TESIS-SM 142501

PELACAKAN OBJEK KECIL MENGGUNAKAN METODE *ADAPTIVE PARTICLE FILTER* BERBASIS SUPER-RESOLUSI

Yabunayya Habibi
1214 201 209

DOSEN PEMBIMBING
Dr. Dwi Ratna S, S.Si, MT
Dr. Budi Setiyono, S.Si, MT

PROGRAM MAGISTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2017

THESIS-SM 142501

THE SMALL OBJECT TRACKING USING ADAPTIVE PARTICLE FILTER BASED ON SUPER-RESOLUTION

Yabunayya Habibi
1214 201 209

SUPERVISORS
Dr. Dwi Ratna S, S.Si, MT
Dr. Budi Setiyono, S.Si, MT

MASTER'S DEGREE
MATHEMATICS DEPARTMENT
FACULTY OF MATHEMATICS AND NATURAL SCIENCES
SEPULUH NOPEMBER INSTITUTE OF TECHNOLOGY
SURABAYA
2017

PELACAKAN OBJEK KECIL MENGGUNAKAN METODE *ADAPTIVE PARTICLE FILTER* BERBASIS SUPER-RESOLUSI

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Sains (M.Si.)
di
Institut Teknologi Sepuluh Nopember

oleh:
YABUNAYYA HABIBI
NRP. 1214 201 209

Tanggal Ujian : 4 Januari 2017
Periode Wisuda : Maret 2017

Disetujui oleh:

Dr. Dwi Ratna Sulistyaningrum, S.Si., M.T. (Pembimbing I)
NIP: 19690405 199403 2 003

Dr. Budi Setiyono, S.Si, M.T. (Pembimbing II)
NIP: 19720207 199702 1 001

Dr. Didik Khusnul Arif, S.Si., M.Si. (Penguji)
NIP. 19730930 199702 1 001

Dr. Imam Mukhlash, S.Si., M.T. (Penguji)
NIP. 19700831 199403 1 003

an. Direktur Program Pascasarjana
Asisten Direktur

Prof. Dr. Ir. Tri Widjaja, M.Eng.
NIP. 19611021 198603 1 001

Direktur Program Pascasarjana,

Prof. Ir. Djauhar Manfaat, M.Sc., Ph.D
NIP. 19601202 198701 1 001

PELACAKAN OBJEK KECIL MENGGUNAKAN METODE *ADAPTIVE PARTICLE FILTER* BERBASIS SUPER-RESOLUSI

Nama mahasiswa	: Yabunayya Habibi
NRP	: 1214 2012 09
Pembimbing	: Dr. Dwi Ratna S, S.Si, MT.
	Dr. Budi Setiyono, S.Si, MT

ABSTRAK

Pelacakan objek dalam sebuah video adalah masalah dalam memperkirakan posisi suatu objek pada bidang gambar ketika bergerak disekitar sebuah *scene*. Pelacakan objek secara umum merupakan permasalahan yang cukup rumit. Hal ini dapat timbul karena gerakan objek yang cepat, perubahan pola penampakan objek, struktur objek yang *non-rigid*, kamera yang bergerak dan lainnya. Tingkat permasalah akan semakin tinggi jika objek memiliki ukuran yang relatif kecil. Jika hal ini terjadi, sebuah objek akan sulit untuk diidentifiksi dan pelacakan menjadi kurang presisi. Untuk mengatasi permasalahan tersebut, pelacakan akan diintegrasikan dengan super-resolusi dimana sebuah citra resolusi tinggi akan dibangun dari beberapa citra resolusi rendah. Dalam penelitian ini, pelacakan objek bergerak menggunakan Adaptive Particle Filter dimana model pergerakan adaptive diimplementasikan untuk mendapatkan pendekatan distribusi yang lebih baik. Dari pengujian terhadap lima video yang telah tersedia, terjadi peningkatan akurasi pelacakan pada video Motorcycle, Bicycle dan Surveillance masing-masing sebesar 52,4%, 60% dan 73,3%. Hal ini juga terjadi pada video paralayang dengan peningkatan sebesar 20%. Peningkatan presisi pada video Helikopter hanya mencapai 8,4 % karena adanya faktor-faktor yang menyebabkan kesalahan estimasi dalam proses pelacakan seperti adanya getaran pada saat akuisisi video.

Kata kunci: *adaptive particle filter*, multi-frame, objek kecil, pelacakan, super-resolusi.

THE SMALL OBJECT TRACKING USING ADAPTIVE PARTICLE FILTER BASED ON SUPER-RESOLUTION

Name : Yabunayya Habibi
NRP : 1214 2012 09
Supervisors : Dr. Dwi Ratna S, S.Si, MT.
Dr. Budi Setiyono, S.Si, MT

ABSTRACT

Object tracking in a video is a problem of estimating the position of an object in the image plane as it moves around a scene. In general, object tracking is a quite complicated problem. Difficulties in object tracking occur due to some constraints or conditions such as object motion, changing appearance patterns, non-rigid object structures, camera motion and others. Level of problems would be higher if the object has a relatively small size. If it happens, an object will be difficult to identify and tracking becomes less precision. In order to overcome these problems, the tracking will be integrated with the super-resolution in where a high-resolution image will be built from several low-resolution image. In this research, tracking of moving object using adaptive particle filter which adaptive motion model is applied to get better distribution approach. From examination of five videos that are available, an increase tracking accuracy in video of Motorcycle, Bicycle and Surveillance respectively 52,4%, 60% and 73,3%. It also occurs in the video of paralayang with an increase of 20%. Increased precision in the helicopter video only reached 8,4% due to the factors that led to the estimation error in the tracking process such as vibration during video acquisition.

Keywords: adaptive particle filter, small object, super-resolution, tracking.

DAFTAR ISI

Halaman

HALAMAN JUDUL i

LEMBAR PENGESAHAN TESIS v

ABSTRAK vii

ABSTRACT ix

KATA PENGANTAR xi

DAFTAR ISI xiii

DAFTAR GAMBAR xv

DAFTAR TABEL ixx

BAB I PENDAHULUAN 1

1.1 Latar Belakang 1

1.2 Rumusan Masalah 2

1.3 Batasan Masalah 3

1.4 Tujuan Penelitian 3

1.4 Manfaat Penelitian 4

BAB II KAJIAN PUSTAKA DAN DASAR TEORI 5

2.1 Penelitian Sebelumnya 5

2.2 Pelacakan Objek 6

2.3 Adaptive Particle Filter 7

2.4 Algoritma Block Matching 12

2.5 Super-resolusi 14

2.6 Metode *Phased Based Image Matching* 15

2.7 Algoritma Projection Onto Convex Sets 16

BAB III METODA PENELITIAN 19

3.1 Objek dan peralatan Penelitian 19

3.2 Tahapan Penelitian 19

3.3 Block Diagram 21

BAB IV SUPER-RESOLUSI DAN PELACAKAN OBJEK 25

4.1 Perancangan Proses 25

4.1.1 Super-resolusi 25
4.1.2 Pelacakan Objek 33
4.1.3 Estimasi Objek 40
4.2 Perancangan Perangkat Lunak 42
4.2.1 Proses Kerja Perancangan Lunak 42
4.2.2 Kebutuhan Perangkat Lunak 43
4.2.3 Perancangan Proses Akuisisi Data 44
4.2.4 Perancangan Antar Muka 46
4.3 Implementasi Pelacakan Objek berbasis Super-resolusi 48
4.3.1 Implementasi Proses Konversi Video ke Frame 48
4.3.2 Proses Pra-pengolahan 51
4.3.3 Super-resolusi 52
4.3.4 Proses Pengambilan Input Data *Frame* Video dan parameter 53
4.3.5 Proses Pemilihan Objek dengan ROI 55
4.3.6 Pelacakan Objek Menggunakan *Adapative Particle Filter* 56
BAB V PENGUJIAN DAN PEMBAHASAN 63
5.1 Data Uji Coba 63
5.2 Pengujian Pemilihan Objek 65
5.3 Pengujian terhadap Pelacakan Objek 66
5.4 Pembahasan Hasil Pengujian 74
5.5 Pembahasan Penyebab Kecilnya Persentase Akurasi 76
BAB VI PENUTUP 77
6.1 Kesimpulan 77
6.2 Saran 78
DAFTAR PUSTAKA 79
LAMPIRAN 81

DAFTAR GAMBAR

Halaman

Gambar 2.1 Taksonomi dari metode pelacakan 6
Gambar 2.2 Ilustrasi *trajectory fitting*..... 10
Gambar 2.3. Prosedur Algoritma *Adaptive Particle Filter* 11
Gambar 2.4. Langkah-langkah dalam algoritma *block matching* 13
Gambar 2.5. Diagram alir algoritma *Hierarchical Block Matching*. 14
Gambar 2.6. Proses kontruksi *frame* super-resolusi dari serangkaian frame resolusi rendah. 15
Gambar 3.1 Tahapan pelacakan objek dengan *Adaptive Particel Filter* berbasis Super-resolusi 21
Gambar 4.1. Ilustrasi citra objek sebagai serangkaian resolusi rendah..... 26
Gambar 4.2. Diagram alir registrasi citra menggunakan PBIM..... 30
Gambar 4.3. Diagram alir rekontruksi citra menggunakan POCS..... 32
Gambar 4.4 Diagram alir dari pelacakan dengan *Adaptive Particle Filter*..... 40
Gambar 4.5. Diagram alir estimasi gerak objek menggunakan *Hierarchical Block Matching* 42
Gambar 4.6. Antar muka halaman utama..... 46
Gambar 4.7. Perancangan halaman pelacakan 47
Gambar 4.8. Perancangan antar muka untuk proses konversi video ke frame..... 47
Gambar 4.9. Perancangan antar muka proses konversi video ke frame..... 48
Gambar 4.10. Antar muka *input* video..... 49
Gambar 4.11. Antar muka proses konversi video ke frame..... 51
Gambar 4.12. Antar muka proses super-resolusi 52
Gambar 4.13. Antar muka pelacakan objek kecil berbasis super-resolusi yang dilanjutkan proses *input* data *frame* video. 54
Gambar 4.14. Bagian frame yang dipilih sebagai ROI 56
Gambar 4.15. Tampilan antar muka hasil pelacakan objek. 62

Gambar 5.1 Citra Objek yang dilacak 65
Gambar 5.2 .Tiga kriteria tingkat ketepatan dalam pelacakan objek 66
Gambar 5.3. Ilustrasi hasil pelacakan 67
Gambar 5.4. Perbandingan parameter jumlah partikel terhadap rata-rata waktu komputasi per-*frame* pelacakan tanpa super-resolusi 70
Gambar 5.5. Perbandingan parameter jumlah partikel terhadap rata-rata waktu komputasi per-*frame* pelacakan berbasis super-resolusi 71
Gambar 5.6. Hasil visualisasi pelacakan. 71
Gambar 5.7 Grafik tingkat akurasi pelacakan objek tanpa super-resolusi. 73
Gambar 5.8. Grafik tingkat akurasi pelacakan objek berbasis super-resolusi 73
Gambar 5.9. Peningkatan akurasi pelacakan berbasis super-resolusi terhadap pelacakan tanpa super-resolusi 76

DAFTAR TABEL

Halaman

Tabel 4.1. Koordinat posisi objek 35
Tabel 4.2. Kebutuhan Perancangan Perangkat Lunak 43
Tabel 4.2. Tabel data proses..... 45
Tabel 5.1 Data Video 63
Tabel 5.2 Hasil Pelacakan objek tanpa super-resolusi 67
Tabel 5.3 Tampilan objek citra dan hasil citra super-resolusi 68
Tabel 5.4 Hasil pelacakan objek berbasis citra super-resolusi..... 69
Tabel 5.5. Perbandingan hasil pelacakan (*frame*) 72
Tabel 5.6. Hasil pelacakan berdasarkan jumlah citra referensi 74
Tabel 5.7. Perbandingan hasil persentase akurasi pelacakan objek 75

DAFTAR LAMPIRAN

Halaman

Lampiran A Kode Program Pelacakan dengan Metode *Adaptive Particle Filter*. 81

Lampiran B Kode Program Estimasi *Hierarchical Block Matching* dengan algoritma pencarian *Exhaustive Search*. 85

Lampiran C Kode Program *Sum of Square Difference* (SSD) 89

Lampiran D Kode Fungsi Resampling 91

Lampiran E Kode Program *Phased Based Image Matching* (PBIM) 93

Lampiran F Kode Program Rekontruksi Citra dengan *Projection Onto Convex Sets* (POCS) 97

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Segala Puji bagi Allah SWT Tuhan semesta alam yang telah melimpahkan hidayah, karunia, rahmat dan anugerah-Nya sehingga penulis dapat menyelesaikan tesis yang berjudul: **"Pelacakan Objek Kecil Menggunakan Metode Adaptive Particle Filter Berbasis Super-resolusi"** dengan baik dimana tesis ini merupakan salah satu persyaratan akademis dalam menyelesaikan program magister Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Institut Teknologi Sepuluh Nopember Surabaya.

