Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 11, November 2018, hlm. 5700-5707 http://j-ptiik.ub.ac.id

Analisis Perbandingan Proses Pengolahan Citra Menggunakan FPGA dan Mikrokomputer

Muhammad Naufal[1], Wijaya Kurniawan[2], Dahnial Syauqy[3]

Program Studi Teknik Komputer, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]lafuan@hotmail.com, [2]wjaykurnia@ub.ac.id, [3]dahnial87@ub.ac.id

Abstrak

Pengolahan citra merupakan aspek penting dalam kehidupan karena kebutuhan manusia semakin hari semakin bertambah, sehingga dibutuhkan sistem yang mampu mengolah citra dengan efektif. Untuk mengetahui sistem apa yang bisa mengolah citra secara efektif, dilakukan analisis perbandingan pengolahan citra antara kedua sistem, FPGA dan Mikrokomputer. FPGA yang digunakan adalah myRIO sedangkan Mikrokomputer yang digunakan adalah Raspberry Pi. Penelitian dilakukan dengan algoritma pengolahan citra yang umum seperti *Gaussian blur, Laplacian edge* dan *Sobel edge*. Pertama citra RGB dikonversi ke *grayscale* agar citra mudah diolah lalu dihilangkan noisenya dengan algoritma *gaussian blur*. Setelah itu citra dideteksi tepi dengan algoritma *laplacian edge* dan *sobel edge*. Pengujian dilakukan dengan mengolah tiga ukuran citra berbeda pada tiga algoritma yang berbeda dan dilakukan sepuluh kali pengujian dan diambil waktu rata-rata pengolahan citra pada kedua sistem. Hasil waktu rata-rata pengolahan citra dengan algoritma *Gaussian blur* adalah 0.485s pada FPGA dan 0.165s pada Mikrokomputer. Untuk algortitma *Laplacian edge* waktu rata-rata adalah 0.492s pada FPGA dan 0.202s pada Mikrokomputer sedangkan untuk algoritma *Sobel edge* waktu rata-rata nya adalah 0.498s pada FPGA dan 0.234s pada Mikrokomputer. Namun sebenarnya untuk semua algoritma, waktu FPGA tetap sama namun berbeda pada tiga citra yang berukuran berbeda masing-masing adalah 0.01053, 0.03074 dan 0.06076 detik.

Kata kunci: *citra, FPGA, mikrokomputer, gaussian blur, laplacian edge, sobel edge*

Abstract

Image processing is an important aspect in life because human needs are increasingly growing day by day, so we need a system that can process image effectively. To find out what systems can effectively process images, a comparison analysis of image processing between two systems, FPGA and Microcomputer is performed. FPGA used is myRIO while the Microcomputer used is Raspberry Pi. The study was conducted with common image processing algorithms such as Gaussian blur, Laplacian edge and Sobel edge. First the RGB image is converted to grayscale for easy image processing and then eliminated its noisenya with gaussian blur algorithm. After that the image is detected edge with laplacian edge and sobel edge algorithms. The test was performed by processing three different image sizes on three different algorithms and performed ten tests and taken the average time of image processing on both systems. The mean time of image processing with Gaussian blur algorithm is 0.485s on FPGA and 0.165s on Microcomputer. For Laplacian edge mean time algorithm is 0.492s on FPGA and 0.202s on Microcomputer while for Sobel edge algorithm its average time is 0.498s on FPGA and 0.234s on Microcomputer. But actually for all algorithms, FPGA time remains the same but different on three different sized images respectively are 0.01053, 0.03074 and 0.06076 seconds.

Keywords: *image, FPGA, microcomputer, gaussian blur, laplacian edge, sobel edge*

1. PENDAHULUAN

Image (*citra*) sebagai salah satu komponen multimedia yang memegang peranan sangat penting sebagai bentuk informasi visual. Citra memiliki karakteristik yang tidak dimiliki oleh data teks, yaitu citra kaya dengan informasi. (Sholihin & Purwoto, 2015). Informasi ini yang bisa kita dapatkan dari sebuah citra membutuhkan suatu metode pengolahan citra

sesuai. Karena kebutuhan manusia yang semakin hari semakin bertambah, dibutuhkan sistem yang mampu mengolah citra dengan efektif. Maka dibutuhkan sistem yang handal dalam pengolahan citra, jika sebuah sistem semakin efektif saat mengolah citra, semakin cepat juga citra itu dapat menyampaikan informasi pada manusia.

