DETEKSI KEMACETAN LALU LINTAS MELALUI KAMERA MENGGUNAKAN PIN HOLE ALGORITHM

Samuel Mahatmaputra; Erwin Permata; William

Computer Science Department, School of Computer Science, Binus University
Jl. K. H. Syahdan No. 9 Palmerah Jakarta Barat 11480
samuelmt@binus.edu

ABSTRACT

This paper describes the modifications to the Pinhole algorithm and applies it to the detection of highway congestion using an IP camera. This algorithm can detect the congestion on the highway with a fairly high degree of accuracy and the small possibility of false alarms, so that it helps reporting congestion and road users to take into account the best and quickest way. The methods used in this research are literature study, questionnaire analysis, and evaluation of application performance. The application made in this study can be further developed for combined detection with many cameras, and can support traffic regulation as well as SMS gateway service to check the condition of a road segment. At the end of this paper are presented techniques and methodology of determining the parameter values used in the calibration.

***Keywords**: pinhole algorithm, congestion detection, IP camera.*

ABSTRAK

Paper ini akan mendeskripsikan modifikasi terhadap Algoritma Pinhole serta menerapkannya pada pendeteksian kemacetan jalan raya menggunakan IP Camera. Algoritma ini dapat mendeteksi kemacetan pada jalan raya dengan tingkat akurasi yang cukup tinggi dan kemungkinan false alarm yang kecil sehingga membantu pelaporan kemacetan dan pengguna jalan dalam memperhitungkan arah jalan terbaik dan tercepat. Metode yang digunakan dalam penelitian ini adalah studi literatur, analisis kuesioner, dan evaluasi kinerja aplikasi. Aplikasi yang dibuat dalam penelitian ini dapat dikembangkan lebih lanjut untuk gabungan pendeteksian dengan banyak kamera, serta dapat membantu pengaturan lalu lintas atau layanan SMS gateway untuk memeriksa kondisi suatu ruas jalan. Pada akhir paper ini juga akan dikemukakan teknik penentuan nilai parameter serta metodologi yang digunakan dalam kalibrasi.

***Kata kunci**: algoritma pinhole, pendeteksian kemacetan, IP camera*

PENDAHULUAN

Meningkatnya kebutuhan akan manajemen transportasi yang baik telah lama dirasakan khususnya untuk kota besar seperti DKI Jakarta dengan tingkat kemacetan lalu lintas yang sangat tinggi. Salah satu langkah awal untuk tujuan tersebut adalah penyediaan sistem pemantauan yang memadai untuk memonitor keadaan ruas-ruas jalan raya serta simpul-simpul kemacetan (*traffic jam monitoring*), sehingga langkah selanjutnya pada *traffic management* dapat dilakukan. Namun untuk dapat melayani begitu banyak titik pemantauan dibutuhkan sistem otomatis yang dapat memantau keadaan ruas jalan (melalui kamera) serta secara otomatis memberikan peringatan kepada pusat maupun kepada masyarakat pengguna jalan raya bila terdeteksi adanya kemacetan lalu lintas, yaitu bila terdapat: kendaraan yang berhenti dalam waktu tertentu atau berjalan dengan sangat lambat pada ruas-ruas jalan raya. Penelitian ini akan difokuskan pada proses segmentasi setiap *frame* pada hasil visual pemantauan kedalam bagian-bagian lebih kecil yang kemudian disebut motion detection *pinhole* yang berupa mesh (m x n *pixel*s). Perubahan nilai *pixel* setiap *pinhole* di atas threshold T akan menjadi indikasi adanya gerakan. Kalibrasi terhadap nilai T yang disesuaikan dengan posisi kamera relatif terhadap obyek pemantauan serta pencahayaan juga akan menjadi parameter dalam penelitian ini. Data terakhir mengindikasikan bahwa Pemerintah Daerah DKI Jakarta telah memiliki 160 CCTV untuk memantau kemacetan dan banjir (Jakarta Post, 2008). Angka ini tentunya diharapkan akan meningkat seiring dengan pertambahan jumlah kendaraan dan ruas jalan dengan volume kendaraan yang sangat tinggi. Dengan terpantaunya volume kendaraan pada ruas-ruas jalan raya oleh para penggunanya, maka diharapkan jalur-jalur alternative dapat dimanfaatkan sehingga mengurangi tingkat kemacetan pada titik-titik tertentu.

Namun, terbatasnya kemampuan manusia (operator/pengawas kamera) untuk memonitor layar-layar pada pusat pemantauan – yaitu rata-rata 20-30 layar secara bersamaan serta mendatakan status dari ruas jalan yang dipantau merupakan salah satu kendala utama. Sebagai perbandingan London memiliki 500.000 kamera pengawas di tahun 2003(The Wall Street Journal, 2003). Bila di kota besar seperti Jakarta memiliki jumlah kamera pemantau sebanyak 10.000 buah untuk memantau kemacetan, tidak bisa dibayangkan kesibukan para operator di pusat-pusat pemantauan yang notabene pernah dialami oleh otoritas kota-kota besar dunia lainnya. Untuk mempersiapkan serta menjawab kebutuhan diatas, penelitian ini akan mengembangkan sebuah modul yang dapat menganalisis gambar yang dikirimkan oleh kamera pemantau, serta secara otomatis menyimpulkan/memberikan status apakah sedang terjadi kemacetan atau tidak pada video tersebut.