Tesis ini dapat diselesaikan tidak terlepas dari begitu banyak bantuan, kerjasama, dan dukungan dari banyak pihak. Sehubungan dengan hal itu, penulis mengucapkan terima kasih kepada:

1. Dr. Dwi Ratna Sulistyaningrum, S.Si, MT dan Dr. Budi Setiyono, S.Si, MT selaku dosen pembimbing yang senantiasa membimbing, meluangkan waktu untuk penulis dalam penelitian dan proses penyusunan tesis dengan sabar, dan memberikan kritik dan saran dalam penyusunannya.
2. Dr. Imam Mukhlash, S.Si, MT selaku Ketua Jurusan Matematika dan penguji Tesis ini.
3. Dr. Didik Khusnul Arif, S.Si, M.Si selaku dosen penguji Tesis ini.
4. Dr. Hariyanto, M.Si selaku dosen wali yang selalu terbuka dan mengarahkan penulis dalam masa studi S2 di matematika ITS.
5. Seluruh jajaran dosen dan staf dan karyawan pascasarjana Matematika ITS yang telah memberikan bekal ilmu dan segala kemudahan dan kelancaran selama mengikuti perkulihan.
6. Seluruh teman-teman mahasiswa pascasarjana Matematika ITS.

Karena keterbatasan pengetahuan dan pengalaman, penulis menyadari bahwa Tesis ini masih banyak kekurangan yang jauh dari kesempurnaan. Oleh karena itu, penulis mengharapkan saran dan kritik dari pembaca untuk kesempurnaan penelitian selanjutnya. Akhir kata, semoga Tesis ini bermanfaat bagi semua pihak yang berkepentingan.

special thanks to

Selama proses penelitian dan penulisan Tesis ini, banyak pihak yang telah memberikan bantuan dan dukungan untuk penulis. Penulis mengucapkan terima kasih dan apresisasi secara khusus kepada:

1. Kedua orang tua tercinta, Bapak H. Afifan dan Ibu Hj. Rohmah Sururoh serta adik-adik tersayang Samlatul Izzah dan Nadial Khusna yang telah memberikan doa, dukungan, motivasi dan semua hal yang tak pernah berhenti sampai terselesainya Tesis.
2. Teman-teman seperjuangan dalam kelompok Tesis yang sama, Pengolahan Citra, yaitu Bayu Kharisma Putra, Reja Agusta Jannatul Firdaus dan Galandaru Swalaganata atas saran, kritik, semangat dan dukungan selama penelitian.
3. Yunita Nur Afifah, Rita Ayu Ningtyas, Rahmat Nursalim, Harmerita, dan teman- teman kelas lainnya yang telah bersama-sama berjuang dan saling membantu dalam proses perkuliahan.
4. Sahabat kuliah selama masa studi di pascasarjana Matamatika ITS yaitu Ruvita Iffahtur Pertiwi, I Gede Eva Purwanta dan Aprilia Divi Yustita yang selalu memberi dukungan hingga terselesainya Tesis ini, menjadi tempat *curhat* dan pendengar yang baik bagi penulis dalam banyak hal. Terima kasih atas persahabatan dan kenangan selama masa studi di pascasarjana ITS. Sampai jumpa lagi di lain waktu.
5. Luthfi Ahmad Muchlashi, M Faizal Winaris beserta teman-teman kos lainnya di Dharmahusada Indah Utara XIV/7C atas segala bentuk semangat dan dukungannya kepada penulis dalam kondisi apapun.

Tentu saja masih banyak pihak lain yang turut berkontribusi andil dalam penyelesaian Tesis ini yang tidak bisa penulis sebutkan satu persatu. Semoga Allah membalas dengan balasan yang lebih baik bagi semua pihak yang telah membantu penulis. *Jazakumullahu khairon Katsira*.

Surabaya, Januari 2017

Penulis

BAB I
PENDAHULUAN

1.1 Latar Belakang

Video *surveillance* merupakan topik penelitian dalam visi komputer yang mencoba untuk mendeteksi, mengenali dan melacak objek dari urutan gambar. Tahap selanjutnya setelah proses tersebut adalah membuat suatu usaha untuk memahami dan menjelaskan perilaku objek dengan mengganti metode tradisional seperti pemantauan dengan kamera oleh operator manusia. Pelacakan objek merupakan tugas penting dalam banyak aplikasi visi komputer seperti pengawasan, navigasi kendaraan, dan navigasi robot otonom. Ada banyak hal-hal yang dilakukan dalam proses pelacakan akan tetapi hal ini bergantung pada tingkat kebutuhan pengguna dalam mengambil informasi objek pada citra saat pertama muncul dalam video. [10].

Pelacakan objek dapat didefinisikan sebagai masalah dalam memperkirakan lintasan suatu objek pada bidang gambar yang bergerak di sekitar sebuah *scene*. Dengan kata lain, sebuah pelacak memberikan label yang konsisten pada objek yang dilacak dalam *frame* yang berbeda dari video. Pelacakan objek secara umum merupakan permasalahan yang cukup rumit. Hal ini dapat timbul karena adanya pergerakan objek yang cepat, perubahan pola penampakan objek, struktur objek yang *non-rigid*, oklusi, kamera yang bergerak dan lainya [14].

Salah satu masalah sulit lainnya adalah melakukan pelacakan untuk objek pada jarak jauh sehingga objek cenderung terlihat kecil (*small object*). Objek kecil memiliki informasi yang sedikit dalam proses pelacakan. Apabila hal ini terjadi proses pelacakan menjadi kurang presisi karena objek kecil tidak memuat informasi yang cukup untuk diidentifikasi dan dapat dianggap sebagai *noise* [6]. Salah satu cara untuk mengatasi permasalahan tersebut adalah mengintegrasikan metode pelacakan dengan teknik super- resolusi. Super-resolusi adalah sebuah proses pembangunan citra resolusi tinggi yang dibangun dari serangkaian citra dengan resolusi rendah [11]. Super-resolusi dapat diintegrasikan ke sistem pelacakan untuk

meningkatkan kualitas visual dari objek dan membantu meningkatkan presisi pelacakan.

Beberapa penelitian sebelumnya yang berkaitan dengan pelacakan dan super-resolusi adalah penelitian dari Hao Sun yang berjudul "*Location and Super-resolution Enhancement of License Plates Based on Video Sequences* ". Dalam penelitiannya, Hao Sun dkk menggunakan template matcing untuk proses pelacakan dan super-resolusi untuk meningkatkan kualitas citra dengan rekonstruksi *Projection Onto Convex Sets* (POCS) dari hasil pelacakan [4]. Integrasi antara pelacakan dan super-resolusi juga pernah disusun oleh Olegs Mise dan Toby P. Breckon yang berjudul " *Super-Resolution Imaging Applied to Moving Targets in High Dynamic Scenes*". Mereka menerapakan metode Sum of Absolute Difference (SAD) dan teknik estimasi gerak *Gradient-Descent* dalam proses pelacakan[7]. Sementara itu, penelitian yang berkaitan dengan pelacakan objek kecil pernah dilakukan oleh Daviesy, Palmery dan Mirmehdi yang berjudul "*Detection and Tracking of Very Small Low Contrast Objects*". Dalam penelitian tersebut, mereka sukses menggunakan kombinasi *wavelet* berdasarkan kerangka dari Kalman untuk melakukan pelacakan dengan multiple objek kecil [2].

Berdasarkan uraian di atas, penelitian ini akan mengintegrasikan pelacakan objek dengan menggunakan metode *Adaptive Particle Filter* berbasis citra super-resolusi . Teknik super-resolusi yang digunakan adalah teknik super-resolusi *multi-frame* untuk menghasilkan citra resolusi tinggi yang lebih baik. Selain digunakan untuk meningkatkan kualitas visual, super-resolusi pada proses pelacakan digunakan untuk memperoleh informasi yang lebih dari objek kecil dengan tujuan agar pergerakan objek dapat terdeteksi sehingga pelacakan objek menjadi lebih presisi.

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, rumusan masalah yang dibahas dalam penelitian ini adalah sebagai berikut.

1. Bagaimana menerapkan proses pelacakan berbasis citra super-resolusi?

2. Bagaimana menerapkan dan mengimplementasikan pelacakan objek kecil menggunakan metode *Adaptive Particle Filter* dari serangkaian citra super-resolusi?
3. Bagaimana menganalisa dan mengetahui tingkat akurasi pelacakan objek kecil menggunakan metode *Adaptive Particle Filter* dari serangkaian citra super-resolusi?

1.3 Batasan Masalah

Pada penelitian ini, batasan masalah yang akan digunakan adalah sebagai berikut.

1. Proses pelacakan dilakukan pada objek tunggal (*single object*) berdasarkan pemilihan dengan ROI (*Region Of Interest*).
2. Objek yang akan dilacak bersifat *rigid*.
3. Fokus objek pada penelitian ini adalah objek kecil (*small objek*) dan tidak adanya kasus oklusi.
4. Keadaan kamera dalam proses perekaman objek bergerak adalah statis.
5. Ukuran atau jumlah piksel dari objek kecil dalam kasus yang dikaji sangat relatif dan tergantung dari ukuran *frame* video yang digunakan. Dalam penelitian ini, ukuran objek yang akan dilacak berkisar antara $\frac{1}{1000}$ sampai $\frac{1}{100}$ ukuran *frame* video yang digunakan.

1.4 Tujuan

Berdasarkan rumusan masalah diatas, langkah awal tujuan dari penelitian ini adalah menerapkan proses pelacakan berbasis citra super-resolusi dalam sebuah block diagram. Dari block diagram tersebut, penelitian dilanjutkan dengan menerapkan dan mengimplementasikan pelacakan pada objek kecil dengan menggunakan metode *Adaptive Particle Filter* dari serangkaian citra super-resolusi. Kemudian, hasil implementasi tersebut akan diuji dengan beberapa video uji coba dan dianalisa untuk mengetahui tingkat akurasi pelacakan.

1.5 Manfaat

Berdasarkan tujuan penelitian ini, beberapa manfaat yang dapat diperoleh dan kemudian dapat diaplikasikan dalam beberapa bidang tertentu diantaramya adalah,

1. Bidang militer

   Sistem pengendalian otomatis untuk melacak sasaran atau posisi target dengan presisi yang lebih tinggi pada penembakan misil.

2. Bidang *Surveillance*

   Pelacakan objek dari jarak jauh dan selanjutnya digunakan untuk mengetahui informasi objek seperti plat nomor, kecepatan objek, jarak dan lain-lain.

3. Bidang olah raga,

   Pemanfaatan dalam *Hawk-eye* atau Sistem yang digunakan untuk melacak posisi bola sehingga akan sangat membantu tugas dari hakim garis.

BAB II
KAJIAN PUSTAKA DAN DASAR TEORI

Bab ini akan menjelaskan uraian singkat mengenai beberapa teori yang digunakan untuk memudahkan dalam pembahasan meliputi penjelasan singkat tentang pelacakan objek beserta penelitian sebelumnya yang terkait dengan penelitian ini, *Adaptive Particle Filter* sebagai metode pelacakan, algoritma *Block Matching* dan super-resolusi . Terlebih dahulu diberikan uraian mengenai penelitian yang telah dilakukan sebelumnya.

