - 1015

ABSTRAK

MPI (*Message-Passing Interface*) merupakan protokol komunikasi untuk pemrograman paralel komputer. Fitur dari MPI adalah data yang disimpan pada setiap komputer sepenuhnya terpisah dari yang tersimpan di komputer lain. Jika satu komputer membutuhkan data dari yang lain, atau ingin mengirim data tertentu ke semua komputer lain, secara eksplisit harus library yang sesuai meminta transfer data. Pada pembahasan ini digunakan virtual mesin sebagai pengganti penggunaan komputer untuk melakukan simulasi pemrograman paralel. Virtual mesin yang digunakan adalah *virtualbox* dengan jaringan NAT yang telah terhubung. Pada pengujian kinerja *Paralel Programming* dengan MPI (*Message Passing Interface*) ini dilakukan proses penggabungan gambar (stitching image) dengan menggunakan ektraksi fitur dengan *Scale Invariant Feature Transform* . Hasil pengujian pertama menghasilkan kecepatan proses (*stitching image*) selama 5.79 detik dengan efisiensi 0,132 di node master, kecepatan proses 5.85 detik di *node slave1* dengan efisiensi 0,087 dan kecepatan proses 5.87 detik pada node slave2 dengan efisiensi 0,065.

PENDAHULUAN

Perkembangan teknologi informasi yang memerlukan komputasi cepat dengan data yang besar merupakan sebuah tantangan di masa sekarang. Teknologi Big Data dan komputasi awan memerlukan analisis dan pemrosesan data besar sehingga para peneliti berupaya untuk menemukan pendekatan terbaik untuk memprosesnya dengan kinerja tinggi, biaya yang rendah dan akurasi tinggi (Dheyab, Abdullah, and Abed, 2019). Untuk mempercepat komputasi pada pemrosesan data yang berjumlah besar diperlukan teknologi HPC *(High Performance Computing)*. *High Performance Computing* merupakan metode komputasi yang digunakan mengatasi permasalahan data yang memiliki kompleksitas tinggi dengan penggunaan banyak data *(Fakhri, et. al.,* 2024).

Salah satu teknik yang digunakan dalam metode HPC adalah komputasi parallel dengan MPI. Komputasi paralel adalah teknik komputasi dengan cara melakukan komputasi secara bersamaan pada beberapa komputer independen secara bersamaan. Beberapa penelitian tentang komputasi parallel ini contohnya pada penelitian (Dheyab, *et. al,* 2019), yang mengusulkan manajemen memori pada arsitektur Big Data untuk menangani data dalam jumlah besar dengan menggunakan algoritma AES dan parallel *Message Passing Interface* (MPI). Penelitian pada (Ferreira *et al.,* 2017) menggunakan komputasi parallel CUDA pada *image reduction* dengan menggunakan *Gaussian Pyramid* untuk mempercepat proses komputasi. Sedangkan pada penelitian (Wibawa, *et.al,* 2024 dilakukan penelitian untuk membandingkan kinerja komputasi Paralel menggunakan *Model Message*

Passing pada SIM RS (Sistem Informasi Manajemen Rumah Sakit). Pada penelitian Ristov, *et. al*. (2014) dilakukan penelitian untuk membandingan kinerja komputasi parallel pada platform e-Learning dan dilakukan *benchmarking* untuk mengukur indikator kinerja *parallel programming* tersebut seperti pada kecepatan, percepatan dan efisiensi dari implementasi *parallel programming.* Dari referensi penelitian terkait sebelumnya, dapat diketahui bahwa komputasi parallel Programming dengan MPI memiliki kinerja yang cukup baik untuk mengatasi pemrosesan data dalam jumlah besar. Oleh karena itu, pada penelitian ini digunakan komputasi parallel untuk mengukur kinerja pemrosesan *Stitching Image* pada gambar digital menggunakan virtual mesin yang bertindak sebagai paralel komputer yang akan dibandingkan kinerjanya dengan pemrosesan sekuensial. Banyaknya virtual mesin yang dibangun sebagai parallel MPI akan bergantung pada besarnya kapasitas memori.