Suatu proses pengolahan citra dibutuhkan perangkat keras dan perangkat lunak yang bekerja berkesinambungan dalam mengolah citra digital. Perangkat keras dalam pengolahan citra yakni kamera dan alat peraga. Proses pengolahan citra umumnya dilakukan dari piksel ke piksel yang bersifat parallel. Adapun sistem dari perangkat kerasi ini terdiri dari beberapa sub sistem komputer, masukan citra, konreol proses interaktif, penyimpanan berkas citra dan perangkat keras khusus pengolahan citra (Ahmad, 2005). Beberapa contoh perangkat keras pengolahan citra antara lain DSP, Mikrokontroller, Mikrokomputer, Komputer, GPU dan FPGA. Pada penelitian ini, penulis memilih FPGA dan Mikrokomputer sebagai *platform* perbandingan karena dua *platform* ini dapat diandalkan dalam pengolahan citra.

Penelitian yang dilakukan Maleeha Kiran, Kan Mei War, Lim Mei Kuan, Liang Kim Meng & Lai Weng Kin Peniliti menggunakan pendekatan *Hardware in the loop* untuk penelitiannya. Penelitian ini bertujuan untuk menginvestigasi performa terbaik untuk sistem pengawasan otomatis dari berbagai *platform.* Sistem yang dibuat menggunakan aplikasi Matlab dan *library* dari *Xilinx System Generator* dengan metode *Illumination Analysis.* Penelitian ini membandingkan waktu pengolahan antara FPGA, C++ dan Matlab. Dari hasil penelitian, FPGA memiliki waktu pengolahan yang lebih cepat dibandingkan C++ dan Matlab. Namun menurut penulis penelitian ini kurang dapat dihandalkan karena masih sebatas *prototype* dan tidak di aplikasikan pada sistem yang sesungguhnya (Kiran, War, Kuan, Meng, & Kin, 2008)

Penelitian yang dilakukan Megah Mulya dan Abdiansah. Peneliti menggunakan metode Multi-threading sebagai pendekatannya. Penelitian ini bertujuan membandingan waktu pengolahan citra digital dengan metode *multi-threading* dan *single-threading* yang biasa dikerakan didalam processor. Hasil penelitian ini adalah metode *multi-threading* lebih cepat dalah pengolahan citra digital. Hal ini disebabkan karena teknik pemrograman multi-threading dapat lebih mengoptimalkan kinerja prosesor terutama untuk komputer dengan prosesor ganda. Namun menurut penulis penelitian ini kurang dapat dihandalkan karena hasil pengujian perangkat lunak tersebut bisa berbeda untuk waktu pengujian yang berbeda. Hal ini disebabkan beban prosesor yang berbeda untuk situasi perangkat lunak aplikasi yang dieksekusi yang sedang dieksekusi oleh sistem operasi. (Mulya & Abdiansah, 2013)

Berdasarkan uraian diatas tentang pentingnya pengolahan citra, maka penulis memilih perangkat keras NI myRIO-1900 berbasis FPGA dan Raspberry Pi berbasis mikrokomputer sebagai objek perbandingan. FPGA merupakan salah satu basis sistem yang dapat digunakan dengan keunggulan kecepatan pengolahan karena desain diimplementasikan hingga tataran perangkat keras (Zuhdy, 2014). Sedangkan Raspberry Pi yang memilki ukuran sangat kecil, keunggulan dalam flexibilitas serta penggunaan energy (Najihi, 2016).

Pada penelitian ini akan diuji waktu pengolahan citra pada kedua sistem, yaitu pada FPGA dan Mikrokomputer. FPGA yang digunakan adalah FPGA NI myRIO sedangkan Mikrokomputer yang digunakan adalah Raspberry Pi Model 3. . Pertama citra RGB dikonversi ke *grayscale* agar citra mudah diolah lalu dihilangkan noisenya dengan algoritma *gaussian blur.* Setelah itu citra dideteksi tepi dengan algoritma *laplacian edge* dan *sobel edge*. Pengujian dilakukan dengan mengolah tiga jenis citra berbeda pada tiga algoritma yang berbeda dan dilakukan sebanyak sepuluh kali pengujian dan diambil waktu rata-rata pengolahan citra pada kedua sistem. Lalu dibandingkan hasilnnya pada kedua sistem.