2.1 Penelitian Sebelumnya

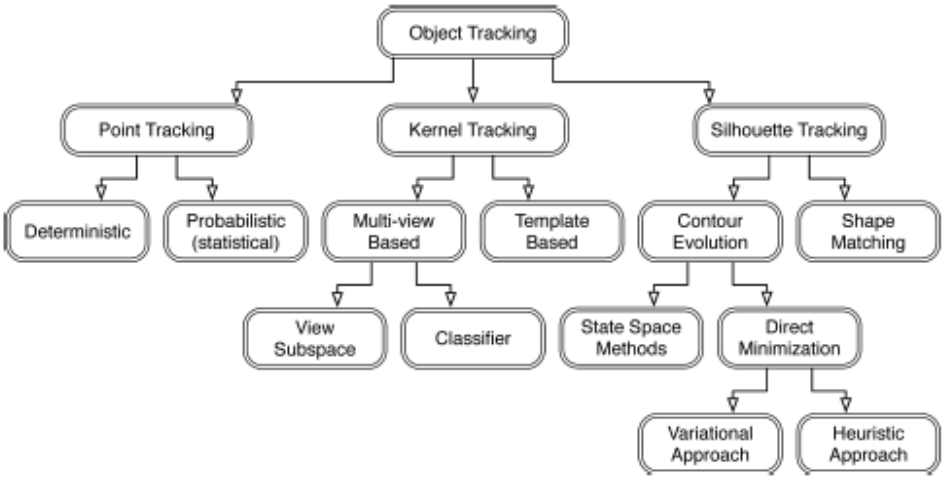
Beberapa penelitian sebelumnya yang terkait dengan pelacakan dan super-resolusi adalah penelitian dari Sun Hao, Luo Lin, Zhou Weiping dan Luo Limin pada tahun 2009 yang berjudul "*Location and Super-resolution Enhancement of License Plates Based on Video Sequences* ". Mereka menggunakan *template matching* untuk proses pelacakan dan *Projection Onto Convex Sets* (POCS) untuk proses super-resolusi. Namun penelitiannya fokus pada peningkatan kualitas citra dengan super-resolusi dari beberapa hasil pelacakan citra objek. Dalam hal ini mereka membutuhkan waktu komputasi yang lama dalam proses super-resolusi.

Selain itu, penelitian dari Olegs Mise dan Toby P. Breckon mengintegrasikan super-resolusi berbasis kombinasi metode Sum of Difference (SAD) dan teknik estimasi gerak Gradient-Descent pada proses pelacakan. Pendekatan ini secara signifikan memperkuat proses pelacakan pada objek terutama pada objek yang bergerak dengan cepat meskipun diterapkan pada performa *hardware* yang rendah. Namun, konsep tersebut tidak digunakan pada kasus objek yang berukuran kecil.

Berdasarkan beberapa penelitian yang telah dilakukan dalam keterangan diatas, penelitian ini melakukan integrasi pelacakan dengan metode *Adaptive Particle Filter* berbasis super-resolusi untuk objek yang berukuran kecil. Subbab selanjutnya akan menjelaskan dasar teori yang digunakan dalam penelitian ini.

2.2 Pelacakan Objek

Pelacakan objek merupakan hal penting dalam bidang *computer vision* dan dibutuhkan dalam aplikasi visual. Perkembangan komputer bertenaga tinggi, keterjangkauan harga kamera video dan meningkatnya kebutuhan analisis video secara langsung telah menghasilkan banyak perkembangan dalam algoritma pelacakan objek. Tujuan dari pelacakan objek adalah untuk menghasilkan lintasan dari sebuah objek dari waktu ke waktu dengan menempatkan posisinya di setiap frame video. Secara sederhana, pelacakan dapat diartikan sebagai cara menemukan objek bergerak selama waktu tertentu dalam sebuah video. Dalam pendekatan pelacakan, obyek dapat direpresentasikan menggunakan bentuk titik , persegi , elips dan lain-lain. Model yang dipilih untuk mewakili bentuk objek yang terbatas pada jenis gerak dan deformasi objek dapat dilalui. Jika sebuah objek direpresentaikan sebagai sebuah titik maka model yang dapat digunakan hanya model translasi. Dalam kasus pelacakan dengan representasi bentuk geometri, ellips dapat mengaproksimasi gerakan dari objek yang tetap (*rigid*). Untuk objek yang tidak tetap (*non-rigid*), silhouette atau kontur adalah representasi yang diskriptif dan kedua model parametrik dan nonparametrik dapat digunakan untuk menspesifikasikan gerakannya [9].



Gambar 2.1. Taksonomi dari metode pelacakan

Pelacakan dapat diformulasikan sebagai korespondensi dari objek terdeteksi oleh *point* yang melintasi bagian frame. Menurut Yilmaz dalam papernya yang membahas secara umum tentang pendekatan pelacakan objek, pelacakan berbasis *point* dibagi menjadi dua kategori seperti pada Gambar 2.1 yaitu metode dengan pendekatan deterministik dan probabilistik. Dalam penelitian ini, metode pelacakan yang digunakan adalah pelacakan berbasis *point* dengan pendekatan statistika. Pendekatan statistika dalam hal ini menggunakan pendekatan ruang keadaan untuk memodelkan karakteristik objek seperti posisi objek dalam gambar, lintasan dan korelasi. Untuk kasus objek tunggal, beberapa metode yang sering digunakan adalah Kalman Filter dan Particle Filter.

2.3 *Adaptive Particle Filter*

Adaptive Particle Filter merupakan metode pengembangan dari Particle Filter dengan menerapkan model pergerakan adaptive untuk mendapatkan pendekatan distribusi yang lebih baik. Sebuah kombinasi *template correlation* dengan kekontinuan gerakan dan perbaikan lintasan dalam likelihood observasi digunakan untuk lebih menyaring gangguan-gangguan yang ada [4]. Ide utamanya untuk mendekati distribusi probabilitas posterior oleh himpuan bobot partikel. Setiap partikel merepresentasikan satu hipotetis keadaan objek. Rata-rata keadaan sebuah objek diestimasi pada setiap langkah dengan rata-rata bobot dari semua partikel. Biasanya resampling digunakan untuk meringankan degenerasi partikel [4].

Suatu matrik dua dimensi dapat merepresentasikan sebuah citra dimana setiap nilai dalam matrik tersebut mewakili sebuah nilai intesitas pencahayaan. Berdasarkan pada paper Yu Huang dan Joan llach pada tahun 2008, vektor keadaan merupakan variabel keadaan yang menggambarkan perilaku suatu sistem. Jika sebuah objek citra didefinisikan sebagai $X = (x, y)$ dengan (x, y) adalah pusat objek maka model ruang keadaan dalam pelacakan objek dapat diformulasikan sebagai berikut

$$X_{t+1} = f(X_t, \mu_t)$$
$$Z_t = g(X_t, \varepsilon_t)$$

dimana X_t representasi vektor keadaan objek yang mencirikan perubahan keadaan pada objek, Z_t adalah vektor observasi, sedangkan μ_t dan ε_t secara berurutan merepresentasikan estimasi *error* dan *noise*. Sebuah estimasi gerak objek dinotasikan V_t dengan sebuah kesalahan prediksi μ_t maka model diatas dapat dibentuk kembali menjadi:

$$X_{t+1} = X_t + V_t + \mu_t \tag{2.1}$$

Untuk melakukan pembobotan partikel dalam observasi, proses perhitungan dapat menggunakan fungsi *likelihood* yang didefinisikan sebagai berikut,

$$P(Z_t|X_t) = P\left(Z_t^{int}\middle|X_t\right)P(Z_t^{mot}|X_t)^{O_{t-1}}P\left(Z_t^{nj}\middle|X_t\right)^{t-O_{t-1}} \tag{2.2}$$

dengan $Z_t = \left\{Z_t^{int}, Z_t^{mot}, Z_t^{trj}\right\}$. Kemudian *intensity measurement* Z_t^{int}diasumsikan menjadi *independent* dari yang lain seperti *motion measurement* Z_t^{mot} atau *trajectory measurement* Z_t^{trj}, jika objek tidak terdeteksi pergerakannya maka inisialisasi $O_t = 0$ dan jika objek terdeteksi pergerakannya maka inisialisasi $O_t = 1$.

Sebuah *intensity measurement* dihitung berdasarkan kemiripan objek yang dilacak dengan partikel menggunakan *Sum of Square Differences* (SSD) yang didefinisikan sebagai berikut.

$$r(X_t) = \sum_{\chi \in W} [T(\chi) - I(\chi + X_t)]^2 \,,\; X_t \in Neib \tag{2.3}$$

dengan W adalah *window* objek (piksel yang berada dalam ROI) , χ atau $Neib$ adalah persekitaran kecil dari partikel X_t, $T(\chi)$ adalah template objek yang dijadikan acuan dan I adalah *frame* pada waktu ke t. Dari persamaan diatas (2.3) diperoleh J kandidat yang merupakan kandidat yang cocok di dalam *Neib* sehingga *likelihood* intensitas dapat dihitung dengan formulasi sebagai berikut.

$$P\left(Z_t^{int}\middle|X_t\right) = q_0 U(.) + C_N \sum_{j=1}^{J} q_j N(r_t, \sigma_t) \tag{2.4}$$

dengan

$U(.)$ merupakan sebuah kekacauan *background* yang diasumsikan berdistribusi *uniform*,

C_N adalah faktor normalisasi,

$q_j = \frac{1-q_0}{J}$ adalah peluang prior,

(dalam papernya, Huang menggunakan $q_0 = 0.5$).

Sebuah *motion measurement* dihitung berdasarkan $(\Delta x, \Delta y)$ atau selisih antara perubahan posisi partikel terhadap (x_{t-1}, y_{t-1}) dan kecepatan rata-rata objek pada waktu sebelumnya $(\overline{\Delta x}, \overline{\Delta y})$ dengan

$$\overline{\Delta x} = \sum_{s=t-k}^{t-1} \frac{|x_s - x_{s-1}|}{k}, \overline{\Delta y} = \sum_{s=t-k}^{t-1} \frac{|y_s - y_{s-1}|}{k} \tag{2.5}$$

(dalam papernya, Huang [4] menggunakan $k = 10$). Jika diketahui sebuah persamaan

$$d_{mot}^2 = (|\Delta x_t| - \overline{\Delta x})^2 + (|\Delta y_t| - \overline{\Delta y})^2, t > 1 \tag{2.6}$$

dengan varians σ_{mot}^2 maka *likelihood* gerakan dapat dinyatakan sebagai berikut

$$P(Z_t^{mot}|X_t) = \frac{1}{\sqrt{2\pi}\sigma_{mot}} \exp\left(-\frac{d_{mot}^2}{2\sigma_{mot}^2}\right) \tag{2.7}$$

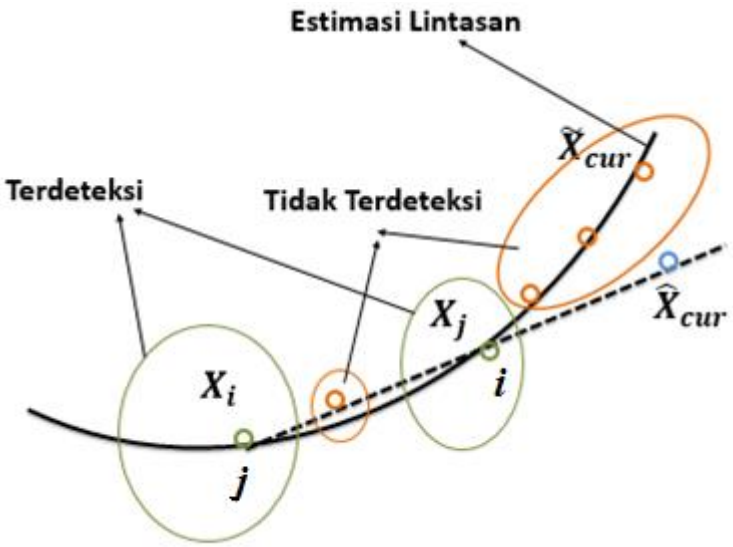
Sebuah *trajectory measurement* dapat diketahui berdasarkan *likelihood* lintasan. *Likelihood* ini tidak akan diproses dalam satu waktu dan keadaan yang sama dengan *likelihood* gerakan. Proses *likelihood* lintasan berdasarkan kedekatan partikel terhadap lintasan yang diperoleh dari posisi objek pada waktu sebelumnya. Fungsi lintasan dalam bentuk polinomial didefinikan sebagai berikut.

$$y = \sum_{i=0}^{m} a_i x^i \tag{2.8}$$

dengan a_i adalah koefisien polinomial dan m adalah order dari polinomial (dalam papernya [4], Huang menggunakan $m = 2$). Oleh karena itu, *likelihood* untuk *trajectory measurement* dapat dinyatakan sebagai

$$P\left(Z_t^{trj}\middle|X_t\right) = \frac{1}{\sqrt{2\pi}\sigma_{trj}} \exp\left(-\frac{d_{trj}/F}{2\sigma_{trj}^2}\right)^2 \tag{2.9}$$

dengan σ_{trj}^2 adalah varians untuk *likelihood* lintasan, $d_{trj} = |y - \sum_{i=0}^{m} a_i x^i|$, $F = \lambda_f^{t0}$ dengan λ_f adalah rasio $0 < \lambda_f < 1$ (dalam papernya [4], Huang menggunakan $\lambda_f = 0.9$) serta t_0 adalah banyaknya kejadian pergerakan objek tidak terdeteksi oleh *motion estimation* pada frame sebelumnya (Ilustrasinya pada Gambar 2.2 ditunjukkan dengan warna kuning).