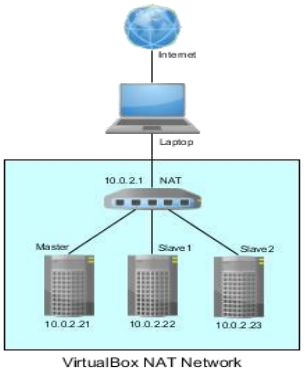
METODE

Instalasi Virtual Mesin

Vitual mesin yang digunakan adalah *Oracle VM Virtualbox* yang dapat diunduh pada alamat *https://www.virtualbox.org/*. Sistem operasi yang digunakan pada virtual mesin adalah debian 8 NET Core 2.0, media penyimpanan 8 GB dan memori sebesar 512 MB. Langkah pertama, instalasi virtual dengan pilih menu *New* pada *Virtualbox* untuk membuat virtual mesin baru. Selanjutnya dilakukan tahap pengalokasian memori. Memori yang digunakan adalah 512MB dan ukuran penyimpanan hardisk yang digunakan adalah 8 GB. Selanjutnya dilakukan pemilihan tipe Virtual Mesin dengan memilih tipe VDI dan dilakukan *mount image* sistem operasi dengan memilih lokasi dari *file image* sistem operasi ke *VirtualBox*. Selanjutnya dilakukan proses pemasangan sistem operasi dengan hanya memilih untuk memasang *system utility* dan *SSH server* saja.

Konfigurasi Jaringan

Pengaturan topologi jaringan dari virtual mesin seperti pada Gambar 1, di mana laptop sebagai mesin induk. Kemudian virtualbox mempunyai tur mampu mengatur jaringan dari virtual mesin seperti NAT (*Network Address Translation*) di mana *virtualbox* mampu menyediakan *gateway* atau *bridge* di mana *ethernet card* divirtualisasikan secara langsung oleh virtual mesin .



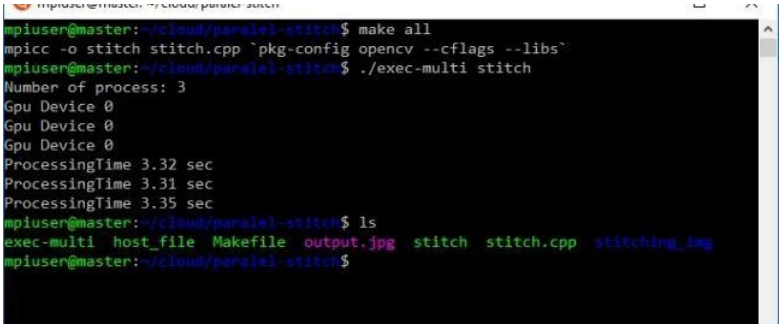
Gambar 1. Topologi jaringan NAT VirtualBox

Menjalankan MPI Cluster di dalam LAN

Program MPI yang berjalan dalam satu mesin untuk memproses kode secara paralel, memanfaatkan lebih dari satu ini CPU. Di ruang lingkup yang lebih luas, dimana mengambil lebih dari satu komputer ke jaringan node yang terhubung bersama di local area network. Untuk menjalankan MPI *cluster* di dalam LAN maka langkah-langkah konfigurasinya seperti berikut ini :

a. Langkah 1: Konfigurasi *File Host*
   File host digunakan oleh sistem operasi pada master untuk memetakan nama host ke alamat IP
b. Langkah 2: Membuat akun pengguna baru
   Membuat akun pengguna baru dengan nama pengguna yang sama di semua mesin.
c. Langkah 3: *Setting SSH*
   Virtual mesin ini akan berkomunikasi melalui jaringan via SSH, dan berbagi data melalui NFS.
d. Langkah 4: Menyiapkan NFS
   User dapat berbagi direktori melalui NFS di *master* dimana *client* melakukan mount untuk bertukar data. *Setting* akan dilakukan pada NFS-Server dan NFS-Client
e. Langkah 5: Menjalankan *MPI program*
   Program MPI yang digunakan disini adalah program *stitching image*, dengan nama *stitch.cpp*. Untuk melakukan kompilasi program secara paralel dengan MPI digunakan perintah*: $ mpicc -o stitch stitch.cpp*

Berikut hasil kompilasi program untuk *stitching image* dengan pemrograman MPI paralel pada Gambar 2. dengan tiga proses komputasi paralel yang digunakan.