2. BLOK DIAGRAM SISTEM

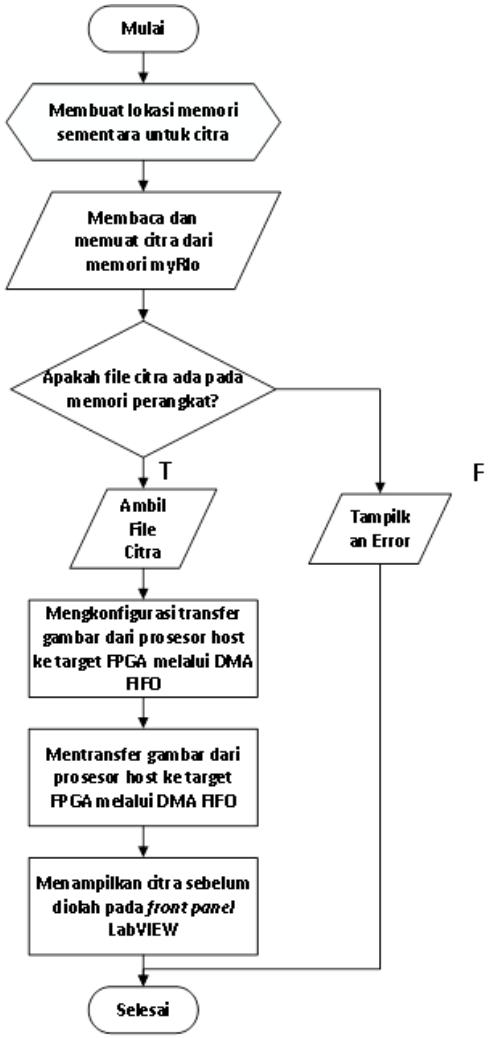
Pada Gambar 1 dijelaskan cara kerja sistem dengan blok diagram. Perancangan pada sistem ini dibagi menjadi perancangan mengambil dan menampilkan citra sebelum diolah, perancagan mengolah citra, perancagan mengambil dan menampilkan citra setelah diolah dan perancangan menghitung waktu pengolahan citra.Perancangan perangkat lunak pada FPGA menggunakan aplikasi LabVIEW sedangkan perancangan pada Mikrokomputer menggunakan OpenCV.

Gambar 1. Blok Diagram Sistem

3. PERANCANGAN SISTEM

3.1 Perancangan Mengambil dan Menampilkan Citra Sebelum Diolah

Pada perancangan ini, sistem dapat mengambil file citra dari memori perangkat yang akan melakukan pengolahan citra. Selain dapat mengambil file citra, sistem juga dapat menampilkan citra yang akan diolah dan dapat dilihat oleh *user*. Perancangan sistem pada FPGA menggunakan beberapa VI dari LabVIEW, VI yang pertama adalah VI *IMAQ Create* yang berfungsi membuat lokasi memori sementara untuk citra. Selanjutnya adalah VI *IMAQ ReadFile* yang berfungsi membaca dan memuat citra dari memory myRIO, VI *IMAQ FPGA Configure Image Transfer to Target* yang berfungsi mengkonfigurasi transfer gambar dari prosesor host ker target FPGA melalui DMA FIFO dan yang terakhir adalah VI I*MAQ FPGA Image Transfer to Target* yang berfungsi mentrasnfer gambar dari prosesor host ke target FPGA melalui DMA FIFO. Setelah semua VI ini tekoneksi dengan benar, maka citra sebelum diolah dapat ditampilkan pada *front panel* dari aplikasi LabVIEW. Diagram alir dari perancagan mengambil dan menampilkan citra sebelum diolah pada FPGA bisa dilihat pada Gambar 2 berikut



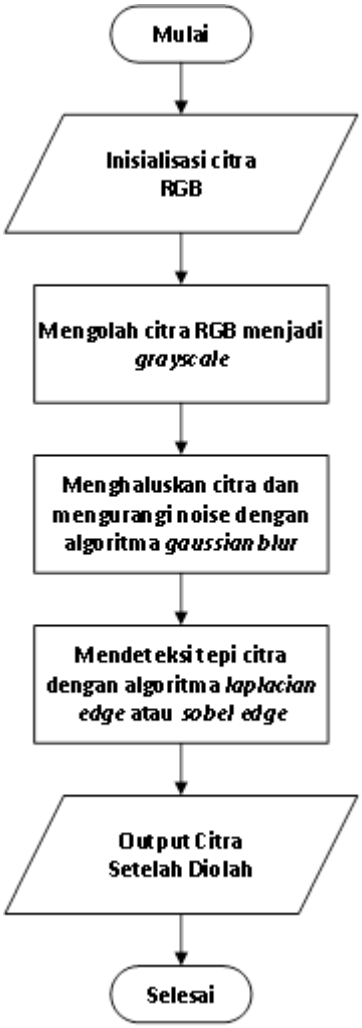
Gambar 2. Diagram Alir Perancangan Mengambil dan Menampilkan Citra Sebelum Diolah pada FPGA