METODE

Metode yang digunakan dalam penelitian ini adalah studi literatur, analisis kuesioner, dan evaluasi kinerja aplikasi. Studi literatur yang dilakukan adalah dengan mereview berbagai artikel terkait computer vision serta paper membahas penerapan serupa Algoritma *Pinhole*, serta metoda untuk mendapatkan video (MJPEG) dari IP Camera yang telah dilakukan pada penelitian terdahulu (Mahatmaputra, 2010). Analisis kuesioner digunakan untuk menganalisis masalah yang sering dihadapai oleh pengguna serta otoritas jalan raya, dilakukan dengan cara pembagian kuesioner secara acak. Evaluasi hasil aplikasi dilakukan dengan cara mengumpulkan data-data yang diambil dari aplikasi melalui percobaan yang dilakukan untuk mendapatkan parameter yang sesuai untuk menyempurnakan hasil aplikasi. Algoritma ini dikemukakan oleh E. Atkoˇci ¯unas, R. Blake, A. Juozapaviˇcius, dan M. Kazimianec dalam jurnal mereka yang berjudul "*Image Processing in Road Traffic Analysis*" (Atkoˇci ¯unas et al, 2005). Pada teori ini, dijelaskan cara kerja dari algoritma ini adalah sebagai berikut. Pertama sebuah gambar dibagi menjadi *frame-frame* kecil dengan ukuran tertentu yang sama besar (contoh 4x4 *pixel*). Kemudian dari *frame* tersebut diambil rata-rata dari *pixel*

dalam format abu-abu/*grayscale*. Rata-rata *pixel* tersebut dibandingkan dengan rata-rata *pixel* di *frame* sesudahnya. Jika perbedaan *pixel* di *frame* tersebut melebihi batasan yang diberikan, *frame* tersebut dinyatakan macet. Jika jumlah *frame* macet dari sebuah gambar melebihi 50% dari jumlah *frame* yang ada pada gambar tersebut, gambar tersebut dinyatakan macet. Kemudian dikumpulkan dalam satu siklus, dimana satu siklus terdiri dari beberapa gambar. Jika kondisi gambar macet melebihi 50% dari jumlah gambar yang ada dalam satu siklus, kondisi jalan dinyatakan macet.

Menurut Ruben Gonzalez (2005) pada aplikasi *patent*-nya yang berjudul "*Object Oriented Video System*" (p.37), diperlukan beberapa kamera untuk memberikan informasi kepada pengguna sistem pada rute-rute jalan yang akan dilewati apakah macet atau tidak, dengan menggunakan suatu sistem yang menganalisis kondisi jalan di setiap titik kamera yang dipantau oleh sistem. Oleh karena itu diperlukan suatu sistem yang dapat memberikan hasil analisis kemacetan di banyak kamera sepanjang rute yang akan dipantau secara cepat.

Penelitian Terdahulu

Untuk dapat mengakses rangkaian gambar dari sebuah kamera pengawas (TCP/IP enabled Camera) *user* memiliki beberapa opsi yang sudah tersedia saat ini di pasar, di antaranya: (1) melalui *browser* ataupun *streaming client* pada PC/Laptop (HTTP dan RTP); (2) melalui *built-in media player* pada *handphone*/ PDA (RTP) (Ibrahim, 2009).

Opsi pertama yaitu dengan menggunakan PC/Laptop telah menjadi opsi terbanyak *user* saat ini, hal ini diakibatkan oleh telah *mature*-nya teknologi yang tersedia serta besarnya sumber daya (*memory capacity, processor speed, graphic compatibility*) yang dimiliki oleh penggunaan PC/Laptop. Seiring dengan berkembangnya teknologi mobile communication, tuntutan *user* pun semakin meningkat dengan pemanfaatan *hand-held devices* seperti *handphone* ataupun PDA.

Sejalan dengan hal tersebut, opsi kedua, yaitu pemanfaatan *built-in media player* dari sebuah *handphone* untuk dapat mengakses *live video streaming* menjadi trend baru di dalam dunia *video steaming* utamanya pada *live video streaming*. *Hand-held devices* seperti *handphone*/PDA tersebut merupakan alat yang sangat *portable* dibandingkan dengan perangat *portable* lainnya yaitu laptop. Sebuah *handphone* dapat mengakses *streaming server* melalui RTP (*real time protocol*) di atas GPRS. *Streaming* yang dikirimkan memiliki dua *channel* yaitu *video* dan *sound*, dengan kualitas gambar yang seadanya mengingat *bandwidth* yang sempit dan *sharing* dengan slot *bandwidth* untuk sound/suara. Format *video streaming* yang paling optimum saat ini adalah MPEG4 – sebuah metode kompresi data dan suara yang paling terakhir oleh Moving Picture Experts Group (MPEG) menggunakan standard ISO/IEC 14496. Dari perbandingan kualitas gambar, format JPEG/MJPEG merupakan format yang ideal untuk *live video surveilance streaming*. Berikut adalah tabel perbandingan kedua opsi tersebut (Tabel 1):

Tabel 1
Perbandingan MPEG4 dan JPEG/MJPEG

Spesifikasi	MPEG4	JPEG/MJPEG
image	*poor*	*excellent*
sound	*yes*	*No*
available to PC	*yes*	*Yes*
available to Mobile	*yes*	*No*
bandwidth	*high*	*Low*