Gambar 2. Hasil Kompilasi Program Stitch.cpp

Stitching Image

Stitching image merupakan proses menggabungkan beberapa gambar yang memiliki bidang yang saling *overlap* untuk menghasilkan satu gambar panorama atau gambar dengan resolusi tinggi. *Stitching image* dilakukan dengan melakukan beberapa tahapan. Pada penelitian ini, dilakukan penggabungan gambar yang saling *overlap* untuk mendapatkan satu gambar panorama. Contoh kumpulan sample gambar yang akan digabungkan dapat dilihat pada Gambar 3.

Gambar 3. Contoh Gambar yang akan Digabung

a. Registrasi citra

Registrasi citra melibatkan fitur yang cocok pada sekumpulan citra atau dengan menggunakan metode direct alignment untuk mencari *alignment* gambar yang dapat meminimalisasi *sum of absolute differences* (SAD) pada setiap piksel yang overlap.

b. Kalibrasi

Kalibrasi bertujuan untuk me-minimalkan perbedaan antara model lensa ideal dan kombinasi lensa kamera yang digunakan.

c. Blending

Blending melibatkan penyesuaian yang terinci pada tahap kalibrasi. Dikombinasikan pada pemetaan ulang gambar ke proyeksi output.

Scale-Invariant Feature Transform

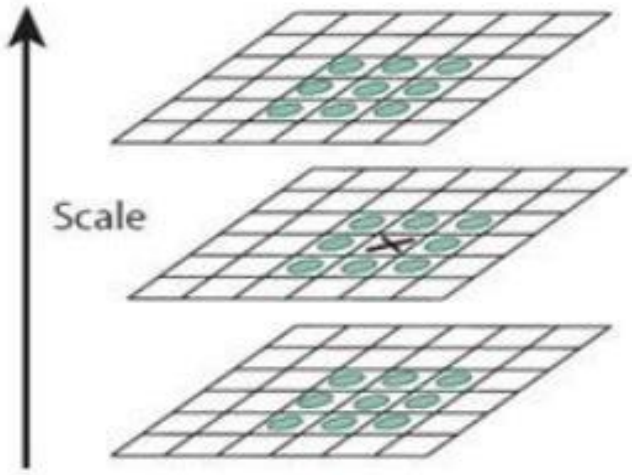
Scale-Invariant Feature Transform (SIFT) merupakan algoritma dalam *computer vision* yang berguna untuk mendeteksi dan mendeksripsikan fitur lokal pada gambar. Algoritma SIFT mengekstrak keypoints dari gambar referensi dan menyimpan *keypoints* tersebut. Setiap objek pada gambar baru dapat dikenali dengan membandingkan fitur yang terdapat pada gambar baru dengan fitur *keypoints* yang tersimpan dan menemukan kandidat fitur berdasarkan pada *Euclidean distance* dari vektor *keypoint*. SIFT memiliki beberapa tahapan yaitu:

a. *Scale-Invariant Feature Detection*

Jika dalam sebuah *image* dimisalkan terdapat gambar sebuah pohon dan kita ingin memisahkan antara daun, batang, ranting dengan tepat. Maka, sebelum memisahkan setiap bagian tersebut harus diberikan rincian atau label pada masing-masing bagian agar tidak tertukar. Pada hal ini digunakan metode *Gaussian Blur* (yang dibuktikan secara matematik, dan didukung beberapa alasan yang menguatkan). SIFT membutuhkan ruang skala ke tingkat berikutnya. Dengan menggunakan ruang skala, dihitung perbedaan antar 2 ruang skala berturut-turut dan kemudian digunakan metode *Different of Gaussian (DoG)* untuk mengeliminasinya (Agus, 2013).

b. *Keypoint localization*

Setelah menghasilkan ruang skala kemudian digunakan untuk menghitung *different of Gaussian* dan dilanjutkan menghitung *Laplacian of Gaussian* dengan hasilnya adalah menghasilkan titik titik atau point. Selanjutnya adalah mencari maxima/minima dalam gambar Dog (*Different of Gaussiant*). Langkahnya perhitungannya adalah menemukan titik maxima/minima dari tiap pixel gambar raw terlebih dulu. Caranya dengan melewati setiap pixel dan diperiksa setiap tetangganya.