Gambar 2.2. Ilustrasi *trajectory fitting*

Jika pergerakan objek dapat dideteksi pada *frame* sebelumnya $cur - 1$ ($O_{cur-1} = 1$) maka kita dapat mengestimasi keadaan dengan menghitung rata-rata semua partikel. Sementara itu apabila memenuhi kondisi sebaliknya, kita pilih salah satu atau lebih partikel yang memiliki bobot paling besar. Jika terjadi kondisi tersebut ($O_{cur-1} = 0$) maka kita akan memperbaiki lintasan dengan memproyeksikan salah satu prediksi posisi benda ke estimasi lintasan. Ilustrasi untuk perbaikan gerakan dapat dilihat pada Gambar 2.2. Gambar tersebut menunjukan dua posisi ketika pergerakan benda dapat terdeteksi yaitu X_j pada frame j dan X_i pada frame i dengan ($i > j$). Sebuah estimasi perbaikan posisi pada lintasan dapat dinyatakan sebagai

$$X_{cur} = \left(1 - \lambda_f^{t\_o}\right)\hat{X}_{cur} + \tilde{X}_{cur} * \lambda_f^{t\_o} \tag{2.10}$$

dengan sebuah prediksi posisi objek

$$\hat{X}_{cur} = X_i + \left(X_i - X_j\right) * \frac{cur-i}{i-j} \text{ untuk } i > j \tag{2.11}$$

Proyeksi dari $\hat{X}_{cur}$ yaitu $\tilde{X}_{cur}$ merupakan titik pada lintasan yang terdekat $\tilde{X}_{cur}$. Setelah mendapatkan posisi benda pada *frame* tersebut kemudian proses pelacakan objek dilanjutkan pada *frame* berikutnya. Dari beberapa penjelasan diatas, prosedur

secara umum tentang algoritma *Adaptive Particle Filter* yang pernah diusulkan sebelumnya oleh Huang [4] dapat ditunjukan dalam Gambar 2.3

Algoritma *Adaptive Particle Filter*

Mulai

Membangkitkan himpunan partikel $\left\{X_t^{(i)}, \pi_t^{(i)} | i = 1, \ldots, n\right\}$ pada saat t kemudian pada saat $t+1$ kita proses langkah-langkah sebagai berikut,

Estimasi

Mengestimasi gerakan V_{t+1} dan memprediksi *error* μ_{t+1}.

Jika $O_t = 1$

Untuk $i = 1 \ldots . n$

$X_{t+1}^{(i)} \sim N(X_t^{(i)} + V_{t+1}, \mu_{t+1})$

Selesai

Jika tidak

$V_{t+1} = 0, \mu_{t+1} = max(\mu_{t+1})$.

Untuk $i = 1 \ldots . n$

$X_{t+1}^{(i)} \sim N(X_t^{(i)} + V_{t+1}, \mu_{t+1})$

Selesai

Selesai

***Update* Bobot**

Untuk $i = 1 \ldots . n$

Perhingan bobot tiap partikel $\pi_{t+1}^{(i)} = P(Z_{t+1} | X_{t+1}^{(i)})$ dengan persamaan (2.2) sesuai dengan kondisi yang terjadi.

Selesai

Resample

Ganti $\left\{\left(X_{t+1}^{(i)}, \pi_{t+1}^{(i)}\right) \middle| i = 1 \ldots . n\right\}$ dengan $\{(\tilde{X}_{t+1}^{(i)}, 1/n) | i = 1, \ldots n\}$

***Update* posisi**

Jika $O_t = 1$

Hitung rata-rata semua bobot partikel dan update posisi.

Jika tidak

Pilih partikel (satu atau lebih) dengan bobot maksimum dan hitung rata-ratanya. Kemudian proyeksikan prediksi posisi yang diperoleh dari persamaan (2.11) ke dalam sebuah lintasan terdekat dengan persamaan (2.10)

Selesai

Selesai

Gambar 2.3. Prosedur Algoritma *Adaptive Particle Filter*

2.4 Algoritma *Block Matching*

Block matching merupakan salah satu algoritma *motion estimation* yang digunakan untuk mendapatkan vektor gerakan dari objek citra. *Block matching* merupakan salah satu dari algoritma *motion estimation* dan efisien dari beberapa teknik *motion estimation* lainnya [1. Ide utama dari Blok Matching adalah membagi frame menjadi matriks (*macro block*) yang kemudian dibandingkan dengan blok yang sesuai dengan sekitaran yang berdekatan pada *frame* sebelumnya untuk membuat vektor yang mengatur gerakan dari *macro block* dari satu lokasi ke lokasi lain di *frame* sebelumnya. Pencocokan satu *macro block* dengan *macro block* lain didasarkan pada output dari fungsi biaya tertentu [1]. Beberapa fungsi biaya (*cost function*) yang paling populer dan memiliki performa ringan secara komputasi adalah Mean Absolute Difference (MAD) seperti pada Persamaan berikut

$$MAD = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left|C_{ij} - R_{ij}\right| \tag{2.12}$$

Sementara itu, Fungsi biaya yang lainnya adalah Mean Square Error (MSE) yang didefinisikan sebagai berikut

$$MSE = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(C_{ij} - R_{ij}\right)^2 \tag{2.13}$$

dengan

N adalah ukuran *macro blok*,

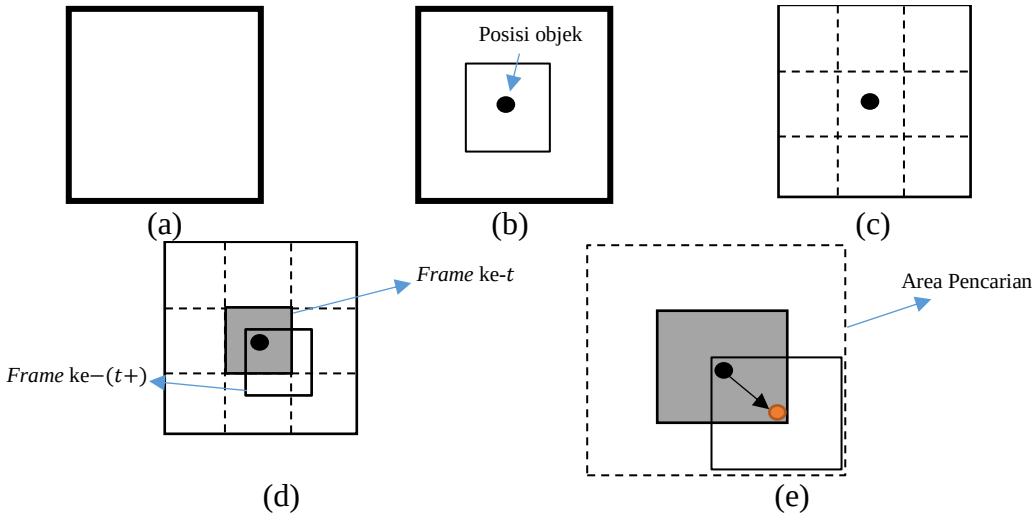
C_{ij} adalah *macro blok* sekarang,

R_{ij} adalah *macro blok* yang dijadikan acuan.

Proses pencocokan antar blok dengan fungsi biaya terkecil akan dipilih sebagai blok yang sesuai kemudian menentukan vektor gerakannya. Ilustrasi yang menunjukkan langkah-langkah algoritma *Block Matching* dapat ditunjukkan seperti pada Gambar 2.4.

Algoritma *Block Matching* yang telah banyak diimplementasikan antara lain *Exhaustive Search* (ES), *Three Step Search* (TSS), *New Three Step Search*

(NTSS), *Simple and Efficient* TSS (SES), *Four Step Search* (4SS), *Diamond Search* (DS) dan *Adaptive Rood Pattern Search* (ARPS) [1]. Metode pencarian yang digunakan dalam penelitian ini adalah Exhaustive Search dengan ilustrasi proses pencariannya disajikan dalam Gambar 2.4

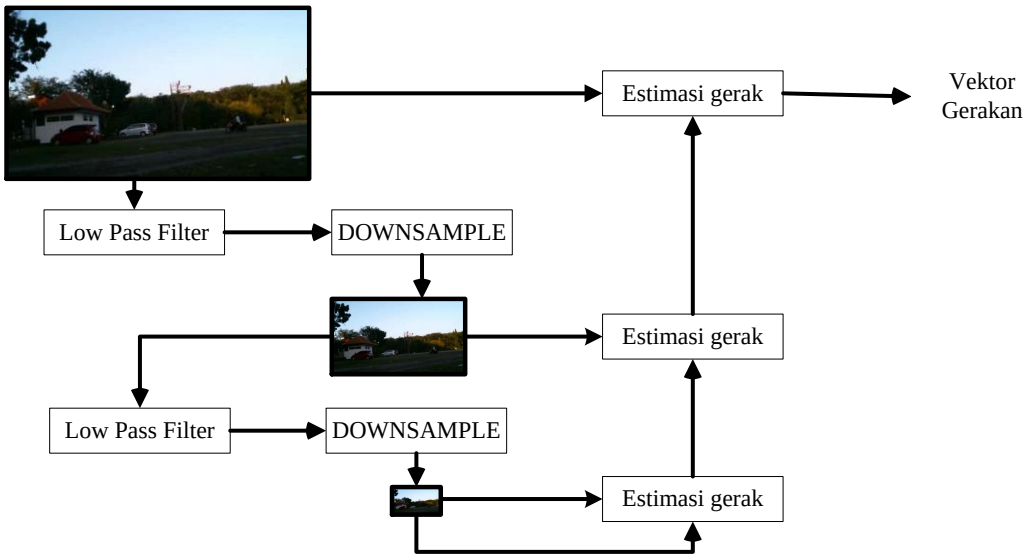


Gambar 2.4. Langkah-langkah dalam algoritma *Block Matching*. (a) Input image *frame* ke-k. (b) Menentukan blok besar berdasarkan posisi objek. (c) Membagi blok besar menjadi beberapa blok kecil. (d) Menentukan blok pada *frame* ke-$k+1$. (e) Pencarian posisi blok yang sesuai sepanjang area pencarian

Exhaustive Search menawarkan hasil yang berkualitas tinggi namun memiliki waktu komputasi yang tinggi. Waktu komputasi menjadi lebih tinggi seiring dengan luasnya area pencarian. Oleh karena itu, Algoritma hirarkis menawarkan pendekatan yang baru, baik cepat dan efisien [8].

Hierarchical Block Matching merupakan salah satu pendekatan yang lebih luas untuk diterapkan dalam bidang *motion vector*. Algoritma tersebut menawarkan hasil yang berkualitas tinggi, efisien dengan biaya komputasi yang rendah serta banyak variasi menjadikannya salah satu dari algoritma yang paling banyak digunakan untuk berbagai macam aplikasi. *Hierarchical Block Matching* mereduksi ukuran dan resolusi citra menjadi lebih kecil. Citra akan direduksi dengan faktor skala 2 atau direduksi sebanyak dua kali dengan ukuran setengah kali ukuran sebelumnya sehingga membentuk suatu piramida citra. Untuk

menghilangkan pengaruh *noise* pada level tertinggi (level 0), piramida citra dibangun dengan menggunakan *low pass filter*. Gambaran umum algoritma Hierarchical *Block Matching* dapat ditunjukan seperti pada Gambar 2.5**.**

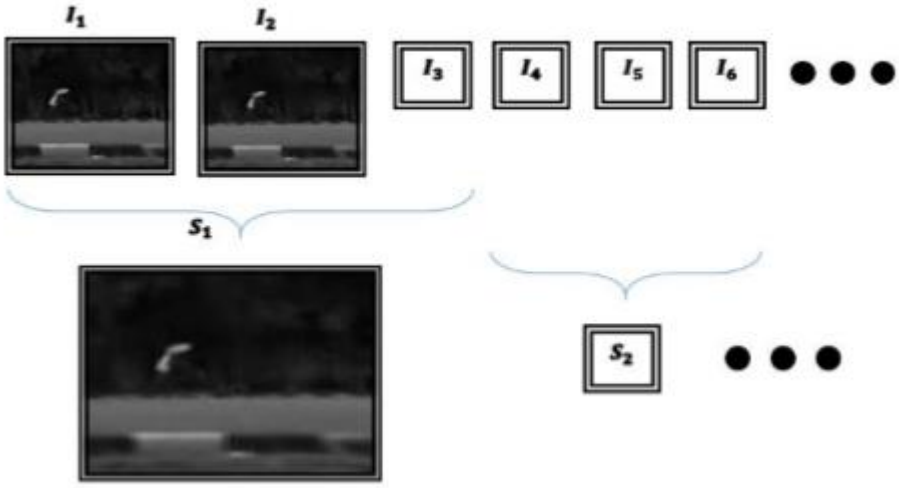


Gambar 2.5. Diagram alir algoritma Hierarchical *Block Matching*.