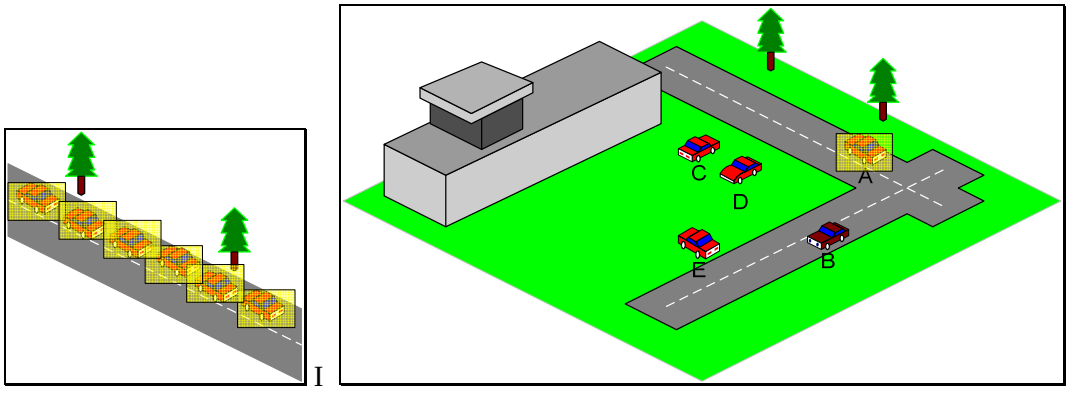
HASIL DAN PEMBAHASAN

Gambaran Umum

Berdasarkan hasil analisis kuesioner sebagai riset pendahuluan dapat ditarik beberapa kesimpulan sebagai berikut: 52.58% responden memilih menggunakan sepeda motor sebagai alterantif transportasi untuk berpergian. Sebagian besar responden (90.72%) sering merasakan kemacetan. Sebagian besar responden (94.85%) merasakan bahwa kemacetan sangat mengganggu kegiatan mereka. 82.47% kerap mencari solusi alternatif jalan lain untuk menghindari kemacetan. 69.07% responden menggunakan alternatif lainnya untuk menghindari kemacetan seperti: bertanya kepada orang di daerah tersebut, melalui siaran radio ataupun televisi, maupun tidak mencari alternatif apapun. 55.67% merasa hasil yang didapat dari alternatif yang mereka gunakan kurang akurat. 84.54% merasa jika ada sistem yang dapat mendeteksi apakah jalan macet atau tidak, akan sangat berguna. Dari hasil yang diperoleh, dapat disimpulkan bahwa sebagian besar responden merasa tidak puas dengan fasilitas dari internet maupun GPS, walaupun internet telah menyajikan begitu banyak data mengenai informasi kemacetan di jalan, dan GPS juga termasuk peralatan yang cukup mahal.

Modul deteksi otomatis kemacetan lalu lintas ini akan bermanfaat langsung bagi pusat-pusat pemantauan, di mana seluruh kamera yang digunakan untuk memonitor ruas-ruas jalan raya tidak perlu dipantau terus menerus secara manual. Modul ini akan otomatis menganalisis dan menyimpulkan gambar bergerak (*video streaming*) yang dikirim dari setiap kamera pengawas, apakah sedang terjadi kemacetan pada ruas jalan yang dipantau atau tidak.

Gambar 1 (I) di bawah ini menunjukkan deteksi terhadap kemacetan akibat akumulasi jumlah *active pinholes* dengan amplitudo yang lebar (pergerakan kendaraan yang sangat lambat atau berhenti). Pada Gambar 1 (II) terdeteksi sebuah kendaraan (mobil A) yang sedang berhenti di persimpangan jalan namun tidak terdeteksi adanya kemacetan lalu lintas akibat jumlah *active pinholes* dibawah *threshold* T. Mobil C, D, dan E berada diluar area pemantauan, sedangkan mobil E sedang berjalan - amplitudo yang sempit.



I

II

Gambar 2. Ilustrasi Traffic-jam detection: (I) kemacetan terdeteksi; (II) terdeteksi kendaraan A berhenti di perempatan jalan tapi kemacetan tidak terdeteksi.

Perancangan modul ini menggunakan metode *motion detection pinhole* dengan penggunaan *m* x *n mesh* untuk *M* x *N* jumlah *pixel* pada sebuah *frame video streaming*. Setiap *pinhole* akan dibandingkan dengan sebuah *threshold* T. Bila terdapat angka rata-rata RGB value yang meningkat secara drastis melebihi T, dapat disimpulkan terjadi gerakan pada area (picture x,y) dimana *pinhole*

berada. Langkah selanjutnya adalah mencari titik tengah atau centroid dari kumpulan *pinhole*s sehingga dapat diambil kesimpulan *fixed position* sebuah object pengamatan. Centroid dapat diperoleh melalui rumus berikut:

$$x_c = \frac{\sum x_j}{n}, \quad y_c = \frac{\sum y_j}{n},$$

dimana (*xc*, *yc*) merupakan posisi (*x*,*y*) dari centroid, sedangkan *xj*, *yj* merupakan *point* terluar/*edge* dari sebuah daerah pengamatan *pinhole*s. Setiap *pinhole* yang mengindikasikan posisi kendaraan akan dianalisis frekuensi perubahan nilai *pixel*-nya. Frekuensi yang tinggi menggambarkan pergerakan kendaraan yang cepat, sedangkan frekuensi yang rendah menggambarkan pergerakan kendaraan yang sangat lambat ataupun berhenti/terdeteksinya kemacetan.

Tahap Pemprosesan Awal

Proses pengembangan awal dalam aplikasi ini adalah mengambil gambar yang dikirimkan oleh *IP Camera* dari alamat *IP Camera* yang sudah ada (Bradski & Adrian, 2008). Kemudian gambar dalam bentuk MJPEG dipecah-pecah menjadi gambar JPEG. Setelah mendapatkan gambar JPEG, gambar tersebut diolah dalam beberapa tahap. Tahap pertama adalah dengan cara membuat gambar tersebut dalam *grayscale*, yaitu dengan cara mengambil nilai *pixel* di tiap titik dengan sintaks sebagai berikut:

```
res = bimg.getRaster().getPixel(j,k,tmp);
```

dimana:
res adalah *int* sebagai hasil *pixel* yang berurut dari index 0, 1, dan 2 mewakili warna merah, biru, dan hijau,
bimg adalah *BufferedImage* yang membaca *file* JPEG yang dimiliki,
j adalah koordinat horizontal dari gambar,
k adalah koordinat vertikal dari gambar,
dan *tmp* adalah *int* yang merupakan sebuah *array int preallocated* opsional.

