PERANCANGAN DETEKSI KEPADATAN LALU LINTAS MELALUI KAMERA IP MENGGUNAKAN ALGORITMA SAD (SUM OF ABSOLUTE DIFFERENCES)

Traffic Density Monitoring and Prediction by Using IP Camera and SAD (Sum of Absolute Differences) Algorithm

Yudhi Septian Rahman[1], R. Rumani M, Drs., Ir., BcTT., MSEE.[2], Casi Setianingsih S.T., MT.[3]
[1,2,3]Prodi S1 Sistem Komputer, Fakultas Teknik, Universitas Telkom

[1]septianrahman@telkomuniversity.ac.id, [2]rumani@telkomuniveristy.co.id,
[3]setiacasi@telkomuniversity.ac.id

Abstrak

Pada saat ini, perkembangan di bidang industri otomotif tengah berkembang dengan cepat, dengan mengandalkan teknologi robotika, maka manufaktur kendaraan bermotor semakin cepat dan mudah. Hal ini membuat jumlah kendaraan bermotor semakin meningkat dengan harga yang terjangkau, dampak lainnya adalah masyarakat semakin gemar membeli kendaraan pribadi yang menyebabkan *volume* kendaraan meningkat dengan pesat dan menimbulkan kepadatan pada ruas jalan di lalu lintas.

Dalam mengatasi kepadatan lalu lintas, petugas kepolisian lalu lintas diterjunkan untuk mengendalikan situasi lalu lintas dengan melakukan rekayasa lalu lintas atau *override* pada lampu lalu lintas. Namun jika kondisi lalu lintas tengah berada pada waktu tertentu yang menyebabkan *volume* kendaraan tidak terkendali, dibutuhkan cara lain untuk mengurai kepadatan lalu lintas. Pada Tugas Akhir ini, suatu kamera yang dapat memantau jumlah kendaraan dengan cara citra video diolah menggunakan algoritma SAD yang bekerja dengan cara membandingkan nilai *distance* kedua *frame* referensi dan *frame* saat tertentu agar dapat menghasilkan pernyataan kondisi lalu lintas. Dengan mengkombinasikan algoritma SAD dan perhitungan *White Pixels* maka dapat melakukan deteksi kendaraan dan menghitung jumlah kendaran dalam suatu ruas jalan pada lalu lintas.

Kata Kunci : *volume, override*, distance, algoritma *SAD, frame, White Pixels*

Abstract

In the development of the Automotive Industries nowaday, people begin relying on robotic technology for vehicles manufacturing, so it makes fast and easy for creating vehicles. This phenomenon makes the number of vehicles is increasing but comes with affordable prices. Another impact from this phenomenon is people are begin to buy vehicles for personal use which cause the volume of traffic increased rapidly and traffic density are everywhere.
To overcome this situation, traffic police officers were deployed to control the traffic situation by performing override the traffic lights. But if the traffic conditions at several certain time causes uncontrolled volume of vehicles. In this research, we tried to place a camera that can monitoring the number of vehicles by using SAD (Sum of Absolute Differences) algorithm. It works by comparing the value between frames in order to generate a statement of a traffic. By combining the SAD algorithm and calculating White Pixels, we can detect vehicles and calculate the number of vehicles in a road section in the traffic.

Keywords: *volume*, *override*, distance, *SAD algorithm, frame, White Pixels*

1. Pendahuluan

Pada perkembangan dunia industri otomotif saat ini, kendaraan bermotor semakin mudah dijangkau oleh semua kalangan masyarakat. Hal ini menyebabkan meningkatnya *volume* kendaraan yang menimbulkan kepadatan lalu lintas. Untuk mengurai kepadatan lalu lintas. Untuk mengurai kepadatan lalu lintas tersebut, petugas kepolisian lalu lintas tidak jarang melakukan *override* pada lampu lalu lintas, yaitu mengubah lampu lalu lintas secara *manual*, atau melakukan rekayasa lalu lintas untuk mencapai efisiensi pada lalu lintas.