Setelah piramida terbentuk, vektor gerakan akan diestimasi menggunakan *Block Matching* dengan metode pencarian *Exhaustive Search* pada masing-masing tingkatan. Pencarian blok yang sesuai berdasarkan algoritma pada fungsi biaya yang telah didefinisikan pada persamaan 2.12 atau persamaan 2.13.

2.4.1 Super Resolusi

Super-resolusi adalah sebuah proses citra yang beresolusi tinggi yang dibangun dari satu set pergeseran subpiksel citra dengan resolusi rendah. Teknik untuk membangun citra super-resolution dapat dilakukan dua cara yaitu *single-frame* dan *Multi-frame* gambar super-resolusi. Teknik super-resolusi *single-frame* merupakan sebuah metode untuk menghasilkan gambar resolusi tinggi dari satu citra resolusi rendah. Sedangkan teknik super-resolusi *multi-frame* adalah sebuah metode untuk menghasilkan resolusi tinggi (HR) gambar dari beberapa gambar resolusi rendah atau *scene* yang sama kemudian meningkatkan resolusi citra dengan menggabungkan informasi [9].



Gambar 2.6. Proses kontruksi *frame* super-resolusi dari serangkaian *frame* resolusi rendah.

Berdasarkan survei yang telah dilakukan Setiyono pada tahun 2012 kolaborasi PBIM dan POCS menghasilkan citra super-resolusi yang baik [11]. Oleh karena itu, penelitian ini akan menggunakan teknik super-resolusi *multi-frame* dengan metode *Phased Based Image Mathching* (PBIM) untuk proses registrasi dan *Projection Onto Covex Sets* (POCS) untuk proses rekonstruksi.

2.5 Metode *Phased Based Image Matching* (PBIM)

Registrasi citra merupakan suatu proses untuk mendapatkan nilai pergeseran dan rotasi piksel dari serangkaian citra beresolusi rendah, melibatkan dua citra atau lebih, yang memiliki scene sama. Akurasi dalam tahap registrasi sampai sub-piksel citra sangat berpengaruh terhadap hasil super resolusi karena akan menjadi aspek penting dalam proses rekonstruksi untuk membangun citra super-resolusi yang lebih baik [11]. *Phased Based Image Matching* merupakan salah satu metode yang didasarkan pada transformasi Fourier diskrit dan sering digunakan dalam proses registrasi citra karena kehandalannya meskipun waktu komputasi yang dibutuhkan relatif kecil [11].

Dua buah citra dimisalkan $f(n_1, n_2)$ dan $g(n_1, n_2)$ dengan dimensi $N_1 \times N_2$. Asumsikan bahwa n_1 berkisar antara $-M_1$ sampai M_1 dan indeks n_2 berkisar antara $-M_2$ sampai M_2 sehingga n_1 berkisar dari $N_1 = 2M_1 + 1$ dan $N_2 = 2M_2 + 1$.

Berikut ini adalah persamaan transformasi Fourier dari citra $f(n_1, n_2)$ dan $g(n_1, n_2)$

$$F(k_1, k_2) = \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} f(n_1, n_2) e^{i2\pi\left(\frac{k_1 n_1}{N_1} + \frac{k_2 n_2}{N_2}\right)} \tag{2.14}$$

$$G(k_1, k_2) = \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} g(n_1, n_2) e^{i2\pi\left(\frac{k_1 n_1}{N_1} + \frac{k_2 n_2}{N_2}\right)} \tag{2.15}$$

dengan

1. $F(k_1, k_2)$ dan $G(k_1, k_2)$ merupakan transformasi Fourier Diskrit dari domain spasial citra $f(n_1, n_2)$ dan $g(n_1, n_2)$
2. $e^{i\theta F(k_1, k_2)}$ dan $e^{i\theta F(k_1, k_2)}$ merupakan *phase*
3. n_1, n_2adalah index domain spasial pada $f(n_1, n_2)$ dan k_1, k_2adalah indeks domain frekuensi pada $f(k_1, k_2)$ dimana $k_1 = -M_1, \ldots, M_1$, $k_2 = -M_2, \ldots, M_2$

Cross phase spectrum (*normalized cross spectrum*) didefinisikan sebagai

$$\hat{R}(k_1, k_2) = \frac{F(k_1, k_2)\overline{G(k_1, k_2)}}{|F(k_1, k_2)G(k_1, k_2)|} = e^{i\theta(k_1, k_2)} \tag{2.16}$$

dengan inversnya

$$\hat{r}(n_1, n_2) = \frac{1}{N_1 N_2} \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} \hat{R}(k_1, k_2) e^{i2\pi\left(\frac{k_1 n_1}{N_1} + \frac{k_2 n_2}{N_2}\right)} \tag{2.17}$$

Hasil dari metode ini berupa translasi piksel yang akan digunakan dalam proses rekonstruksi [7].

2.6 Algoritma *Projection Onto Convex Sets* (POCS)

Rekonstruksi citra pada Super Resolusi menyatakan proses pembangunan ulang dari serangkaian citra resolusi rendah untuk mendapatkan output berupa citra resolusi tinggi. Proses rekonstruksi akan memproyeksikan *grid* resolusi tinggi dari nilai pergerakan hasil proses registrasi. Salah satu metode rekonstruksi citra adalah POCS yang pertama kali diusulkan oleh Stark dan Oskui pada tahun 1989 dengan kehandalannya dalam mengatasi citra *noise* dan *blur* [12]. Konsep utama dalam

metode ini adalah mengestimasi citra super-resolusi yang dibatasi dalam sebuah himpunan konvek dan hasilnya diperoleh dengan proses iterasi [3].

Sebuah citra resolusi rendah $g(x,y)$ merupakan citra resolusi tinggi $f(x,y)$ dengan sebuah nilai degradasi oleh *point spread function* $h(x,y)$ dan *noise* $N(x,y)$ dengan model seperti persamaan sebagai berikut

$$g(x,y) = h(x,y)f\big(x+s_x, y+s_y\big) + N(x,y) \tag{2.18}$$

sehingga sebuah himpunan *convex* untuk setiap piksel dalam citra dapat dinyatakan sebagai

$$C_i = \{[g(x,y) - h(x,y)f(x,y)] \leq N(x,y)\} \tag{2.19}$$

Sementara itu, sebuah operator proyeksi yang memproyeksikan sebuah citra ke himpunan konvex dinyatakan dengan T_c, f_0 merupakan perbesaran citra dengan algoritma interpolasi dan perbesaran citra setelah iterasi ke-k dinyatakan sebagai f_k sehingga sebuah persamaan proyeksi rekonstruksi citra ke himpunan konvek dinyatakan sebagai

$$f_{k+1} = Tc_m Tc_{m-1} \ldots\ldots Tc_1 f_k \tag{2.20}$$

dengan $Tc_i = I + \lambda_i(Pc_i - I)$ dan $0 < \lambda_i < 2$ untuk $i = 1,2,\ldots m$. Persamaan diatas dapat dijalankan secara iteratif sampai memenuhi tingkat kendala *noise* pada citra resolusi rendah. Jika operator proyeksi disubstitusikan ke persamaan diatas maka didapatkan

$$f_{k+1} = f_k + \lambda_i \frac{g_i - h_i' f_k}{\|h_i\|^2} h_i^2 \tag{2.21}$$

dengan g_i adalah elemen ke-i dari vektor $g(x,y)$ dan h_i' adalah baris ke-i dari matrik $h(x,y)$. Proses iterasi akan terus berjalan sampai memenuhi tingkat kendala *noise* atau kriteria pemberhentian [11].

BAB III
METODA PENELITIAN

Bagian ini akan membahas uraian singkat mengenai tahapan penelitian yang dilakukan. Terlebih dahulu beberapa uraian akan dijelaskan tentang objek dan peralatan penelitian, tahapan penelitian yang akan dilakukan, proses integrasi pelacakan dengan super-resolusi yang digambarkan dalam blok diagram beserta penjelasannya.

3.1 Objek dan Peralatan Penelitian

Objek pengamatan merupakan hal atau sesuatu yang dijadikan sasaran pengamatan. Dalam hal ini yang dijadikan objek pengamatan adalah pergerakan suatu objek kecil dalam suatu rekaman video. Objek kecil dapat terjadi karena objek yang diamati dalam kondisi sangat jauh dari pengamat. Peralatan yang digunakan untuk pengamatan adalah kamera sebagai alat untuk mengambil atau merekam video. Kemudian video tersebut akan diolah dengan metode yang akan digunakan pada program perangkat lunak untuk pengujian dan simulasi pelacakan objek.

3.2 Tahapan Penelitian

Tahapan penelitian yang dilakukan dalam penelitian yang diusulkan adalah sebagai berikut.

1. Studi Literatur

   Dalam penelitian ini, studi literatur merupakan tahap untuk mengkaji lebih mendalam tentang pengolahan citra digital terutama tentang pelacakan pada objek dengan metode *Adaptive Particle Filter*, estimasi pergerakan objek dengan *Block Matching* beserta konsep super resolusi yang akan diimplementasikan pada pelacakan objek pada video. Pengkajian ini dilakukan dengan membaca buku, artikel atau jurnal yang berkaitan dengan kasus pelacakan pada objek kecil dan super-resolusi.

2. Akuisi data

   Akuisisi data dapat didefinisikan sebagai suatu proses untuk mengambil, mengumpulkan dan menyiapkan data sehingga memprosesnya untuk mendapatkan data yang dikehendaki. Akuisisi data berupa pengambilan data video pada kasus objek kecil. Data video dapat diperoleh dari internet berupa dataset atau pengambilan video langsung oleh peneliti seperti video pergerakan pesawat dari jarak jauh, video olah raga dan video pada kasus objek kecil lainnya.

3. Penerapan pelacakan objek pada video berbasis super-resolusi

   Proses pelacakan dalam penelitian ini mengintegrasikan metode pelacakan dengan super-resolusi. Super-resolusi digunakan untuk meningkat kualitas visual dan informasi dari objek citra yang akan dilacak. Super-resolusi menggunakan teknik super-resolusi *multi-frame* yang terdiri dua tahap yaitu proses registrasi kemudian dilajutkan proses rekonstruksi. Oleh karena itu, dalam penerapannya proses akan berjalan secara sekuensial dimana super-resolusi akan dijalankan sebelum proses pelacakan. Hasil dari tahap ini ditunjukkan dalam Gambar 3.1.

4. Implementasi pada MATLAB

   Setelah menerapkan pelacakan objek dengan metode *Adaptive Particle Filter* berbasis super-resolusi tahap selanjutnya adalah mengimplementasikan metode tersebut pada program pada perangkat lunak digunakan sebagai alat untuk mensimulasikan proses pelacakan pada objek bergerak.

5. Pengujian dan analisa hasil

   Pengujian akan dilakukan pada program yang telah dibangun dan menghitung prosentase akurasi metode *Adaptive Particle Filter* dari hasil super-resolusi. Video hasil pengamatan akan menjadi input dalam pengujian program kemudian ouput dari program adalah video dengan pelacakan objek. Kemudian menarik kesimpulan dari analisa yang didapatkan serta saran untuk penelitian lebih lanjut.