Perancangan sistem pada Mikrokomputer,, pertama diinisialisasi dulu variablenya dan menggunakan fungsi *cv2.imread()* yang tersedia dan kita deklarasikan nama file untuk citra yang akan diolah. File citra yang akan diolah diletakkan satu folder dengan kode program agar memudahkan untuk eksekusi. Untuk menampilkan citra sebelum diolah, digunakan fungsi *cv2.imshow()* yang tersedia pada Library OpenCV. .

3.2 Perancangan Mengolah Citra

Pada perancagan ini sistem menginisialiasi citra RGB yang telah dipilih *user,* lalu citra RGB tersebut diolah menjadi citra *grayscale.* Setelah itu citra dihaluskan dan dikurangi noisenya dengan algoritma *gaussian blur* lalu citra didteksi tepinya dengan algoritma *laplacian edge* atau *sobel edge.* Perancangan mengolah citra pada FPGA menggunakan fungsi *Color Plane Extraction* yang ada pada *Library Vision*

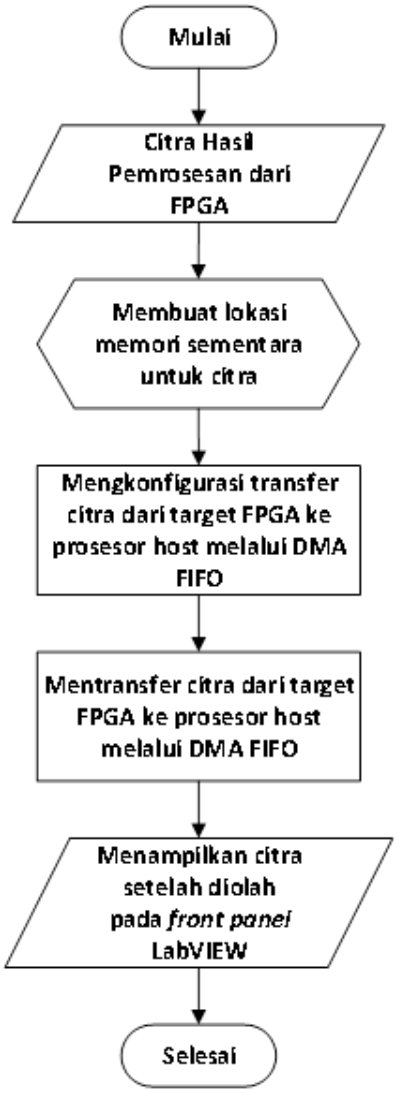
Assistant yang berfungsi mengekstrak salah satu dari tiga bidang warna *(Red, Green, Blue, Hue, Saturation, Luminance, Value, Intensity)* dari sebuah citra. Disini penulis *Luminance Plane* untuk mengolah citra RGB menjadi *grayscale*. Setelah citra diolah menjadi *grayscale*, citra akan di *smoothing* dengan fungsi *Filters* di *Vision Assistant* dengan algoritma *Gaussian Blur* agar citra lebih halus dan lebih mudah untuk mendeteksi tepi. Selanjutnya citra akan diolah dengan algoritma *Sobel Edge* atau *Laplacian Edge* dan terdeteksi tepinya.



Gambar 3. Diagram Alir Perancangan Mengolah Citra

Perancangan mengolah citra pada Mikrokomputer menggunakan beberapa *Library* dari OpenCV. Pertama citra RGB diolah menjadi citra *grayscale* dengan fungsi *cv2.COLOR_RGB2GRAY* yang ada di Library OpenCV. Setelah citra menjadi *grayscale*, citra akan dihaluskan dan dikurangi noisenya dengan fungsi *cv2.GaussianBlur()* supaya deteksi tepi menjadi lebih akurat. Lalu citra akan dideteksi tepinya dengan algoritma *laplacian edge* atau *sobel edge* menggunakan fungsi *cv2.Laplacian()* dan fungsi *cv2.Sobel()* sehingga terdeteksi tepinya. Diagram alir dari perancagan mengolah citra bisa dilihat pada Gambar 3

3.3 Perancangan Mengambil dan Menampilkan Citra Setelah Diolah



Gambar 4. Diagram Alir Perancangan Mengambil dan Menampilkan Citra Setelah Diolah pada FPGA.