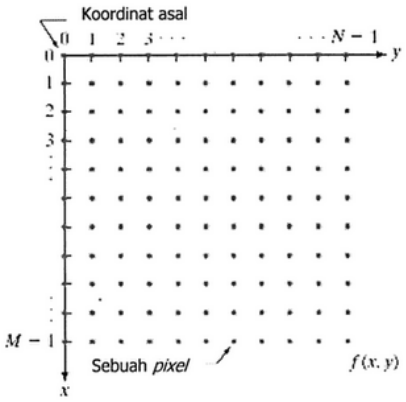
Oleh karena itu, diperlukan koordinasi lalu lintas yang lebih efektif agar ketika kepadatan lalu lintas terjadi, dapat ditangani dengan tepat. Saat ini, salah satu langkah untuk memantau lalu lintas adalah dengan memasang CCTV (*Closed-circuit Television*) yang dikelola oleh TMC (*Traffic Management Center*) Polda. Namun saat ini yang dapat diketahui adalah pemantauan lalu lintas hanya sebatas *live streaming*.

2. Dasar Teori

2.1 Citra Digital

Pengolahan citra digital [1] secara umum adalah pemrosesan gambar dua dimensi menggunakan komputer. Dalam konteks yang lebih luas, pengolahan citra digital mengacu pada pemrosesan setiap data dua dimensi. Citra digital merupakan sebuah larik (*array*) yang berisi nilai-nilai *real* maupun kompleks yang direpresentasikan dengan deretan bit tertentu.

Suatu citra dapat didefinisikan sebagai fungsi f*(x,y)* berukuran M baris dan N kolom, dengan x dan y adalah koordinat spasial, dan *amplitudo* f di titik koordinat (*x,y)* dinamakan intensitas atau tingkat keabuan dari citra pada titik tersebut. Apabila nilai x, y, dan nilai *amplitudo* f secara keseluruhan berhingga (*finite*) dan bernilai diskrit maka dapat dikatakan bahwa citra tersebut adalah citra digital.



Gambar 2.1 *Koordinat citra digital* [1]

Citra digital dapat ditulis dalam bentuk matriks sebagai berikut:

$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \quad \text{..... (2.1)}$$

Gambar 2.2 *Matriks citra digital* [1]

Nilai pada suatu irisan antara baris dan kolom (pada posisi *x,y*) disebut dengan *picture elements*, *image elements*, *pels*, atau *pixels*. Istilah *pixel* paling umum digunakan pada citra digital.

2.2. Jenis Citra

Nilai suatu *pixel* [1] memiliki nilai dalam rentang tertentu, dari nilai minimum sampai nilai maksimum. Jangkauan yang digunakan berbeda-beda tergantung dari jenis warnanya. Namun secara umum jangkauannya adalah 0 - 255. Citra dengan penggambaran seperti ini digolongkan ke dalam citra *integer*. Berikut adalah jenis-jenis citra berdasarkan nilai pixelnya.

2.2.1 Fragment

Citra *biner* [1] adalah citra digital yang hanya memiliki dua kemungkinan nilai *pixel* yaitu hitam dan putih. Citra *biner* juga disebut sebagai citra B&W (*Black and White*) atau citra monokrom. Hanya membutuhkan 1-bit untuk mewakili nilai setiap *pixel* dari citra *biner*.

2.2.2 Citra *Grayscale*

Citra *Grayscale* [1] merupakan citra digital yang hanya memiliki satu nilai kanal pada setiap pixelnya, dengan kata lain nilai bagian RED = GREEN = BLUE. Nilai tersebut digunakan untuk menunjukan tingkat intensitas. Warna yang dimiliki adalah warna dari hitam, keabuan, dan putih. Tingkatan keabuan disini merupakan warna abu dengan berbagai tingkatan dari hitam hingga mendekati warna putih. Citra *Grayscale*.

2.2.3 Citra Warna (8-Bit)

Setiap *pixel* dari citra warna (8-bit) hanya diwakili oleh 8-bit dengan jumlah warna maksimum yang dapat digunakan adalah 256 warna. Ada dua jenis citra warna 8-bit. Pertama, citra warna 8-bit dengan menggunakan palet warna 256 dengan setiap paletnya memiliki pemetaan nilai (*colormap*) RGB tertentu [1]. Model ini lebih sering digunakan. Kedua, setiap *pixel* memiliki format 8-bit sebagai berikut:

Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0
R	R	R	G	G	G	B	B

Gambar 2.3 *Deret warna 8-bit* [1]

2.2.4 Citra Warna (16-Bit)

Citra warna 16-bit (biasanya disebut sebagai citra *highcolor*) dengan setiap pixelnya diwakili dengan 2 *byte memory* (16-bit) [1]. Warna 16-bit memiliki 65.536 warna. Dalam formasi bitnya, nilai merah dan biru mengambil tempat di 5 bit di kanan dan kiri. Komponen hijau memiliki 5-bit ditambah 1-bit ekstra. Pemilihan komponen hijau dengan deret 6-bit dikarenakan penghilatan manusia lebih sensititf terhadap warna hijau.