Kemudian kita mengubah warna yang ditampung dalam variabel *res* menjadi warna dalam *grayscale* dengan cara sebagai berikut:

Nilai *Grayscale* = Merah + Biru + Hijau

Setelah mendapatkan nilai *grayscale* untuk semua titik, kemudian kita mengambil nilai modus untuk semua titik untuk mendapatkan gambar *background*. *Background* ini yang akan digunakan untuk menghilangkan *background*.

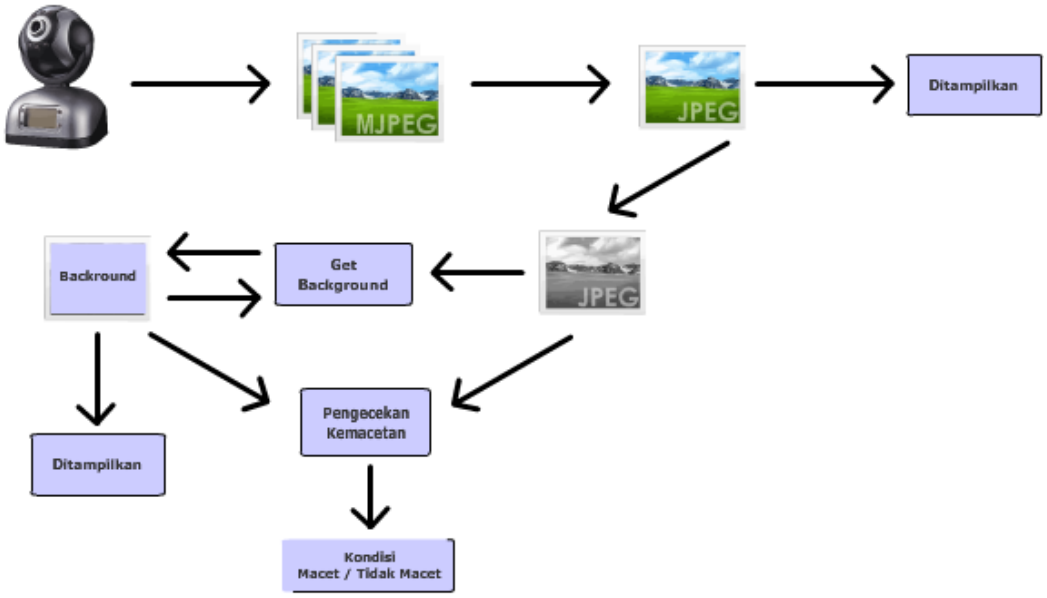
Gambar yang ada dibuat menjadi gambar Object dengan cara:

Gambar Objek = Gambar Asli – Gambar *Background*

Dalam perhitungan kemacetan di sini, dilakukan tiga kali perhitungan kemacetan, yaitu kemacetan per *pinhole*, kemacetan per gambar, dan kemacetan dalam kurun waktu tertentu yang menggambarkan kondisi kemacetan jalan sebagai hasil perhitungan.

Gambar objek yang dihasilkan dibagi-bagi dengan ukuran yang sama menjadi *pinhole-pinhole* kecil. Kemudian dari tiap *pinhole* ini diambil nilai rata-rata dari *pixel* yang ada dalam *pinhole* tersebut. Nilai rata-rata dari tiap *pinhole* tersebut akan dibandingkan dengan *pinhole* yang ada pada posisi yang sama di gambar sebelumnya. Dalam batasan tertentu nilai pada *pinhole* tersebut dapat dikatakan macet. Nilai kemacetan *pinhole* dikumpulkan per gambarnya, dan dijadikan nilai kemacetan dari suatu gambar (Brodsky, 2007). Dengan kondisi jika lebih dari 50% *pinhole* yang macet dari gambar yang diamati, maka gambar tersebut dinyatakan macet. Nilai kemacetan per gambar juga diakumulasikan dalam periode beberapa gambar (ditentukan nilainya dalam percobaan). Jika lebih dari 50% dari

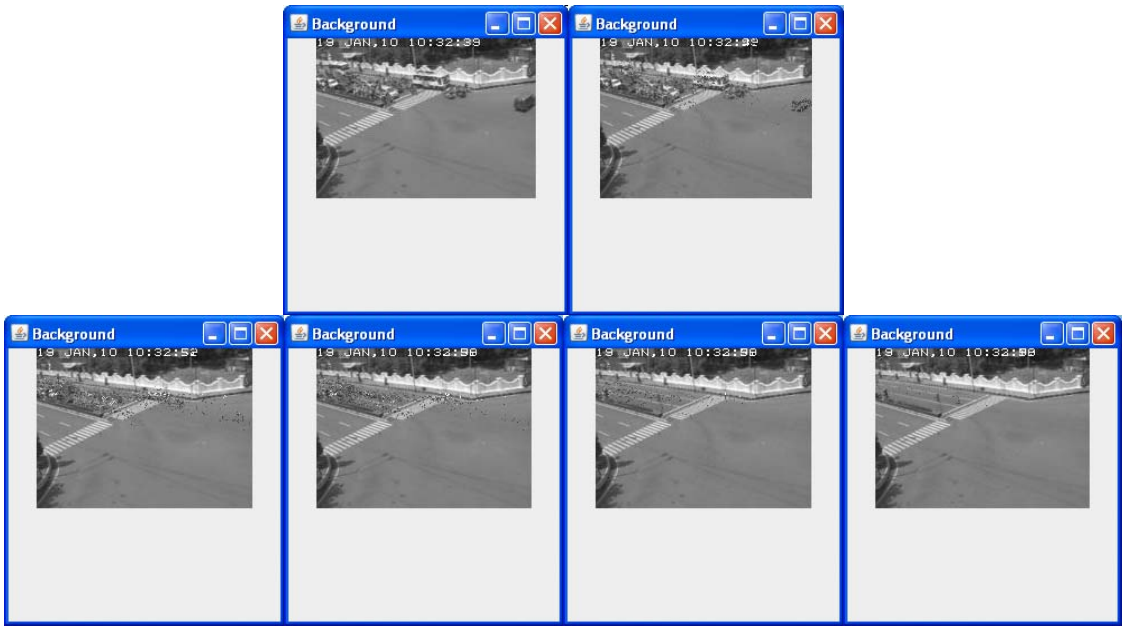
jumlah gambar yang ditentukan, kondisi jalan saat itu dinyatakan sebagai macet. Tahapan pengolahan video ini terangkum pada Gambar 3.