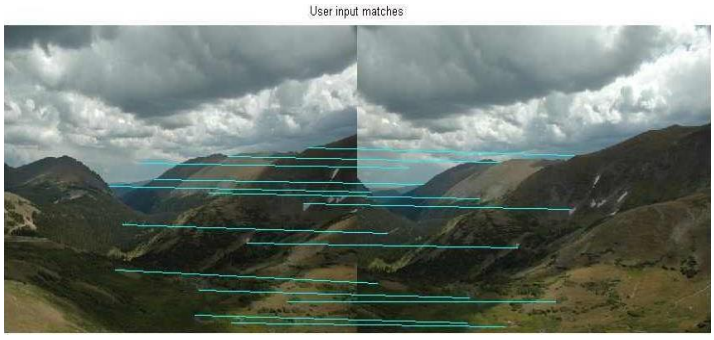
Gambar 4. Menentukan Titik *Maxima* dan *Minima* untuk *Keypoint*

c. *Orientation Assigment*

Setelah mendapatkan titik pixel yang merupakan *keypoint* hasil seleksi, langkah berikutnya adalah menetapkan orientasi dari setiap keypoint. Caranya adalah dengan mendapatkan arah *gradient* dan besaran sekitar setiap *keypoint*. Kemudian diketahui orientasi yang paling menonjol di wilayah *keypoint* tersebut. (Agus, 2013)

Penggabungan Citra

Dua citra yang telah dihitung menggunakan metode SIFT untuk mendapatkan keypoint dari dua citra tersebut, kemudian digunakan sebagai acuan mencari fitur yang sama pada gambar kedua. Misal ditemukan ada 10 keypoint digambar 1 dan 12 keypoint digambar kedua. Kemudian keypoint digambar 1 akan dicocokkan dengan gambar kedua. Walaupun jumlah keypoint tidak sama, dari keypoint tersebut akan dicocokkan keypoint yang memiliki nilai vektor yang sama. Proses pencarian keypoint yang sama dari dua citra dinamakan *matching.*



Gambar 5. *Matching* pada Dua Image

Titik yang sudah dilakukan *matching* akan digunakan untuk menentukan transformasi citra pertama ke citra kedua agar kedua citra homogen. Kemudian setelah kedua citra sudah homogen, maka citra pertama dijumlahkan dengan citra kedua dengan syarat wadah penjumlahan lebih besar dua kali dari luas kedua citra.

Kompilasi Penggabungan *Image* atau *Stitching*

Setelah program *stitching* atau penggabungan *image* selesai dibuat. Maka berlanjut ke tahap selanjutnya yaitu penambahan library atau header mpi ke program stitching agar dapat melakukan proses penggabungan image secara paralel dari beberapa server. Gambar 6, menunjukkan proses akhir dari stitching. Bagian hitam adalah wadah baru yang digunakan untuk menyatukan dua image. Dikarenakan nilai yang hitam merupakan dua kali nilai matriks yang sama maka nilai tengah tersebut dibagi dua agar tidak melebihi batas maksimum *channel rgb*.

Gambar 6. Hasil *stitching image*

HASIL DAN PEMBAHASAN

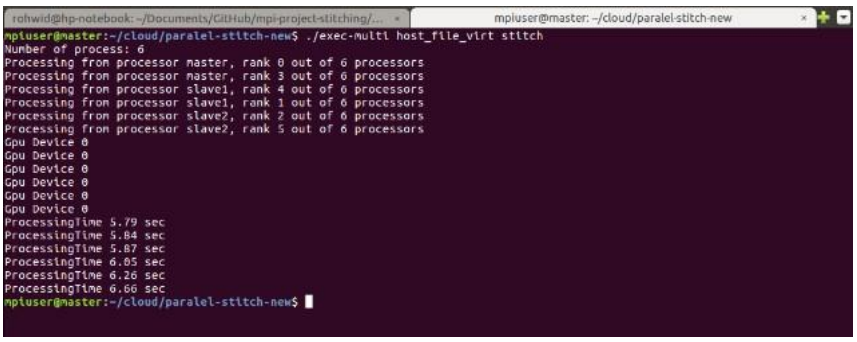
Tahap pengujian dilakukan dengan membandingkan hasil kompilasi pada sebuah komputer dan dibandingkan kinerjanya dengan melakukan kompilasi secara paralel menggunakan virtual mesin dengan 1 master dan 2 *slave*. Pada dasarnya proses eksekusi program dengan menggunakan MPI, program di eksekusi secara *round robin* atau bergantian dalam kurun waktu tertentu, sesuai dengan beban program yang akan dikompilasi.