Bit-15	Bit-14	Bit-13	Bit-12	Bit-11	Bit-10	Bit-9	Bit-8	Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0
R	R	R	R	R	G	G	G	G	G	G	B	B	B	B	B

Gambar 2.4 *Deret warna 16-bit* [1]

2.3 *Image Processing*

Image Processing [3] adalah teknik untuk menambah kemampuan dari *raw images* yang diterima melalui perangkat kamera, satelit, sensor untuk berbagai kebutuhan. Beragam teknik telah dikembangkan pada *Image Processing*, hampir seluruhnya dikembangkan untuk keperluan sains dan teknologi seperti pengambilan citra melalui pesawat tanpa awak. *Image Processing* menjadi populer karena terdapat banyak pihak yang mengembangkannya, konsumsi *memory* yang rendah, piranti lunak *Image Processing.*

2.4 *Image Segmentation*

Image Segmentation [3] adalah proses yang membagi-bagi sebuah citra menjadi bagian bagian penyusunnya atau menjadi objek. Bagian atau objek tersebut memiliki *level* yang tergantung terhadap operasi yang dilakukan, sebagai contoh yaitu proses segmentasi berhenti ketika objek yang dicari telah ditemukan, contoh lainnya adalah kasus pencarian kendaraan dilakukan pada citra sebuah jalan, tahap pertama segmentasi adalah mencari objek sebuah jalan pada citra tersebut dan mencari objek kendaraan didalam objek sebuah jalan tersebut.

2.5 Algoritma SAD (*Sum of Absolute Differences*)

Dalam sebuah jurnal ilmiah [4] Algoritma SAD (*Sum of Absolute Differences*) adalah algoritma yang berdasarkan *pixel* dari sebuah objek untuk mencari perbedaan. SAD adalah algoritma untuk mengukur tingkat

kesamaan blok antara dua citra. SAD bekerja dengan mengambil nilai *absolute difference* antara *pixel* citra referensi dan citra yang akan dijadikan objek perbandingan.

Pada citra video, algoritma SAD bekerja pertama dengan menentukan citra referensi atau *background*. Kemudian melakukan proses operasi *subtraction*, dan terakhir adalah menentukan batas (*threshold*). *Threshold* merupakan penentu untuk pendeteksian gerak. Algoritma SAD dapat direpresentasikan [4] melalui persamaan berikut:

$$SAD = \sum_{i=1} n \sum_{j=1} n \, |\mathrm{I} - \mathrm{J}|$$

Dimana i adalah *frame i*, j adalah *frame* j, dengan n adalah jumlah *pixel* pada *frame*.

2.6 Video Digital

Dalam penjelasan [2] Video adalah teknologi menangkap, merekam, memproses, menyimpan, dan merekonstruksi suatu urutan dari beberapa gambar. Video pertama kali dikembangkan untuk sistem televisi CRT (*cathode ray tube*). Video digital pada dasarnya tersusun atas serangkaian *frame* yang ditampilkan dengan kecepatan tertentu (*frame*/detik) dengan satuan fps (*frame per second*). Jika laju *frame* cukup tinggi, maka mata manusia melihatnya sebagai rangkaian yang kontinu.

Video digital memiliki karakteristik diantaranya adalah *pixel* yang menentukan dimensi atau resolusi, karakteristik lainnya adalah kuantisasi atau kedalaman bit dan *frame rate*.

2.7 *Thresholding*

Thresholding [5] adalah metode yang digunakan untuk mengatur jumlah derajat keabuan yang ada pada citra. Dengan menggunakan *thresholding* maka derajat keabuan bisa diubah sesuai keinginan, misalkan diinginkan menggunakan derajat keabuan 16, maka selanjutnya adalah membagi nilai derajat keabuan dengan 16. Proses *thresholding* ini pada dasarnya adalah proses pengubahan kuantisasi pada citra, sehingga untuk melakukan *thresholding* dengan derajat keabuan dapat digunakan rumus:

$$x = \frac{w}{b}$$

Dimana :

w adalah nilai derajat keabuan sebelum *thresholding*.

b adalah jumlah derajat keabuan yang diinginkan.

x adalah nilai derajat keabuan setelah *thresholding*.