6. Publikasi.

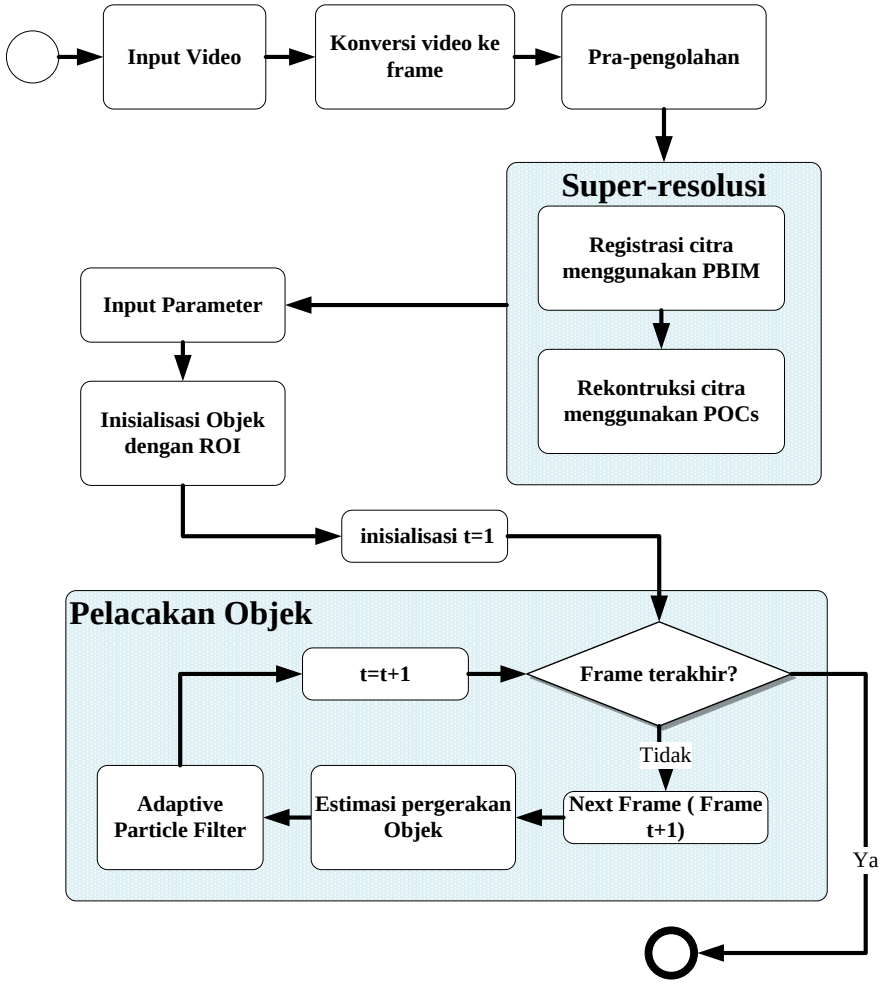
   Pada tahapan ini, penelitian yang telah dilakukan akan diseminasikan pada seminar internasional.

7. Penulisan Tesis

   Penulis akan menuliskan semua informasi yang dibutuhkan dan hasil yang telah diperoleh selama pengerjaan penelitian.

3.3 Block *Diagram*

Block diagram dari proses pelacakan objek dengan metode *Adaptive Particle Filter* berbasis super-resolusi dapat dilihat pada Gambar 3.1



Gambar 3.1 Tahapan pelacakan objek dengan *Adaptive Particel Filter* berbasis Super-resolusi

Penjelasan proses pelacakan berbasis super-resolusi pada Gambar 3.1 adalah sebagai berikut.

1. Input Video

   Input video yang digunakan berupa rekaman video *offline*. Video yang akan dipilih berdasarkan kriteria video yang diperoleh pada akuisisi data yaitu, video pergerakan objek kecil (*small object*) dengan karakteristik masing-masing video yang berbeda-beda.

2. Konversi video ke *frame*

   Berdasarkan input video yang telah dipilih, video tersebut akan dikonversi menjadi beberapa rangkaian *frame* untuk pengolahan dalam proses pelacakan objek yang berjalan secara berurutan, *frame by frame*.

3. Pra-pengolahan

   Tahap ini akan melakukan ekualisasi histogram yang diperlukan untuk mendapatkan histogram yang merata di setiap *frame* sehingga penampakan objek terlihat lebih jelas.

4. Super-resolusi

   Super Resolusi adalah teknik untuk mendapatkan objek citra yang beresolusi tinggi dari citra yang beresolusi rendah. Proses super-resolusi digunakan pada *frame* video untuk meningkatkan kualitas dan resolusi citra atau *frame*. Pada penelitian ini, proses super-resolusi menggunakan teknik super-resolusi *multi-frame* yang terdiri dari dua tahap yaitu

   a. Registrasi

      Registrasi citra pada super-resolusi dilakukan untuk mengetahui nilai pergeran dan rotasi dari serangkaian citra resolusi rendah. Proses registrasi citra mengambil beberapa buah citra referensi, misalnya mengambil tiga buah rangkaian citra beresolusi rendah dengan kriteria dua *frame* berikutnya dari citra yang akan di proses super-resolusi secara berurutan. Sedangkan pada bagian akhir dari *frame* video ($lastframe$-1 atau $lastframe$-2), proses registrasi akan mengambil dua buah *frame* sebelumnya dari cita yang akan

diproses super-resolusi. Kemudian hasil dari registrasi akan diproses lebih lanjut pada rekonstruksi citra.

b. Rekonstruksi

Proses rekonstruksi citra super-resolusi merupakan proses pembangunan ulang atau penyusunan ulang dari rangkaian citra resolusi rendah sebagai masukan untuk mendapatkan hasil berupa citra resolusi tinggi. Rekonstruksi citra menggunakan metode *Projection Onto Convex Sets* (POCS) kemudian hasilnya akan digunakan untuk proses pelacakan.

5. Inisialisasi Parameter

Terdapat tiga parameter yang dibutuhkan dalam pelacakan dalam penelitian ini yang meliputi parameter jumlah partikel, ukuran blok dan area pencarian.

6. Inisialisasi target ROI

ROI (*Region of Interest*) merupakan bagian dari citra yang dipilih sebagai area untuk memisah antara *foreground* and *background* (segmentasi). Pemilihan target ROI dipilih oleh user pada *frame* pertama sesuai dengan object yang akan dilacak. Setelah melakukan pemilihan target, posisi objek pada *frame* pertama dapat diketahui berdasarkan *centroid* objek dari ROI. Hasil dari proses ini akan digunakan sebagai masukan pada proses pelacakan.

7. Pelacakan Objek

Tahap ini terdapat beberapa proses yang diperlukan dalam pelacakan objek, diantaranya adalah sebagai berikut

a. Estimasi Pergerakan Objek

Estimasi pergerakan objek menggunakan metode *Hierarchical Block Matching* dengan metode pencarian *exhaustive* dan fungsi biaya MAD. Proses ini digunakan untuk mengetahui nilai vektor gerakan *frame* video pada saat *frame* t dan $t+1$ dimana nilai tersebut akan digunakan dalam proses lebih lanjut dalam metode *Adaptive Particle Filter*.

b. *Adaptive Particle Filter*

Adaptive Particle Filter digunakan sebagai metode pelacakan dimana pelacakan objek dilakukan berdasarkan objek citra yang dipilih dalam ROI. Proses diawali dengan membangkitkan sejumlah n partikel dan memberikan bobot awal. Bobot dari masing-masing partikel akan di-*update* tergantung keadaan objek dari hasil perhitungan berdasarkan fungsi *likelihood* pada persamaan 2.2. Sebuah nilai vektor gerakan objek dari hasil proses estimasi pergerakan objek digunakan untuk memindahkan partikel-partikel dari *frame* satu ke *frame* lainnya untuk observasi sesuai dengan model pada persamaan 2.1. Kemudian *Adaptive particle Filter* digunakan untuk menentukan *centroid* objek pada *frame* ke-$(t + 1)$ berdasarkan kandidat partikel yang memiliki bobot besar. Dalam metode metode ini, apabila pergerakan objek tidak terdeteksi maka proses pelacakan objek dilakukan dengan melakukan proyeksi dari hasil prediksi berdasarkan posisi objek pada *frame* sebelumnya. Proses ini dilakukan per-*frame* dari *frame* ke-2 sampai *frame* terakhir. Kemudian hasil pelacakan dengan target yang telah diidentifikasi akan ditandai dengan *bounding box*.

BAB IV
SUPER-RESOLUSI DAN PELACAKAN OBJEK

Bab ini akan membahas tentang perancangan pelacakan objek berbasis citra super-resolusi yang dilanjutkan dengan implementasi kedalam sebuah perangkat lunak untuk proses pengujian terhadap metode yang digunakan. Hal-hal yang akan dijelaskan dalam bab ini meliputi pembahasan tentang perancangan proses, perancangan perangkat lunak dan implementasinya kedalam perangkat lunak yang disertakan penjelasan mengenai cara untuk mendapatkan data keluaran yang sesuai dengan tujuan dari penelitian ini.

4.1 Perancangan Proses

Proses perancangan disusun dari beberapa tahapan penyusunan menggunakan metode yang akan digunakan. Secara umum proses pelacakan objek dengan *Adaptive Particle Filter* berbasis super-resolusi dapat ditunjukkan dalam diagram alir yang telah disajikan dalam Gambar 3.1. Selanjutnya, proses perhitungan dan rincian tentang perancangan proses super-resolusi dan pelacakan objek bergerak akan dijelaskan pada bagian berikut ini.

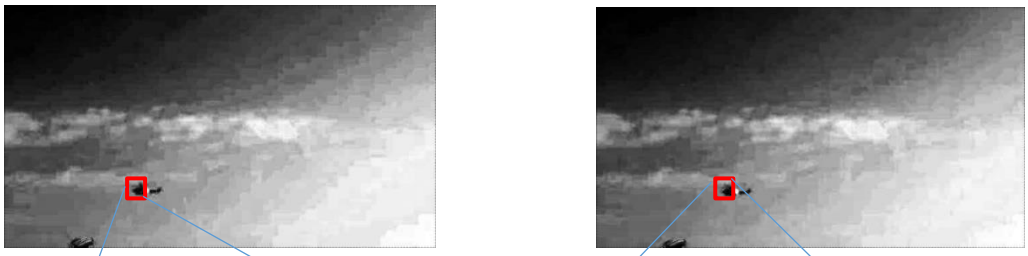
4.1.1 Super-resolusi

Proses super-resolusi citra berdasarkan tiga buah citra masukan yaitu *frame* ke-t sebagai citra reference atau citra yang akan diperbesar resolusinya dan dua buah *frame* di sekitarnya seperti pada Gambar 2.6. Dari citra masukan tersebut akan dicari nilai pergerakan translasinya dalam proses registrasi sebagai bahan untuk proses rekonstruksi citra super-resolusi.

a) Proses Registrasi Citra

Proses registrasi citra menggunakan PBIM dimana dua atau lebih buah citra masukan akan dicari nilai pergerakannya dengan cara membandingkan kedua citra tersebut. Sebuah citra mengalami perpindahan sub-piksel dalam ruang kontinyu dengan indeks bilangan real direpresentasikan sebagai $f_c(x_1 -$

$\delta_1, x_2 - \delta_2)$. Asumsikan $f(n_1, n_2)$ dan $g(n_1, n_2)$ merupakan sampel spasial dari $f_c(x_1, x_2)$ dan $f_c(x_1 - \delta_1, x_2 - \delta_2)$. Dua buah citra diambil dari citra *frame scene* yang sama dengan informasi sebagai berikut



$$f(n_1, n_2) = \begin{bmatrix} 2 & 3 & 1 \\ 1 & 1 & 2 \\ 3 & 1 & 1 \end{bmatrix} \quad \text{dan} \quad g(n_1, n_2) = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 1 & 2 \\ 2 & 3 & 1 \end{bmatrix}$$

Gambar 4.1. Ilustrasi citra objek sebagai serangkaian resolusi rendah

Sebuah tranformasi Fourier seperti dalam persamaan 2.14 dan persamaan 2.15 dari citra $f(n_1, n_2)$ dan $g(n_1, n_2)$ dinyatakan sebagai

$$F(k_1, k_2) = \sum_{n_1=0}^{2} \sum_{n_2=0}^{2} f(n_1, n_2) e^{i2\pi(\frac{k_1 n_1}{N_1} + \frac{k_2 n_2}{N_2})}$$

sehingga, untuk $k_1 = 0; k_2 = 0$

$$F(0,0) = \sum_{n_1=0}^{2} \sum_{n_2=0}^{2} f(n_1, n_2) e^{i2\pi(\frac{0 n_1}{3} + \frac{0 n_2}{3})}$$

$$= \sum_{n_1}^{2} \sum_{n_2}^{2} f(n_1, n_2).1$$

$$= f(0,0) + f(0,1) + f(0,2) + \ldots\ldots + f(2,2)$$

$$= 2 + 3 + 1 + 1 + 1 + 2 + 3 + 1 + 1$$

$$= 15$$

.

.