Pada perancangan ini sistem dapat mengambil file citra dari memori perangkat yang telah melakukan pengolahan citra. Selain dapat mengambil file citra, sistem juga dapat menampilkan citra yang telah diolah dan dapat dilihat oleh *user*. Perancangan sistem ini pada FPGA digunakan VI dari LabVIEW, VI yang pertama adalah VI *IMAQ Create* yang berfungsi membuat lokasi memori sementara untuk citra. Selanjutnya adalah VI *IMAQ FPGA Configure Image Transfer from Target* yang berfungsi

mengkonfigurasi transfer citra dari FPGA ke prosesor host melalu DMA FIFO dan yang terakhir adalah VI I*MAQ FPGA Image Transfer from Target* yang berfungsi mentransfer citra dari FPGA ke prosesor host melalu DMA FIFO. Setelah semua VI ini tekoneksi dengan benar, maka citra sebelum diolah dapat ditampilkan pada *front panel* dari aplikasi LabVIEW. Diagram alir perancangan mengambil dan meanmpilkan citra setelah diolah pada FPGA dapat dilihat pada Gambar 4

Perancangan mengambil dan menampilkan citra setelah diolah pada Mikrokomputer adalah sebagai berikut. Untuk mengambil citra yang telah diolah, digunakan fungsi *cv2.imwrite()* untuk menulis citra yang telah kita olah ke memori Raspberry Pi. Sedangkan untuk menampilkan citra setelah diolah, digunakan fungsi *cv2.imshow()* yang tersedia pada Library OpenCV.

3.4 Perancangan Menghitung Waktu Pengolahan Citra

Pada fungsi ini sistem dapat mengambil data waktu pengolahan citra saat sistem bekerja. Waktu dihitung dari proses mengambil dan menampilkan citra sebelum diolah, proses pengolahan citra dan proses mengambil dan menampilkan citra setelah diolah. Perancangan menghitung waktu pengolahan citra pada FPGA menggunakan aplikasiLabVIEW, digunakan *Flat Sequence Structure* yang berfungsi untuk memastikan subdiagram berfungsi secara berurutan dari diagram kiri ke diagram kanan. Setelah *Flat Sequence Structure* terbentuk, digunakan fungsi *Tick Count* yang berfungsi sebagai *timer dan* menghitung nilai waktu dalam satuan *millisecond.*

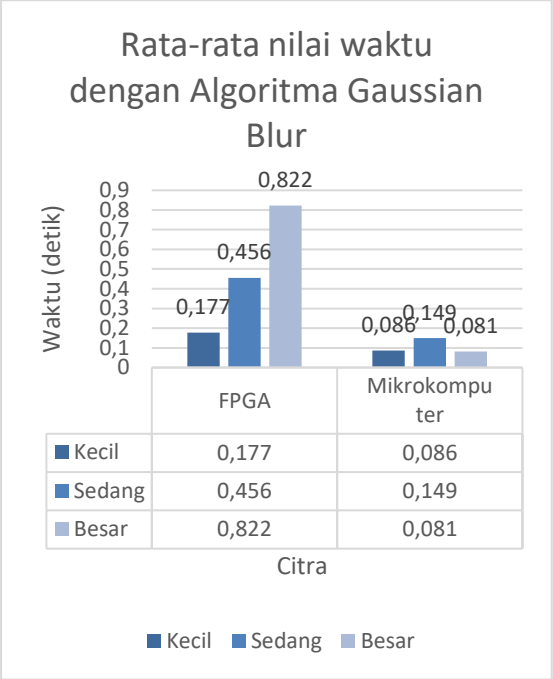
Pada perancangan menghitung waktu pengolahan citra di Mikrokomputer, perncangan menghitung waktu menggunakan *Library* OpenCV, digunakan fungsi yang telah tersedia pada OpenCV yaitu *cv2.getTickCount()* dan *cv2.getTickFrequency()*.Fungsi*cv2.getTickCount()* digunakan di awal dan akhir kode program sebagai penanda awal waktu awal dan akhir program di eksekusi. Lalu di akhir program waktu akhir dikurangi waktu awal dan dibagi oleh fungsi *cv2.getTickFrequency()* sehingga dapat dihasilkan waktu eksekusi kode program.