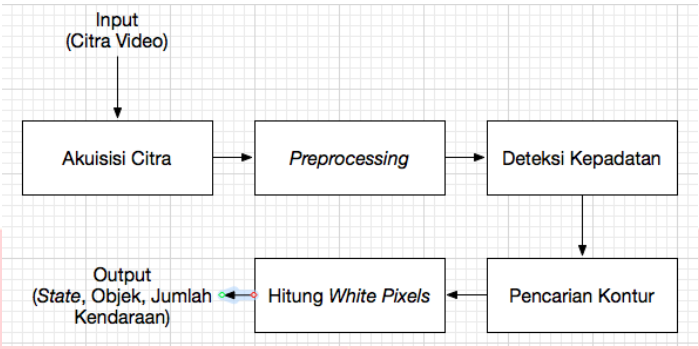
Pada *thresholding* yang tinggi tidak tampak adanya perbedaan karena keterbatasan indra penglihatan manusia, tetapi untuk *thresholding* tingkat rendah seperti 2, 4, 8, 16 dapat terlihat cukup jelas perbedaanya.

3. Perancangan Sistem

3.1 Gambaran Umum Perancangan Sistem

Sistem yang dibuat pada tugas akhir ini adalah sistem yang dapat mendeteksi kepadatan lalu lintas di sebuah persimpangan, cara kerja sistem ini terdiri dari beberapa tahap. Tahap pertama sistem akan merekam keadaan ruas jalan, kemudian hasil rekaman tersebut diolah oleh sistem untuk mengasilkan pernyataan dari kondisi ruas jalan tersebut. Tahap berikutnya adalah mencari kontur dari rekaman tersebut, rekaman berupa video yang akan dibagi menjadi *frame*. Terakhir adalah menghitung jumlah *pixel* dari suatu *region of interest* untuk menghitung kendaraan yang melintas pada ruas jalan.

Sistem dirancang menggunakan bahasa pemrograman C++ dengan memanfaatkan *library* dari OpenCV untuk kebutuhan *image processing*. Berikut adalah alur pada sistem yang dibuat oleh penulis untuk menggambarkan cara sistem bekerja dalam gambaran umum.



Gambar 3.1 *Alur sistem deteksi kepadatan kendaraan*

3.2 Analisa Kebutuhan Sistem

Sistem yang dirancang pada tugas akhir ini terdapat proses pada perangkat lunak, proses tersebut diantaranya adalah menerima *input* berupa citra video, melakukan segmentasi pada citra video kemudian melakukan *preprocessing* atau *enhancement* pada citra video yang telah disegmentasi menjadi *frame*. Proses selanjutnya adalah proses deteksi, pencarian, dan perhitungan terhadap *frame*. Pada saat sistem bekerja, maka sistem memiliki kebutuhan dengan ketentuan seperti berikut:

3.2.1 Kebutuhan Perangkat Lunak

Dalam prosesnya, sistem membutuhkan beberapa fungsi, diataranya adalah :

1. Mengambil *input* video dari kamera IP.
2. Membuat citra referensi berdasarkan *input* yang telah diakuisisi.
3. Melakukan *enhancement* dan segmentasi pada *input* video.
4. Melakukan proses *converting* warna pada *frame*.
5. Melakukan perhitungan *absolute difference*.
6. Melakukan perhitungan dengan fungsi *countNonZero*.
7. Menentukan nilai *threshold* berdasarkan hasil *absolute difference*.
8. Menentukan kondisi ruas jalan berdasarkan hasil *absolute difference*.
9. Menghitung kendaraan yang melintas berdasarkan hasil fungsi *countNonZero*.