.

dan $k_1 = 2; k_2 = 2$

$$F(2,2) = \sum_{n_1=0}^{2}\sum_{n_2=0}^{2} f(n_1,n_2)e^{i2\pi(\frac{2n_1}{3}+\frac{2n_2}{3})}$$

$$= f(0,0)e^{i2\pi\left(\frac{2.0}{3}+\frac{2.0}{3}\right)} + f(0,1)e^{i2\pi\left(\frac{2.0}{3}+\frac{2.1}{3}\right)}+\ldots..+f(2,2)e^{i2\pi\left(\frac{2.2}{3}+\frac{2.2}{3}\right)}$$

$$= f(0,0) + f(0,1)(-0.5 - 0.866i)+\ldots..+f(2,2)2\frac{4}{3}$$

$$= 2 + (-1{,}5 - 2{,}5981i)+\ldots+\frac{8}{3}$$

$$= 0$$

Dengan cara yang sama, transformasi Fourier pada citra $f(x,y)$ dan $g(x,y)$ didapatkan hasil sebagai berikut

$$F = \begin{bmatrix} 15+0i & 1{,}5-0{,}866i & 1{,}5+0{,}866i \\ 1{,}5+0{,}866i & 0+0i & -1{,}5+4{,}3301i \\ 1{,}5-0{,}866i & -1{,}5-4{,}33i & 0+0i \end{bmatrix} \text{ dan}$$

$$G = \begin{bmatrix} 16+0i & -2+0i & -2+0i \\ 1+1{,}7321i & 1+0i & -3{,}5+0{,}866i \\ 1-1{,}7321i & -3{,}5-0{,}866i & 1+1{,}732i \end{bmatrix}$$

Dari hasil transformasi Fourier dua citra tersebut, selanjutnya adalah menghitung *Cross Spectrum Phase* seperti pada persamaan 2.16 sehingga didapatkan hasil sebagai berikut

$$\hat{R} = \begin{bmatrix} 1{,}0114-0i & -0{,}3936-0{,}1048i & -0{,}3936+0{,}1048i \\ 0{,}0113+0{,}0072i & 0{,}0884-0{,}9974i & -0{,}0024-0{,}0015i \\ 0{,}0113-0{,}0072i & -0{,}0024+0{,}0012i & 0{,}0884+0{,}9974i \end{bmatrix}$$

Dari nilai *Cross Spectrum Phase* yang telah diperoleh, selanjutnya nilai pergeseran objek dari dua citra dapat dicari dengan menghitung sudut fase

$$\hat{R}(k_1,k_2) = \frac{F(k_1,k_2)\overline{G(k_1,k_2)}}{|F(k_1,k_2)G(k_1,k_2)|}$$

$$= e^{i\theta(k_1,k_2)} \cong x + yi$$

dengan

$$e^{i\theta(1,1)} = 0{,}0884 - 0{,}9974i$$

sehingga, untuk $k_1 = 1; k_2 = 1 \rightarrow e^{i\theta(1,1)} = 0{,}0884 - 0{,}9974i$ berlaku

$$\tan\theta = \frac{y}{x}$$

$$= \frac{-0{,}9974}{0{,}0884} \cong 11{,}2828$$

$$\theta = arc\tan(11{,}2828)$$

$$= 1.4824\, radian \approx 84.97^\circ$$

Begitu juga berlaku untuk $k_1 = 0; k_2 = 1 \rightarrow e^{i\theta(0,1)} = 1{,}0114 - 0i$ berlaku

$$\tan\theta = \frac{y}{x}$$

$$\tan\theta = \frac{0}{1{,}0114}$$

$$\tan\theta = 0$$

$$\frac{y}{x} = 0$$

$$\theta = arc\tan(0)$$

$$= 0\, radian \approx 0^\circ$$

menghasilkan $\theta_1 = 84.97^\circ\ (0{,}472\pi)$ dan $\theta_2 = 84.97^\circ\ (0\pi)$

Dengan mempertimbangkan perbedaan sifat transformasi Fourier dari ruang kontinyu dan ruang diskrit, sebuah citra dalam domain frekuensi dapat ditunjukkan sebagai

$$F_c(k_1, k_2) = \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} f_c(x_1 - \delta_1, x_2 - \delta_2) e^{i2\pi\left(\frac{k_1(n_1-\delta_1)}{N_1} + \frac{k_2(n_2-\delta_2)}{N_2}\right)} \Bigg| \begin{matrix} x_1 = n_1 T_1 \\ x_2 = n_1 T_2 \end{matrix}$$

Dalam proses digitalisasi, persamaan diatas akan menjadi

$$F_c(k_1, k_2) = \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} f(n_1, n_2) e^{i2\pi\left\{\left(\frac{k_1 n_1}{N_1} + \frac{k_2 n_2}{N_2}\right) - \left(\frac{\delta_1}{N_1} - \frac{\delta_2}{N_2}\right)\right\}}$$

$$= F(k_1, k_2) e^{-\frac{i2\pi}{N_1} k_1 \delta_1} e^{-\frac{i2\pi}{N_2} k_2 \delta_2} \approx G(k_1, k_2)$$

sehingga didapatkan

$$\hat{R}(k_1, k_2) = e^{i\theta(k_1,k_2)} \cong e^{\frac{i2\pi}{N_1} k_1 \delta_1} e^{\frac{i2\pi}{N_2} k_2 \delta_2}$$

$$= e^{\frac{i2\pi}{N_1} k_1 \delta_1 + \frac{i2\pi}{N_2} k_2 \delta_2}$$

$$\theta(k_1, k_2) = \frac{i2\pi}{N_1} k_1 \delta_1 + \frac{i2\pi}{N_2} k_2 \delta_2 \qquad (4.1)$$

Apabila nilai θ yang telah kita peroleh disubstitusikan ke persamaan 4.1 maka untuk $k_1 = 0; k_2 = 1$ didapatkan

$$\theta(0,1) = \frac{2\pi.0}{3}\delta_1 + \frac{2\pi.1}{3}\delta_2$$
$$0 = \frac{2\pi.1}{3}\delta_2$$
$$0 = \delta_2$$

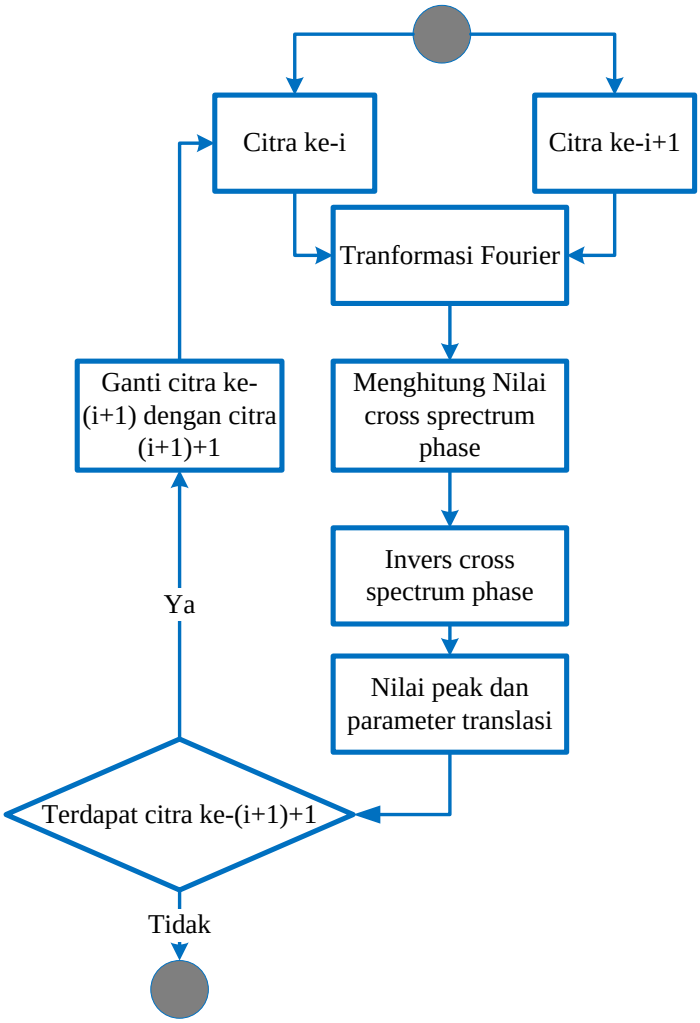
dan begitu juga untuk $k_1 = 1; k_2 = 1$ didapatkan

$$\theta(1,1) = \frac{2\pi.1}{3}\delta_1 + \frac{2\pi.1}{3}\delta_2$$
$$\theta(1,1) = \frac{2}{3}\pi(\delta_1 + \delta_2)$$
$$0{,}472\pi = \frac{2}{3}\pi(\delta_1 + \delta_2)$$
$$0{,}708 = (\delta_1 + \delta_2) \qquad (4.2)$$

Penyelesaian persamaan 4.2 menghasilkan $\delta_1 = 0{,}708$ dan $\delta_2 = 0$

Kemudian hasil dari *cross spectrum phase* dihitung nilai inversnya berdasarkan persamaan 2.17 yang menghasilkan nilai *peak*. Dua buah dari *scene* yang sama apabila dibandingkan memiliki *peak* yang menonjol dan tajam dari yang lain. Sebaliknya, dua citra yang tidak dari *scene* yang sama menghasilkan nilai *peak* yang turun secara tajam. Posisi *peak* menunjukkan lokasi piksel yang mengalami perpindahan translasi antara dua gambar secara signifikan.

Nilai perpindahan δ_1 dan δ_2 , akan disimpan untuk proses selanjutnya. Kemudian citra pertama sebagai *reference* dibandingkan dengan citra yang lain untuk mendapatkan nilai perpindahan translasi lainnya atau proses ini akan berulang sebanyak n citra masukan. Keseluruhan proses registrasi citra menggunakan PBM disajikan dalam diagram alir pada Gambar 4.2.



Gambar 4.2. Diagram alir registrasi citra menggunakan PBIM

b) Proses Rekonstruksi Citra

Proses rekonstruksi merupakan proses untuk mengestimasi nilai-nilai piksel yang belum diketahui dengan cara memproyeksikan piksel-piksel dari grid resolusi rendah pada grid resolusi tinggi. Proyeksi tersebut berdasarkan referensi nilai translasi yang telah didapatkan dalam proses registrasi terhadap operator proyeksi ke himpunan convex tertutup dengan menjadikan citra pertama dari rangkaian citra masukan sebagai acuan proyeksi.

Pada langkah pertama, sebuah proses *upsampling* (pengubahan posisi piksel citra masukan ke piksel resolusi tinggi yang disertai estimasi nilai piksel sekitarnya berdasarkan nilai translasi) untuk membentuk g dan sebuah

perbesaran citra pertama dengan algoritma interpolasi f_0 akan ditransformasi ke domain frekuensi.

Ilustrasi:

Seperti pada contoh sebelumnya dengan citra yang sama, sebuah citra resolusi rendah $A = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 0 \\ 1 & 3 & 2 \end{bmatrix}$ akan mengalami perbesaran citra dengan algoritma interpolasi

$$f_0 = \begin{bmatrix} 1{,}9951 & 2{,}3395 & 3{,}0787 & 2{,}7668 & 1{,}4039 & 0{,}7742 \\ 1{,}6907 & 1{,}9534 & 2{,}5185 & 2{,}3754 & 1{,}5240 & 1{,}1304 \\ 1{,}0383 & 1{,}1215 & 1{,}3040 & 1{,}5240 & 1{,}7816 & 1{,}8996 \\ 1{,}3694 & 1{,}2155 & 0{,}8853 & 1{,}0306 & 1{,}6513 & 1{,}9380 \\ 2{,}6840 & 2{,}2354 & 1{,}2624 & 0{,}8950 & 1{,}1331 & 1{,}2458 \\ 3{,}2911 & 2{,}7075 & 1{,}4398 & 0{,}8352 & 0{,}8938 & 0{,}9247 \end{bmatrix}$$

dan proses *upsampling* menghasilkan $g = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 3 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$. Untuk proses selanjutnya, kedua citra tersebut akan ditransformasikan ke domain frekuensi. Kemudian suatu *point spread function* $h = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}$ akan digunakan untuk proses degradasi citra pada f dengan operasi konvolusi $(f * h)$.

$$f(x,y) * h(x,y) = \sum_{a=0}^{n_1-1} \sum_{b=0}^{n_2-1} f(a,b)h(x-a, y-b)$$

$$= f(0,0)h(0,0) + f(0,1)h(0,-1) + f(1,0)h(-1,0) + f(1,1)h(-1,-1)$$

dengan $h(0,-1) \approx h(0,1), h(-1,0) \approx h(1,0)$ dan $h(-1,-1) \approx h(1,1)$

$$= 59{,}8577 \times 1 + (2.9464 - 4.9891i) \times (2) + \cdots$$

$$= 65.9996 - 9.5468i$$

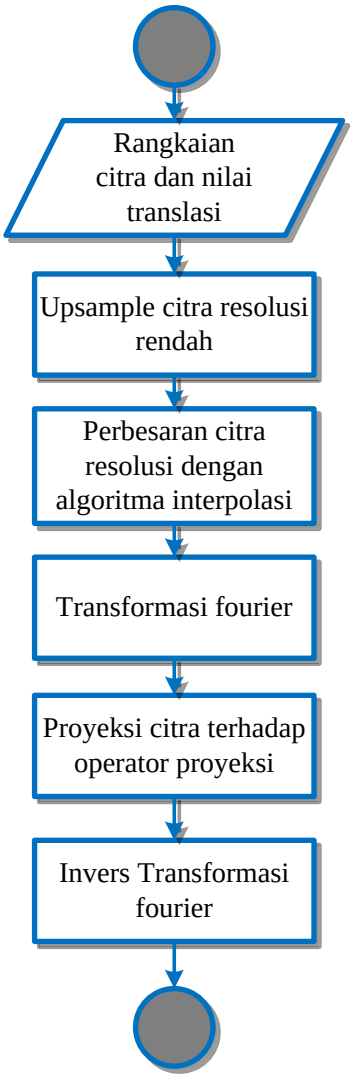
begitu juga proses untuk piksel setelahnya. Beberapa proses diatas merupakan proses perhitungan terhadap operator proyeksi

$$f_{k+1} = f_k + \lambda_i \frac{g_i - h_i f_k}{\|h_i\|^2} {h_i}^2$$

dengan λ_i bernilai antara 0 sampai 2 [11]. Proses perhitungan berjalan secara iteratif sampai kriteria pemberhentian terpenuhi. Tahapan terakhir adalah mengubah kembali citra hasil rekontruksi ke domain spasial dengan menghasilkan citra sebagai berikut,

$$f_1 = \begin{bmatrix} 2 & 2 & 3 & 3 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 & 1 \\ 1 & 1 & 1 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 2 & 2 \\ 3 & 2 & 1 & 1 & 1 & 1 \\ 3 & 3 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Keseluruhan proses rekontruksi citra menggunakan POCS disajikan dalam diagram alir pada Gambar 4.3.