Gambar 3. Tahapan pengolahan video.

Hasil Pengambilan *Background*

Aplikasi hasil penelitian "Pendeteksian Kemacetan dengan mengimplementasikan Algoritma *Pinhole*" (Atkoˇci ¯unas, 2005) melakukan pengolahan *background* secara terus menerus untuk mendapatkan hasil *background* yang lebih sempurna. Hasil pengambilan *background* dapat dicontohkan sebagai berikut (Gambar 4):



Gambar 4. Proses pengambilan *background*.

Dalam pengambilan gambar ini dilakukan teknik pengambilan *modus*, sehingga semakin lama aplikasi berjalan, maka *background* yang dihasilkan semakin baik. Proses pengambilan *background* awal ditentukan selama tiga menit. Hal ini dilakukan sebagai usaha untuk mendapatkan *background* yang *valid* terlebih dahulu. Dalam prakteknya, gambar *background* setelah tiga menit terkadang masih belum sempurna, sehingga dilakukan pengambilan *background* secara terus menerus.

Parameter Hasil

Dalam aplikasi ini terdapat beberapa parameter yang ditentukan menurut suatu pertimbangan dan asumsi (Gonzalez, 2005; Ibrahim, 2009).

Parameter pertama adalah ukuran gambar. Menimbang dari segi koneksi, serta kecepatan *download*/pengambilan gambar sebagai sumber data, ditentukan ukurang yang kecil, tetapi masih dalam taraf bisa dilihat/diamati, yaitu 160 x 120 *pixel*, tetapi pada kenyataannya gambar yang didapat berukuran 192 x 144 *pixel*.

Parameter kedua adalah besar *download.* Gambar berukurang 192 x 144 *pixel* disimpan dengan ukuran berkisar antara 5 kB sampai dengan 7 kB. Ditambah dengan header dari MJPEG, sehingga diperkirakan 9 kB.

Parameter ketiga adalah ukuran *pixel pinhole.* Parameter ukuran *pixel* ditentukan dengan percobaan beberapa ukuran, yaitu 1 x 1, 2 x 2, 4 x 4, 8 x 8, dan 16 x 16 pada data MJPEG yang diambil pada saat yang hampir bersamaan. Dengan parameter lainnya yaitu toleransi perbedaan *pixel* sebesar 5 dan banyak gambar untuk tiap analisis 10. Berikut sample hasil pendeteksian dengan ukuran *pinhole* yang berbeda, yaitu 1 x 1, 2 x 2, 4 x 4, 8 x 8, 16 x 16:

Hasil Percobaan dengan Ukuran 1 X 1

Jumlah *pinhole* macet yang diperlukan untuk menentukan kemacetan adalah 2420. Tabel 2 memuat hasil percobaannya.

Tabel 2
Hasil Jumlah Pinhole Macet pada 1 x Deteksi pada 10 Gambar dengan Ukuran Pixel 1 x 1

Gambar ke	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Jumlah*Pinhole* Macet	4544	4075	4432	4073	3770	4588	4523	4568	4634	3891

Jumlah gambar macet adalah 10 sehingga dihasilkan kondisi macet.

Hasil Percobaan dengan Ukuran 2 x 2

Jumlah *pinhole* macet yang diperlukan untuk menentukan kemacetan adalah 605. Tabel 3 memuat hasil percobaannya.

Tabel 3
Hasil Jumlah Pinhole Macet pada 1 x Deteksi pada 10 Gambar dengan Ukuran Pixel 2 x 2

Gambar ke	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Jumlah *Pinhole* Macet	888	554	655	681	513	620	758	749	774	416

Jumlah gambar macet adalah 7 sehingga dihasilkan kondisi macet.

Hasil percobaan dengan Ukuran 4 x 4

Jumlah *pinhole* macet yang diperlukan untuk menentukan kemacetan adalah 152. Tabel 4 memuat hasil percobaannya.

Tabel 4
Hasil Jumlah Pinhole Macet pada 1 x Deteksi pada 10 Gambar dengan Ukuran Pixel 4 x 4

Gambar ke	1	2	3	4	5	6	7	8	9	10
Jumlah *Pinhole* Macet	184	145	97	126	82	110	93	173	128	82

Jumlah gambar macet adalah 2 sehingga dihasilkan kondisi tidak macet.

Hasil Percobaan dengan Ukuran 8 X 8

Jumlah *pinhole* macet yang diperlukan untuk menentukan kemacetan adalah 38. Tabel 5 memuat hasil percobaannya.

Tabel 5
Jumlah Pinhole Macet pada 1 x Deteksi pada 10 Gambar dengan Ukuran Pixel 8 x 8

Gambar ke	1	2	3	4	5	6	7	8	9	10
Jumlah *Pinhole* Macet	47	57	56	7	3	45	55	34	42	37

Jumlah gambar macet adalah 6 sehingga dihasilkan kondisi tidak macet.

Hasil Percobaan dengan Ukuran 16 X 16

Jumlah *pinhole* macet yang diperlukan untuk menentukan kemacetan adalah 9. Tabel 6 memuat hasil percobaannya.