3.3 *Input* Citra Video

Untuk melakukan deteksi kepadatan lalu lintas, maka dibutuhkan citra video dari ruas jalan sebagai parameter proses deteksi kepadatan dan operasi lainnya. Citra video diperoleh dengan memanggil fungsi *VideoCapture* dari class *VideoCapture* dari library OpenCV. Fungsi *VideoCapture* memiliki parameter sebagai *input* video yang dapat berupa *file* video dengan format *mp4* atau *avi*, atau dengan *streaming* melalui alamat IP. *VideoCapture* akan berjalan otomatis saat sistem pertama diaktifkan melalui *command prompt* dengan memanggil *file executable* sistem dan menyisipkan argumen untuk parameter *VideoCapture*.

3.4 *Preprocessing*

Preprocessing adalah tahap awal untuk *image processing*, untuk melakukan tahap *Preprocessing* maka yang perlu diketahui sebelumnya adalah sebuah *frame* dengan format warna RGB akan dirubah menjadi *grayscale*. Konversi warna pada *frame* ini bertujuan untuk kebutuhan algoritma *image processing* yang hanya beroperasi pada 1 matriks tunggal sehingga format warna RGB perlu dilakukan proses konversi.

3.5 Deteksi Kepadatan

Deteksi Kepadatan adalah proses untuk mendeteksi kepadatan pada ruas jalan dengan menggunakan metode *Absolute Difference*, yaitu melakukan pengurangan nilai pixel antara *frame* aktual dengan frame referensi. Hasil dari pengurangan tersebut adalah nilai *distance* pada frame aktual, untuk menentukan kondisi ruas jalan, pertama menetapkan dahulu nilai batas ambang atau *threshold* antara kondisi ruas jalan padat dan lancar.

Nilai *distance* dihitung kembali untuk membuat nilai rata-rata dengan cara membagi nilai *distance* dengan waktu *frame* aktual. Nilai hasil pembagian tersebut akan digunakan sebagai nilai *threshold*.

3.6 Hitung *White Pixels*

Hitung *White Pixels* adalah tahap pencarian *pixel* pada *frame* yang telah melewati proses *thresholding*. Pada metode berguna untuk menghitung jumlah kendaraan yang melintas, metode ini bekerja dengan beberapa tahap yaitu pertama membuat garis imajiner deteksi *white pixels*, kemudian membuat kondisional pada garis tersebut dan terakhir adalah membuat *statement* kendaraan dihitung.

Pembuatan garis imajiner dapat dilakukan dengan membuat *rectangular* pada *window* citra video. Kondisional dilakukan dengan membuat pernyataan *if*, jika pada area pencarian kendaraan memiliki jumlah nilai *white pixels* diatas nilai *threshold* deteksi *white pixels* maka jumlah kendaraan yang terhitung akan ditambah.

4.1 Pengujian video

Pengujian dilakukan dengan durasi selama 4 menit dengan kondisi ruas jalan dinyatakan setiap 5 detik dengan resolusi video 320 x 240 piksel dan 15 *frame per second*. Berikut adalah tabel dari hasil pengujian sistem saat pengambilan gambar pada ruas jalan selama 5 menit dengan nilai *threshold* sebesar 282.000 piksel :

Tabel 4.1 Hasil pengujian deteksi kepadatan

No	Detik	Kondisi Aktual	Jumlah Kendaraan	*Distance*	Hasil
1	5	Lancar	2	153.098	Benar
2	10	Lancar	4	102.233	Benar
3	15	Lancar	5	323.782	Salah
4	20	Lancar	4	202.456	Benar
5	25	Lancar	4	103.905	Benar
6	30	Lancar	3	83.993	Benar
7	35	Lancar	1	11.971	Benar
8	40	Lancar	2	8.836	Benar
9	45	Lancar	2	308.641	Salah
10	50	Lancar	1	67.457	Benar
11	55	Lancar	3	354.458	Salah
12	60	Padat	2	123.362	Salah
13	65	Padat	0	104.978	Salah
14	70	Padat	0	17.277	Salah
15	75	Padat	0	279.848	Benar
16	80	Padat	0	516.584	Benar
17	85	Padat	1	657.017	Benar
18	90	Padat	0	54.190	Salah
19	95	Padat	1	423.384	Benar
20	100	Lancar	3	82.576	Benar
21	105	Lancar	3	27.433	Benar
22	110	Lancar	4	468.071	Salah
23	115	Lancar	1	60.629	Benar