4. PENGUJIAN DAN HASIL

Untuk menguji performa dari sistem ini, maka akan dilakukan pengujian waktu proses pengolahan citra pada kedua sistem. Pengujian akan dilakukan dengan tiga buah citra RGB yang masing-masing citra rusa berformat .jpg yang bereukuran besar 1920x1080, berukuran sedang1366x766 dan berukuran kecil 800x449. Pengujian akan dilakukan sebanyak sepuluh kali dan diambil waktu proses nya untuk di rata-rata.

4.1. Hasil Pengujian dengan Algoritma *Gaussian Blur*

Pengujian pada kedua sistem berhasil mengolah citra dengan algoritma *gaussian blur*. Grafik nilai rata-rata waktu proses pengolahan citra pada kedua sistem dapat dilihat pada Gambar 5 berikut ini.

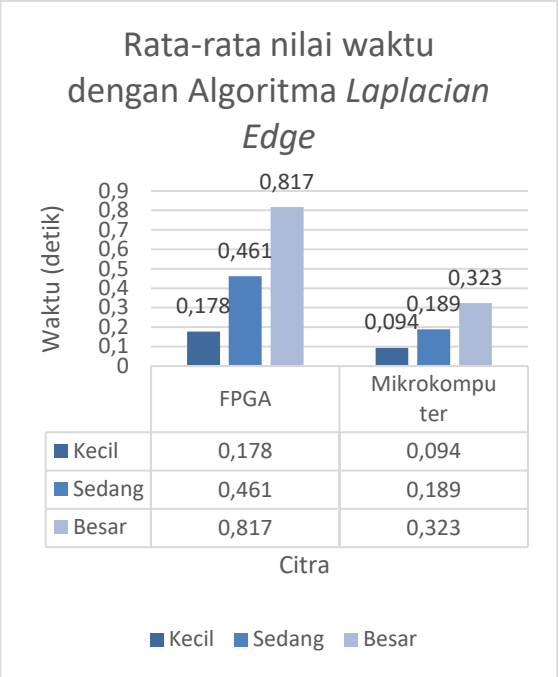


Gambar 5. Grafik Perbandingan Nilai Waktu Pengolahan Citra dengan Algoritma *Gaussian Blur*

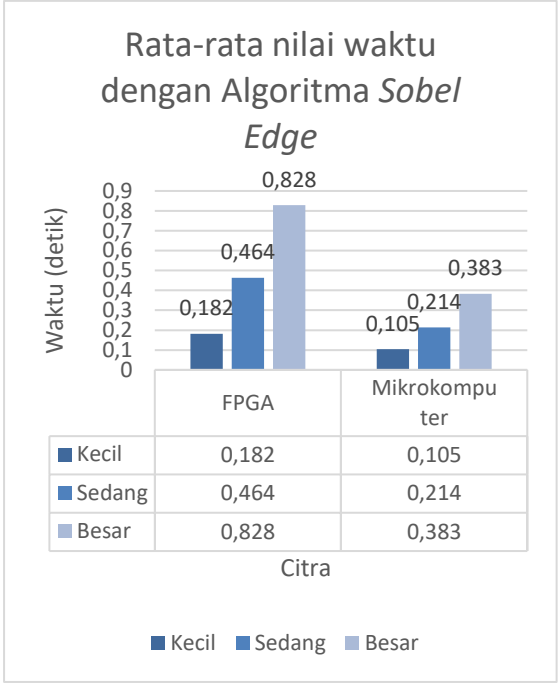
Hasil pengujian waktu menunjukkan, waktu pengolahan pada FPGA lebih cepat pada citra crayon dibandingkan dengan waktu pada Mikrokomputer. Sedangkan untuk citra wajah dan rumah, mikrokomputer lebih cepat waktu proses pengolahannya dibandingkan FPGA.

4.2. Hasil Pengujian dengan Algoritma *Laplacian Edge*

Pengujian pada kedua sistem berhasil mengolah citra dengan algoritma *laplacian edge*. Grafik nilai rata-rata waktu proses pengolahan citra pada kedua sistem dapat dilihat pada Gambar 6 berikut ini



Gambar 6. Grafik Perbandingan Nilai Waktu Pengolahan Citra dengan Algoritma *Laplacian Edge*



Gambar 7. Grafik Perbandingan Nilai Waktu Pengolahan Citra dengan Algoritma *Sobel Edge*

Hasil pengujian waktu menunjukkan, waktu pengolahan pada FPGA lebih cepat pada citra crayon dibandingkan dengan waktu pada Mikrokomputer. Sedangkan untuk citra wajah dan rumah, mikrokomputer lebih cepat waktu proses pengolahannya dibandingkan FPGA. Jika dibandingkan dengan pengujian dengan algoritma *gaussian blur*, nilai waktu pada FPGA dan mikrokomputer tidak jauh berbeda pada kedua algoritma.