1. **Pengujian sekuensial dengan Sebuah Komputer**
   Program dieksekusi dengan menggu-nakan *compiler* g++ dan ditambah dengan *library* dari *opencv* maka diperoleh hasil selama 1.53 detik pada saat memproses 13 buah gambar dan 2,37 detik pada saat memproses 23 buah gambar.

Tabel 1. Hasil Pengujian *Stitching Image* dengan Sebuah Komputer

Jumlah Komputer	*Stitching Image*	Waktu (detik)
1	13 gambar	1,53
1	23 gambar	2,37

2. **Pengujian *Stitching image* dengan MPI Paralel Virtual mesin**
   Program dieksekusi dengan menggunakan *compiler* mpic++ dan ditambah dengan *library* dari opencv maka diperoleh hasil selama 5,79 detik, 5,85 detik, 5,87 detik, 6,05 detik, 6,26 detik dan 6,66 detik pada node master, slave 1 dan slave 2 secara *round robin* seperti pada Gambar 7 pada saat memproses gambar secara paralel.



Gambar 7. Proses *Stitching* dengan Paralel Virtual Mesin

Hasil pengujian dapat dilihat pada Tabel 2 dan 3 di bawah ini:

Tabel 2. Hasil Pengujian *Stitching Image* dengan Paralel Virtual Mesin dengan 13 Gambar

Jumlah PC (virtual mesin)	*Stitching Image*	Waktu Paralel (detik)
2	13 gambar	5,79
3	13 gambar	5,85
4	13 gambar	5,87

Tabel 3. Hasil Pengujian *Stitching Image* dengan Paralel Virtual Mesin dengan 33 Gambar

Jumlah PC (virtual mesin)	***Stitching Image***	**Waktu Paralel (detik)**
2	23 gambar	6,05
3	23 gambar	6,26
4	23 gambar	6,66

Hasil pengujian pertama menghasilkan kecepatan proses *(stitching image)* selama 5.79 detik di node master, 5.85 detik di node slave1 dan 5.87 detik pada node slave2 dengan sample gambar sebanyak 13 buah dan dilakukan komputasi parallel secara *round robin.* Hasil pengujian kedua menghasilkan kecepatan proses *(stitching image)* selama 6.05 detik di node master, 6.26 detik di node slave1 dan 6.66 detik pada node slave2 dengan sample gambar sebanyak 23 buah.

3. Pengujian *Speed Up* dan Efisiensi

Pengujian hasil Speed up (S) merupakan pengujian dengan membandingkan hasil yang didapat antara waktu sekuensial dengan waktu paralel. Pada hasil pengujian sekuensial dengan sebuah komputer/CPU dibutuhkan waktu 1,53 detik untuk melakukan *stitching image* dengan 13 buah gambar dan dibutuhkan waktu selama 2,37 detik melakukan *stitching image* dengan 23 buah gambar. Untuk mengetahui nilai *speed up* maka waktu hasil pengolahan data dengan pengujian sekuensial yang menggunakan satu komputer akan dibagi dengan nilai waktu pengolahan data menggunakan program paralel MPI sesuai dengan jumlah komputer yang digunakan. Sedangkan pengujian efisiensi (E) didapatkan dari perbandingan antara *nilai speed up (S)* dengan jumlah komputer/CPU yang digunakan (p). Nilai Efisiensi mengukur seberapa efisien penggunaan sejumlah komputer di dalam topologi jaringan paralel yang dibangun (Ferreira *et al.,* 2017)

Tabel 4. Hasil Pengujian Stitching Image dengan Paralel Virtual Mesin pada 13 Gambar

Jumlah virtual mesin	Waktu Paralel (detik)	*Stitching Image (gambar)*	*Speed Up (S)*	Efisiensi (E)
2	5,79	13	0,264	0,132
3	5,85	13	0,261	0,087
4	5,87	13	0,261	0,065

Tabel 5. Hasil Pengujian Stitching Image dengan Paralel Virtual Mesin pada 23 Gambar

Jumlah virtual mesin	Waktu Paralel (detik)	*Stitching Image (gambar)*	*Speed Up (S)*	Efisiensi (E)
2	6,05	23	0,391	0,195
3	6,26	23	0,378	0,126
4	6,66	23	0,355	0,088