Tabel 6
Jumlah Pinhole Macet pada 1 x Deteksi pada 10 Gambar dengan Ukuran Pixel 16x16

Gambar ke	1	2	3	4	5	6	7	8	9	10
Jumlah *Pinhole* Macet	2	6	4	2	10	11	9	9	10	9

Jumlah *pinhole* macet yang diperlukan untuk menentukan kemacetan adalah 6. Kondisi jalan yang sedang diamati, adalah tidak macet, sehingga dapat dikatakan ukuran 4 x 4 dapat mengambil keputusan lebih baik.

Toleransi Perbedaan Pixel

Pencahayaan yang sering kali berubah menambah tingkat kesulitan dalam mendeteksi kemacetan, khususnya dalam hal mendapatkan gambar objek atau melakukan penghilangan *background*. Misalnya saja *pixel* abu-abu yang didapat sekarang adalah 65, kemudian *pixel* abu-abu di tempat yang sama, tetapi gambar yang berbeda dengan *object* di area tersebut masih sama, bisa saja mendapatkan *pixel* abu-abu 66. Hal ini disebabkan oleh karena tingkat pencahayaan yang selalu berubah-ubah sehingga perlu adanya toleransi perbedaan *pixel*.

Parameter toleransi perbedaan *pixel* ditentukan dari beberapa hasil percobaan dengan nilai toleransi yang berbeda, yaitu 5, 10 , 15 ,20 , dan 25. Untuk parameter lainnya, ukuran *pinhole* 4 x 4,

banyak gambar per analisis 10, dengan jumlah *pinhole* macet untuk menentukan macet adalah 152. Berikut contoh hasil pendeteksian dengan nilai toleransi yang berbeda (Tabel 7 – 11):

Nilai Toleransi = 5

Tabel 7
Hasil Jumlah Pinhole Macet pada 1 X Deteksi Pada 10 Gambar dengan Nilai Toleransi 5

Gambar ke	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Jumlah*Pinhole* Macet	123	60	50	91	148	131	86	113	120	103

Jumlah gambar macet adalah 0, sehingga dihasilkan kondisi tidak macet.

Nilai Toleransi = 10

Tabel 8
Hasil Jumlah Pinhole Macet pada 1 x Deteksi pada 10 Gambar dengan Nilai Toleransi 10

Gambar ke	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Jumlah*Pinhole* Macet	263	234	211	270	299	275	269	235	245	275

Jumlah gambar macet adalah 10, sehingga dihasilkan kondisi macet.

Nilai Toleransi = 15

Tabel 9
Hasil Jumlah Pinhole Macet pada 1 x Deteksi pada 10 Gambar dengan Nilai Toleransi 15

Gambar ke	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Jumlah*Pinhole* Macet	324	363	398	405	370	369	318	335	299	309

Jumlah gambar macet adalah 10, sehingga dihasilkan kondisi macet.

Nilai Toleransi = 20

Tabel 10
Hasil Jumlah Pinhole Macet pada 1 x Deteksi pada 10 Gambar dengan Nilai Toleransi 20

Gambar ke	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Jumlah*Pinhole* Macet	407	382	397	421	415	382	365	347	361	360

Jumlah gambar macet adalah 10, sehingga dihasilkan kondisi macet.

Nilai Toleransi = 25

Tabel 11
Hasil Jumlah Pinhole Macet pada 1 x Deteksi pada 10 Gambar dengan Nilai Toleransi 25

Gambar ke	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Jumlah*Pinhole* Macet	379	384	401	415	425	416	393	350	366	372

Jumlah gambar macet adalah 10, sehingga dihasilkan kondisi macet.

Gambar yang sedang diawasi dalam kondisi tidak macet. Sehingga dapat disimpulkan bahwa nilai parameter untuk toleransi perubahan *pixel* yang lebih baik adalah 5.

Jumlah Gambar untuk Setiap Analisis

Jumlah gambar untuk tiap sampel analisis kemacetan selain mempengaruhi kecepatan analisis, juga mempengaruhi ketepatan analisis. Oleh karena itu, perlu diatur nilai parameter dari banyak gambar untuk tiap analisis. Nilai parameter untuk menentukan banyak gambar per analisis ditentukan dari hasil pengamatan.

Dari beberapa gambar di bawah ini (Gambar 5 – 8) telah dilakukan pengamatan dengan hasil sebagai berikut:

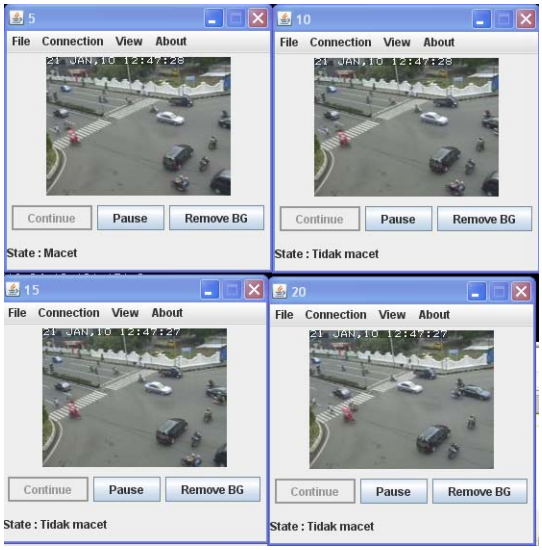
Banyak gambar per analisis = 5
Dengan banyak gambar per analisis 5 gambar, analisis yang dihasilkan terlalu cepat berubah, sehingga dirasakan kurang.

Banyak gambar per analisis = 10
Waktu yang diperlukan cukup memadai untuk melakukan analisis jalan. Hasil yang didapat cukup akurat, sekalipun tidak 100%.