4.3 Hasil Pengujian dengan Algoritma *Sobel Edge*

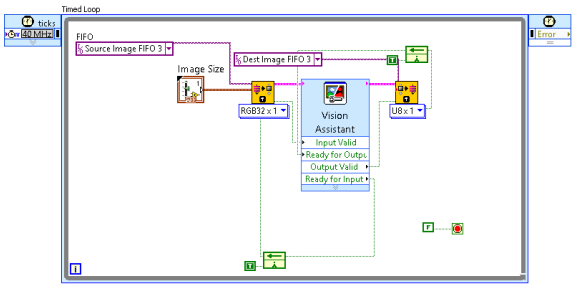
Pengujian pada kedua sistem berhasil mengolah citra dengan algoritma *sobel edge*. Grafik nilai rata-rata waktu proses pengolahan citra pada kedua sistem dapat dilihat pada Gambar 7.

Hasil pengujian waktu menunjukkan, waktu pengolahan pada FPGA lebih cepat pada citra crayon dibandingkan dengan waktu pada Mikrokomputer. Sedangkan untuk citra wajah dan rumah, mikrokomputer lebih cepat waktu proses pengolahannya dibandingkan FPGA. Jika dibandingkan dengan pengujian dengan algoritma *gaussian blur*, nilai waktu pada FPGA tidak berbeda jauh pada kedua algoritma dan nilai waktu pada mikrokomputer mengalami peningkatan dengan algoritma *gaussian blur* dan *laplacian edge*.

5. ANALISIS HASIL PENGUJIAN

Berdasarkan hasil pengujian dapat dikatakan sistem dapat dengan baik mengolah citra, namun ada perbedaan nilai waktu pada kedua *platform* yaitu di FPGA maupun Mikrokomputer. Hasil waktu rata-rata pengolahan citra dengan algoritma *Gaussian blur* adalah 0.485s pada FPGA dan 0.165s pada Mikrokomputer. Untuk algortitma *Laplacian edge* waktu rata-rata adalah 0.492s pada FPGA dan 0.202s pada Mikrokomputer sedangkan untuk algoritma *Sobel edge* waktu rata-rata nya adalah 0.498s pada FPGA dan 0.234s pada Mikrokomputer.

Hasil pengujian membutikkan, bahwa Mikrokomputer lebih unggul dalam kecepatan keseluruhan pengolahan citra dibandingkan FPGA pada beberapa algoritma dan jenis citra. Hal ini disebabkan karena pada blok diagram LabVIEW, yang dihitung adalah kecepatan transfer data citra dari Host ke FPGA sebelum citra diolah dan transfer data citra dari FPGA ke Host setelah citra diolah. Bisa dilihat dari data nilai rata-rata waktu FPGA untuk citra yang sama dengan algoritma yang berbeda, waktunya juga tidak jauh berbeda. Untuk waktu pengolahan citra, pada blok diagram LabVIEW di sisi VI FPGA digunakan *Single Cycle Timed Loop*. Pada Gambar 8 adalah blok diagram fungsi pengolahan citra yang menggunakan *Single Cycle Timed Loop*.



Gambar 8. *Single Cycle Timed Loop* pada Fungsi Pengolahan Citra

Single Cycle Timed Loop ini berfungsi untuk memastikan setiap fungsi yang ada di dalam loop ini tereksekusi selama satu *cycle* pada *clock rate* yang kita tentukan.Didalam Loop ini terdapat blok diagram dari VI *Vision Assistant*, yang dimana jika di *compile* maka akan otomatis memparalellisasi code dihasilkan. MyRIO memiliki *clock rate* bawaan sebesar 40 MHz yang berarti memiliki clock cycle sebesar 25 ns. Sedangkan kecepatan mengolah citra pada FPGA berbeda bergantung pada citra yang akan diolah, kecepetan mengolah citra pada FPGA bisa di estimasi dengan Tools yang ada di VI *Vision Assistant* yaitu tools *Perfomance Meter*.