No	Detik	Kondisi Aktual	Jumlah Kendaraan	*Distance*	Hasil
24	120	Lancar	2	295.057	Salah
25	125	Lancar	2	299.577	Salah
26	130	Lancar	1	446.849	Salah
27	135	Lancar	2	62.099	Benar
28	140	Lancar	1	60.704	Benar
29	145	Lancar	4	108.688	Benar
30	150	Lancar	2	511.829	Salah
31	155	Lancar	2	198.368	Benar
32	160	Lancar	4	283.169	Salah
33	165	Lancar	1	129.758	Benar
34	170	Lancar	2	509.502	Salah
35	175	Lancar	2	95.729	Benar
36	180	Padat	3	622.233	Benar
37	185	Lancar	2	233.134	Benar
38	190	Padat	1	1.027.449	Benar
39	195	Padat	3	325.194	Benar
40	200	Lancar	2	633.931	Salah
41	205	Lancar	2	703.908	Salah
42	210	Lancar	2	730.302	Salah
43	215	Padat	2	575.072	Benar
44	220	Padat	1	591.838	Benar
45	225	Padat	1	515.498	Benar
46	230	Padat	1	305.526	Benar
47	235	Padat	3	865.587	Benar
48	240	Padat	1	726.319	Benar

Pada tabel 4.1 terdapat 48 nilai *distance*, terdapat 17 nilai *distance* yang salah dan 31 nilai *distance* yang benar, maka tingkat akurasi dapat diketahui sebagai berikut :

$$Nilai\ Akurasi = \frac{31}{48}\ x\ 100\% = \ 64.6\%$$

5. Kesimpulan dan Saran

5.1 Kesimpulan

Berdasarkan hasil pengujian pada perancangan sistem deteksi kepadatan lalu lintas menggunakan algoritma SAD, menghasilkan kesimpulan seperti berikut:

1. Algoritma SAD (Sum of Absolute Differences) mendeteksi gerakan berdasarkan nilai yang dihasilkan oleh perhitungan algoritma tersebut.
2. Kesalahan deteksi kondisi ruas jalan terjadi karena selisih nilai *distance* dan *threshold* yang kecil. Hal ini menyebabkan kesalahan pada sistem dalam membuat *state* untuk kondisi ruas jalan jika kondisi aktual ruas jalan adalah padat, maka sistem menyatakan ruas jalan tersebut terjadi kepadatan.

5.2 Saran

Perancangan sistem deteksi kepadatan lalu lintas dengan menggunakan algoritma SAD dapat dikembangkan lebih lanjut untuk memperoleh hasil deteksi yang lebih akurat. Saran untuk pengembangan dari perancangan sistem ini yaitu:

1. Perancangan alat menggunakan beberapa kamera dari beberapa sudut pada lalu lintas.

2. Mengambil citra pada beberapa waktu yang berbeda yaitu pagi hari, siang hari, dan malam hari.
3. Menguji ketahanan alat dengan beroperasi selama 24 jam.
4. Membuat *User Interface* agar pengguna lebih mudah mengoperasikan sistem.

Referensi

[1] Putra, Darma, Pengolahan Citra Digital, Yogyakarta: Andi Publisher, 2010.

[2] "Pengertian Video Digital", April 2013. [Online]. "http://belajartanpabuku.blogspot.co.id/2013/04/pengertian-video-digital.html". Diakses 4 April 2017.

[3] Rao, K.M.M, "Overview of Image Processing," *National Remote Sensing Centre*, 2006.

[4] Panchal, Chirag S. "Depth Estimation Analysis Using Sum of Absolute Difference Algorithm," *International Journal of Advanced Research in Electrical, Electronics and Instrumentaion Engineering*, 2014.

[5] Marvin Chandra Wijaya, Semuil Tjiharjadi, "Mencari Nilai Threshold Yang Tepat Untuk Perancangan Pendeteksi Kanker Trofoblas," *Seminar Nasional Aplikasi Teknologi Informasi 2009*, 2009.

[6] Seo Jonghoon, Chae Seungho, "Fast-Contour Tracing Algorithm Based on a Pixel-Following Method for Image

[7] "Pengolahan Video", Agustus 2014. [Online]. "http://aldotbro.blogspot.sg/2014/08/konsep-pengolahan-video.html". Diakses 4 April 2017.

[8] "Pengantar Pengolahan Citra", September 2012. [Online]. " http://vinaaryantii.blogspot.sg/2012/09/pengantar-pengolahan-citra.html". Diakses 4 April 2017.