Proses *stitching image* secara paralel dengan virtual mesin lebih dari satu berlangsung lebih lama dari pada menggunakan *single* komputer. Hal tersebut terjadi karena diperlukannya proses sinkronisasi data antar *node*. Sehingga faktor koneksi jaringan turut berpengaruh. Selain itu sesuai dengan konsep pemrosesan paralel pada GPU yaitu jika beban perhitungan yang diproses ringan contohnya penjumlahan dua angka *float*, maka pemrosesan dengan *single* komputer akan lebih cepat dibandingkan dengan pemrosesan paralel dengan GPU. Dari hasil pengujian terlihat juga apabila sample gambar yang diproses lebih banyak maka hasil pemrosesan *stitching image* akan menjadi lebih lama. Namun, perbedaan waktu komputasi dan speed up yang terlihat tidak terlalu besar. Pada pengujian di 13 gambar, perbedaan waktu antara menggunakan 3 virtual mesin dan 4 virtual mesin hanya 0,02 detik. Pada pengujian dengan 23 gambar, rata-rata efisiensi terlihat lebih baik. Hal ini membuktikan bahwa parallel programming dengan MPI memberikan efisiensi yang cukup baik pada pemrosesan data yang lebih banyak.

SIMPULAN

Penggunaan virtual mesin yang dipasang pada sebuah komputer dapat melakukan pemrosesan data secara paralel. Hal tersebut dikarenakan sifat dari virtual mesin yang digunakan adalah *full-virtualization* yang berarti processor dari komputer secara utuh dijadikan *virtual*. Jumlah dari virtual mesin hanya bergantung pada kapasitas RAM. Hasil pengujian dari proses kompilasi virtual komputer didapatkan bahwa pemrosesan dengan *single* komputer masih lebih cepat daripada pemrosesan secara parallel dengan menggunakan virtual komputer, namun perbedaan kecepatan yang terjadi tidak terlalu signifikan dan semakin banyak sample gambar yang digunakan maka efisiensi yang dihasilkan juga semakin baik, yang membuktikan bahwa komputasi parallel bekerja lebih efisien pada data yang berjumlah besar.

DAFTAR PUSTAKA

S. A. Dheyab, M. N. Abdullah, and B. F. Abed, "A novel approach for big data processing using message passing interface based on memory mapping," J. Big Data, vol. 6, no. 1, p. 112, Dec. 2019, doi: 10.1186/s40537-019-0275-3.

A. Fakhri, A. Nasir, M. Nordin, and A. R. Mamat, "A Study of Image Processing in Agriculture Application under High Performance Computing Environment," 2012. Accessed: Jan. 10, 2024. [Online]. Available:

https://www.semanticscholar.org/paper/A-Study-of-Image-Processing-in-Agriculture-under-Fakhri-Nasir/744c8609d6e7e2410296dae77a1ac4f4d87da8b5

C. B. R. Ferreira et al., "Parallel CUDA Based Implementation of Gaussian Pyramid Image Reduction," 2017. Accessed: Jan. 10, 2024. [Online]. Available: https://www.semanticscholar.org/paper/Parallel-CUDA-Based-Implementation-of-Gaussian-Ferreira-Soares/ee75015035d5dd115c555d6c14e858dfe9ca24a8

I. P. A. P. Wibawa, I. D. Giriantari, and M. Sudarma, "Komputasi Paralel Menggunakan Model Message Passing Pada SIM RS (Sistem Informasi Manajemen Rumah Sakit)," Maj. Ilm. Teknol. Elektro, vol. 17, no. 3, Art. no. 3, Dec. 2018, doi: 10.24843/MITE.2018.v17i03.P20.

Ristov, S., Gusev, M., & Velkoski, G. (2014, April). Cloud e-learning and benchmarking platform for the parallel and distributed computing course. In 2014 IEEE Global Engineering Education Conference (EDUCON) (pp. 645 -651). IEEE

S. agus, "Pencocokan Citra Berbasis Scale Invariant Feature Transform (SIFT) Menggunakan Arc Cosinus," SkripsiFakultas Ilmu Komput., 2013, Accessed: Jan. 11, 2024. [Online]. Available: http://eprints.dinus.ac.id/12478/