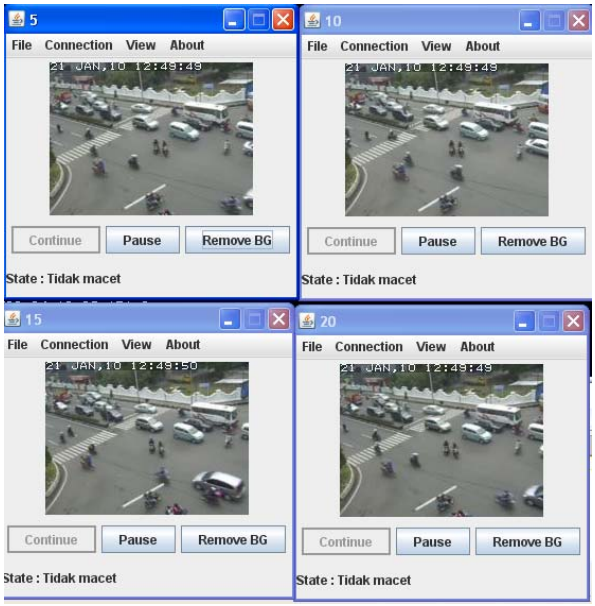
Banyak gambar per analisis = 15
Waktu yang diperlukan untuk menganalisis cukup lama, terkadang hasil analisis terlambat dan tidak sesuai dengan kondisi jalan.

Banyak gambar per analisis = 20
Waktu yang diperlukan terlalu lama, sehingga hasil analisis sangat terlambat dan tidak sesuai dengan kondisi jalan.

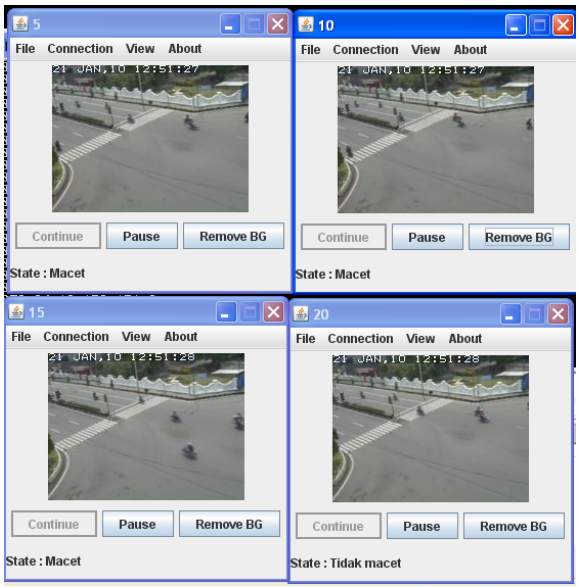
Sehingga dapat disimpulkan, nilai parameter untuk banyak gambar per analisis yang dapat menghasilkan nilai optimum adalah 10.



Gambar 5. Hasil pendeteksian dengan banyak gambar tiap analisis yang divariasikan – 1. Kiri atas = 5, kanan atas = 10, kiri bawah = 15, kanan atas= 20.



Gambar 6. Hasil pendeteksian dengan banyak gambar tiap analisis yang divariasikan – 2. Kiri atas = 5, kanan atas = 10, kiri bawah = 15, kanan atas = 20

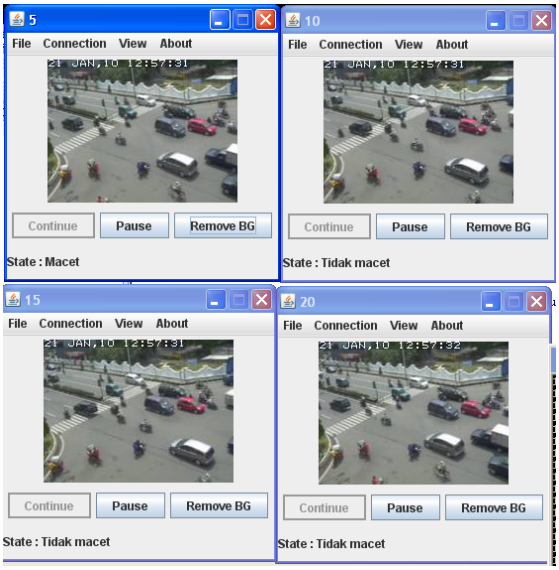


Gambar 7. Hasil pendeteksian dengan banyak gambar tiap analisis yang divariasikan – 3. Kiri atas = 5, kanan atas = 10, kiri bawah = 15, kanan atas = 20.

Besar Area Analisis

Gambar yang didapatkan belum mencukupi untuk melakukan analisis kemacetan karena daerah yang diamati masih termasuk jalan yang lancar, dan tidak rawan macet. Untuk sumber yang lain, belum ditemukan yang lebih baik dari sumber yang digunakan saat ini. Oleh karena itu diinisiatifkan untuk menggunakan bagian dari area yang diamati yang bisa mencukupi kondisi penganalisisan kemacetan, yaitu di bagian kiri atas. Di daerah tersebut terdapat lampu merah yang bisa

dijadikan patokan, yaitu ketika lampu merah, diasumsikan macet, dan ketika lampu hijau diasumsikan tidak macet. Dapat diperkirakan, daerah yang diamati adalah 25% dari gambar secara keseluruhan (Gambar 9).



Gambar 8. Hasil pendeteksian dengan banyak gambar tiap analisis yang divariasikan – 3. Kiri atas = 5, kanan atas = 10, kiri bawah = 15, kanan atas = 20.

Gambar 9. Frame utuh.

Diambil 25% hanya pada wilayah tertentu, yaitu di area kiri atas, sehingga area yang diamati adalah (Gambar 10):

Gambar 10. Penggalan area yang diamati pada bagian kiri atas

Jika diperhatikan dari gambar tersebut, sekitar 30% dari gambar tersebut adalah *background* (Gambar 11).