Hasil estimasi waktu pengolahan citra pada FPGA menunjukan, waktu pengolahan pada citra berukuran kecil adalah 0.01053 detik, citra berukuran sedang adalah 0.03074 detik dan citra berkuran besar adalah 0.06076 detik. Hasil kecepatan ini dipengaruhi oleh karakteristik dari FPGA sendiri yang memiliki kemampuan pemrosesan parallel dan logika yang dibuat bekerja secara kombinatorial pada level perangkat keras.

Sedangkan hasil pengujian pada Mikrokomputer Raspberry Pi menunjukan waktu pengolahan yang lebih cepat dibandingkan FPGA. Namun pengolahan citra pada Raspberry Pi dilakukan secara sekuensial pada level perangkat lunak, tidak seperti pengolahan citra pada FPGA yang dilakukan secara parallel pada level perangkat keras. Ini dibuktikan dengan hasil pengujian pada Raspberry Pi, algoritma *Gaussian Blur* memiliki waktu pengolahan lebih cepat dibandingkan algoritma *Laplacian Edge* dan *Sobel Edge* karena algoritma *Gaussian Blur* lebih sederhana dibandingkan kedua algoritma lainnya. Begitu juga dengan hasil dari pengolahan citra dengan algoritma *Laplacian Edge* lebih cepat daripada algoritma *Sobel Edge*, ini disebabkan algoritma *Sobel Edge* memiliki aritmatika lebih komples dalam perhitungannya dibandingkan dengan algoritma *Laplacian Edge.*

6. KESIMPULAN

Berdasarkan hasil perancangan sistem, hasil pengujian sistem dan analisis penilitian dapat disimpulkan sebagai berikut. Sistem pengolahan citra dapat dirancang dan di implementasikan dengan algoritma *Gaussian Blur, Laplacian Edge* dan *Sobel Edge* pada platform FPGA dapat di rancang menggunakan aplikasi LabVIEW dan Library dari NI Vision Assistant, sedangkan pada platform FPGA algoritma tersebut dapat dirancang dengan OS Raspbian dan Library dari OpenCV.

Pada pengujian nilai waktu pengolahan citra, rata-rata waktu pada tiga algoritma menggunakan tiga citra yang berbeda selama 10 kali dengan algoritma *Gaussian blur* adalah 0.485s pada FPGA dan 0.165s pada Mikrokomputer. Untuk algortitma *Laplacian edge* waktu rata-rata adalah 0.492s pada FPGA dan 0.202s pada Mikrokomputer sedangkan untuk algoritma *Sobel edge* waktu rata-rata nya adalah 0.498s pada FPGA dan 0.234s pada Mikrokomputer. Namun sebenarnya untuk semua algoritma, waktu FPGA tetap sama namun berbeda pada tiap citra berukuran kecil, sedang dan besar masing-masing, adalah 0.01053 detik, 0.03074 detik dan 0.06076 detik.

Berdasarkan hasil pengujian diatas dapat diambil kesimpulan bahwa waktu akuisisi citra sebelum dan setelah citra diolah di Raspberry Pi memang lebih unggul dibandingkan FPGA, namun untuk waktu pengolahan citra FPGA jauh lebih unggul dikarenakan karakteristik FPGA yang mampu bekerja secaral parallel dan logika yang dibuat bekerja secara kombinatorial pada level perangkat keras

DAFTAR PUSTAKA

Ahmad, U. (2005). *Pengolahan Citra Digital & Teknik Pemrogramannya.* Bogor: Graha Ilmu.

Kiran, M., War, K. M., Kuan, L. M., Meng, L. K., & Kin, L. W. (2008). Implementing image processing algorithms using ‘Hardware in the loop’ approach for Xilinx FPGA. *International Conference on Electronic Design*.

Mulya, M., & Abdiansah. (2013). Penerapan Multi-threading untuk Meningkatkan Kinerja Pengolahan Citra Digital.

Universitas Sriwijaya.

Najihi, A. (2016). ANALISIS KINERJA IP PBX SERVER PADA SINGLE BOARD CIRCUIT RASPBERRY PI. *Universitas Muhammadiyah Palangkaraya*.

Sholihin, R. A., & Purwoto, B. H. (2015). PERBAIKAN CITRA DENGAN MENGGUNAKAN MEDIAN FILTER dan METODE HISTOGRAM EQUALIZATION. *Universitas Muhammadiyah Surakarta*.

Zuhdy, A. H. (2014). Implementasi Programmable DAC pada FPGA Xilink Spartan-6 Berbasis VHDL. *Universitas Gajah Mada*.