Gambar 11. Daerah yang diarsir merah merupakan *background*.

Sehingga dapat diasumsikan area yang diamati memiliki 70% daerah gambar object dan 30% daerah gambar *background*.

Presentase Jumlah *Pinhole* untuk Melakukan Analisis

Presentase jumlah *pinhole* untuk menentukan sebuah gambar macet atau tidak ditentukan dengan asumsi kondisi macet pada suatu gambar memerlukan jumlah *pinhole* 50% dari keseluruhan jumlah *pinhole* ditambah dengan 1. Sehingga didapatkan presentasenya adalah 50%.

Dalam aplikasi ini dapat dihitung jumlah *pinhole* macet yang diperlukan untuk menentukan kondisi macet pada suatu gambar, dengan perhitungan berikut:

Jumlah *pinhole* 1 Gambar = ((panjang gambar)*(lebar gambar)) / ((panjang *pinhole*) *(lebar *pinhole*))
Jumlah *pinhole* 1 Gambar = (192 x 144) / (4 x 4) = 1728

Sehingga didapat pada 1 gambar terdapat 1728 *pinhole*.

Area pengamatan adalah 25% dari keseluruhan area, yaitu area kiri atas. Sehingga daerah yang diamati adalah:

Jumlah *pinhole* yang diamati = Jumlah *pinhole* 1 gambar * 25%
Jumlah *pinhole* yang diamati = 1728 * 25% = 432

Sehingga didapatkan *pinhole* yang diamati adalah 432 *pinhole*.

Pada *pinhole* tersebut masih terdapat area yang dianggap *background*, yaitu sekitar 30%, sehingga *pinhole* yang merupakan *pinhole* dari objek adalah:

Jumlah *pinhole* objek = Jumlah *pinhole* yang diamati * 70%
Jumlah *pinhole* objek = 432 * 70% = 302,4 dibulatkan menjadi 302

Sehingga didapatkan *pinhole* dari gambar objek sebanyak 302 *pixel*.

Dapat diasumsikan, jumlah *pinhole* macet dalam satu gambar yang diperlukan untuk menentukan macet atau tidaknya suatu gambar adalah sebagai berikut:

Jumlah *pinhole* macet yang diperlukan = (Jumlah *pinhole* objek * 50%) + 1
Jumlah *pinhole* macet yang diperlukan = (302 * 50%) + 1 = 152.

Implementasi dan Evaluasi

Evaluasi dilakukan untuk pendeteksian kemacetan lalu lintas terhadap 78 kasus yang berbeda. Hasil Ujicoba Pendeteksian Kemacetan Lalu lintas dihasilkan data, yaitu 55 hasil deteksi tepat dengan total 78 deteksi. Presentase hasil analisis = (jumlah benar / total deteksi) * 100. Sehingga didapatkan 70% deteksi yang memberikan hasil pendeteksian ini benar.

PENUTUP

Pada akhirnya dapat disimpulkan bahwa aplikasi hasil penelitian "Pendeteksian Kemacetan dengan Mengimplementasikan Algoritma *Pinhole*" dapat menganalisis kemacetan dengan tingkat keakuratan 70 %. Beberapa batasan yang belum dapat diselesaikan oleh aplikasi hasil penelitian "Pendeteksian Kemacetan dengan Mengimplementasikan Algoritma *Pinhole*" ini adalah sebagai berikut: (1) dalam mengambil *background* belum dapat menghilangkan bayangan (*classic shadowing problem*); (20 *background* yang didapat memerlukan waktu untuk direset karena adanya pergantian cuaca, pergantian pagi ke siang, siang ke malam. Namun, belum dapat mengukur waktu yang tepat untuk dapat di-*reset*. Algoritma ini dapat diimplementasikan pada tiap-tiap jalan di daerah yang rawan macet, untuk menghasilkan hasil pendeteksian kemacetan yang baik, tetapi pada setiap titik kemacetan haruslah digunakan beberapa aplikasi pada IP-Camera yang berbeda untuk menghasilkan hasil yang optimal. Jika hanya dengan satu IP-Camera, hasil yang didapat akan kurang efektif. Penggunaan aplikasi ini juga dapat digabungkan dengan layanan SMS gateway untuk memeriksa kemacetan pada suatu jalan yang diinginkan. Aplikasi ini masih dapat dikembangkan lebih lanjut lagi dengan menggunakan algoritma-algoritma yang lebih efektif dalam melakukan analisis seperti memperbaiki teknik menghilangkan *background* atau melakukan penelitian lebih lanjut dalam optimasi menentukan parameter.

DAFTAR PUSTAKA

Atkoˇci ¯unas, E., Blake, R., Juozapaviˇcius, A., Kazimianec, M. (2005). *Image Processing in Road Traffic Analysis. Nonlinear Analysis: Modelling and Control. 10* (4), 315–332.

Bradski, Gary & Adrian K. (2008). *Learning* OpenCV: *Computer Vision with the OpenCV Library*. Massachusetts: O'Reilly Media.

Brodsky T. (2007). *Relevant Image Detection in a Camera, Recorder, or Video streaming Device.* United States Patent Application 20070024707. Pub. No. US 2007/0024707 Al.

Gonzalez, R. (2005). *Object Oriented Video System.* United States Patent Application 20070005795. International Application No. PCT/AU2000/001296.

Ibrahim, Amin A. (2009). Detecting and Preventing the Electronic Transmission of Illicit Image. *Social Informatics and Telecommunications Engineering, 31* (1), 36 – 39.

Mahatmaputra, S. (2010). *Analisis dan Perancangan Aplikasi Monitoring IP Camera menggunakan Protocol HTTP pada Mobile Phone*. Yogyakarta: SNATI.

The Jakarta Post. (2008). *Jakarta Lags in Living Quality and Safety*. Jakarta.