Gambar 4.3. Diagram alir rekonstruksi citra menggunakan POCS

4.1.2 Pelacakan Objek

Proses pelacakan objek menggunakan metode *Adaptive Particle Filter* dimana pengolahan *frame* berdasarkan *frame* hasil super-resolusi. Objek yang akan dilacak sesuai area yang telah dipilih dengan ROI. Pelacakan objek dimulai dengan membangkitkan sejumlah n partikel yang disertakan bobot partikel. Kemudian proses dilanjutkan dengan estimasi pergerakan (*motion estimation*) untuk mendapatkan vektor gerakan dari objek yang bergerak dan prediksi error. Apabila pergerakan objek terdeteksi maka dilanjutkan dengan *update* bobot berdasarkan *likelihood* intesitas dan pergerakan. Apabila tidak terdeteksi pergerakannya maka *update* bobot berdasarkan *likelihood* intensitas dan lintasan. Bobot akan mengalami perubahan sesuai dengan fungsi *likelihood* yang telah didefinisikan pada subbab 2.3 dan proses perhitungannya adalah sebagai berikut

a) *Intensity Measurement*

Intensiy measurement dihitung berdasarkan *likelihood* seperti pada persamaan 2.4 berdasarkan kemiripan objek yang dilacak T dengan persekitaran partikel I pada *frame* t menggunakan *Sum of Square Differences* (SSD). Dengan mengambil potongan citra hasil super-resolusi pada contoh sebelumnya, misalkan kita punya dua buah matrik

$$template = \begin{bmatrix} 2 & 3 & 3 \\ 2 & 2 & 2 \\ 1 & 1 & 2 \end{bmatrix} \text{ dan } Image = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

Kedua matrik tersebut akan dihitung menggunakan SSD seperti pada persamaan 2.3 . Sebuah persekitaran kecil dari partikel atau $Neib$ berukuran $M-1 \times N-1$ seperti persamaan 2.3 dapat tuliskan kembali menjadi

$$r(X_t) = \sum_{i=0}^{M-1}\sum_{j=0}^{N-1}[T(i,j) - I(x+i, y+j)]^2$$

$$= \sum_{i=0}^{M-1}\sum_{j=0}^{N-1}[T(i,j) - I(x+i, y+j)][T(i,j) - I(x+i, y+j)]$$

$$= \sum_{i=0}^{M-1}\sum_{j=0}^{N-1}T(i,j)^2 - \quad 2\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}T(i,j)I(x+i, y+j)$$

$$+\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} I(x+i,y+j)^2$$

Untuk posisi partikel di $x=0; y=0 \Longrightarrow$

$$\sum_{i=0}^{2}\sum_{j=0}^{2} T(i,j)^2 = 2^2+3^2+3^2+\ldots.+1^2+2^2 = 40$$

$$\sum_{i=0}^{2}\sum_{j=0}^{2} T(i,j)I(x+i,y+j)$$

$$= (2\times 1)+(3\times 2)+(3\times 2)+\cdots+(1\times 1)+(2\times 1) = 28$$

$$\sum_{i=0}^{2}\sum_{j=0}^{2} I(x+i,y+j)^2 = 1+2^2+2^2+\ldots.+1^2+1^2 = 21$$

sehingga

$$r(x,y) = (40)-(2\times 28)+(21)$$
$$= 5$$

Dari perhitungan menggunakan SSD jika didapatkan hasil sebagai berikut

- Nilai kemiripan masing-masing partikel $r = \{184, 152,\ 192,\ 169, 130\}$
- Jumlah kandidat masing-masing partikel $J = \{1,3,2,1,1\}$
- Probabilitas prior $q_j = \frac{1-0.5}{\sum_{i=0}^{N-1} J_i} J_i$

  $= \left\{\frac{1}{16}, \frac{3}{16}, \frac{2}{16}, \frac{1}{16}, \frac{1}{16}\right\}$ dengan $q_0 = 0.5$
- Factor normalisasi

$$C_N = \frac{1}{\sum_{i=0}^{N-1} r_i} = \left[\frac{1}{184+152+192+169+130}\right]$$
$$= 0{,}00121$$

- Sebuah nilai terdistribusi *uniform* $U(.) = 0{,}5469$

maka dengan menggunakan fungsi *likelihood* pada persamaan 2.4

$$P\left(Z_t^{int}\middle|X_t\right) = q_0 U(.) + C_N \sum_{j=1}^{J} q_j N(r_t, \sigma_t)$$

$$= 0{,}5(0{,}5469)$$

$$+0{,}00121\left[\frac{1}{16}187{,}57+\frac{3}{16}154{,}76+\frac{2}{16}192{,}72+\frac{1}{16}168{,}9+\frac{1}{16}130{,}7\right]$$

$$= 0{,}5(0{,}5469) + 0{,}00121[83{,}5]$$

$$= 0{,}27345 + 0{,}10103$$

$$= 0{,}37448$$

didapatkan nilai *likelihood* untuk *intensity measurement* sebesar 0,37448.

b) *Motion Intensity*

Sebuah *likelihood* pada persamaan 2.7 digunakan untuk menghitung *motion intensity* berdasarkan selisih antara perubahan posisi partikel dengan kecepatan rata-rata objek pada waktu sebelumnya.

Motion intensity akan dihitung misalkan dengan kondisi pada *frame* ke-7 objek dapat dideteksi pergerakannya dan diketahui posisi objek pada *frame* sebelumnya adalah sebagai berikut

Tabel 4.1. Koordinat posisi objek

Frame	**Koordinat pada sumbu-x**	**Koordinat pada sumbu-y**
1	286	117
2	282	117
3	281	117
4	277	117
5	270	120
6	266	121

Proses perhitungan *motion intensity* pada *frame* ke-7 atau $t = 7$ dijalankan dengan mengetahui nilai $\overline{\Delta x}$ dan $\overline{\Delta y}$ dari perbedaan posisi partikel lama dengan partikel baru seperti pada persamaan 2.5. Apabila perbedaan posisi partikel lama dan partikel baru diketahui sebagai ($\Delta x_t = -6\,; \Delta y_t = -2$), $\sigma_{motion} = 1$ dan banyaknya koordinat posisi objek pada *frame* sebelumya yang dibutuhkan , misal $k = 4$ maka

$Untuk\ (t = 7), (\mathrm{k} = 4)$ didapatkan

$$\overline{\Delta x} = \sum_{s=t-k}^{t-1} \frac{|x_s - x_{s-1}|}{k}$$

$$= \sum_{s=2}^{6} \frac{|x_s - x_{s-1}|}{k}$$

$$= \frac{|x_2 - x_1| + |x_3 - x_2| + |x_4 - x_3| + |x_5 - x_4|}{4}$$

$$+ \frac{|x_6 - x_5|}{4}$$

$$= \frac{|282 - 286| + |281 - 282| + |277 - 281|}{4}$$

$$\frac{+|270 - 277| + |266 - 270|}{4}$$

$$= \frac{4 + 1 + 4 + 7 + 4}{4}$$

$$= \frac{20}{4} \approx 5$$

dan

$$\overline{\Delta y} = \sum_{s=t-k}^{t-1} \frac{|y_s - y_{s-1}|}{k}$$

$$= \sum_{s=2}^{6} \frac{|y_s - y_{s-1}|}{k}$$

$$= \frac{|y_2 - y_1| + |y_3 - y_2| + |y_4 - y_3| + |y_5 - y_4|}{4}$$

$$+ \frac{|y_6 - y_5|}{4}$$

$$= \frac{|117 - 117| + |117 - 117| + |177 - 117|}{4}$$

$$\frac{+|120 - 117| + |121 - 120|}{4}$$

$$= \frac{0 + 0 + 0 + 3 + 1}{4}$$

$$= \frac{4}{4} \approx 1$$

Kemudian menghitung perubahan posisi partikel dengan kecepatan rata-rata objek pada waktu sebelumnya berdasarkan persamaan 2.6

$$\begin{aligned} d_{mot}^2 &= (|\Delta x_t| - \overline{\Delta x})^2 + (|\Delta y_t| - \overline{\Delta y})^2 \\ &= (|6| - |5|)^2 + (|2| - |1|)^2 \\ &= 1^2 + 1^2 \approx 2 \end{aligned}$$

Kemudian hasil dari perhitungan diatas akan digunakan untuk perhitungan selanjutnya dengan menggunakan fungsi *likelihood* pada persamaan 2.7

$$\begin{aligned} P(Z_t^{mot}|X_t) &= \frac{1}{\sqrt{2\pi}\sigma_{mot}} \exp\left(-\frac{d_{mot}^2}{2\sigma_{mot}^2}\right) \\ &= \frac{1}{\sqrt{2\pi}.1} \exp(-\frac{2}{2.1^2}) \\ &= 0{,}1468 \end{aligned}$$

sehingga didapatkan nilai *likelihood* untuk *intensity measurement* sebesar 0,1468.

c) *Trajectory intensity*

Trajectory intensity dihitung berdasarkan kedekatan partikel terhadap lintasan yang diperoleh dari posisi objek pada *frame* sebelumnya seperti pada Tabel 4.1. Dari tabel tersebut akan dibuat suatu bentuk interpolasi dengan pendekatan polinomial berderajat 2 untuk mengetahui koefisien-koefisiennya

$$y = \sum_{i=0}^{m} a_i x^i = a_0 x^0 + a_1 x + a_2 x^2$$

Koefisien tersebut dapat dicari dengan beberapa metode dalam metode linear sehingga ditemukan nilai koefisiennya dengan nilai $a_2 = 0$; $a_1 = 8{,}3 \times 10^{-6}$; $a_0 = 1{,}2947 \times 10^{-3}$. Sebuah partikel pada posisi $x = 281$; $y = 117$ akan digunakan untuk mencari nilai *closeness metric* berdasarkan koefisien yang telah didapatkan pada proses diatas

$$\begin{aligned} d_{trj} &= |y - \sum_{i=0}^{m} a_i x^i \,| \\ &= |y - (a_0 x^0 + a_1 x + a_2 x^2)| \\ &= \,|[117] - (1{,}2947 \times 10^{-3} + (8{,}3 \times 10^{-6}) \times 281) + 117^2 \times 0| \end{aligned}$$

$= 0.003627$

Dua *frame* sebelumnya dimisalkan tidak terdeteksi gerakannya dengan $(\lambda_f = 0{,}9)$ dan $\sigma_{trj}^2 = 1$

$$P\left(Z_t^{trj} \middle| X_t\right) = \frac{1}{\sqrt{2\pi}\sigma_{trj}} \exp\left(-\frac{\frac{d_{trj}}{F}}{2\sigma_{trj}^2}\right)^2$$

$$= \frac{1}{\sqrt{2\pi} \times 1} \exp\left(-\frac{\frac{0{,}003627}{0.9^2}}{2 \times 1^2}\right)^2$$

$$= 0.00159$$

sehingga didapatkan nilai *likelihood* untuk *trajectory measurement* sebesar 0,00159.

Setelah Bobot yang telah diperbarui akan dipilih bobot terbaik pada proses *resampling* apabila nilai *effective sample size* (gagasan yang ditetapkan untuk sampel dari distribusi ketika pengamatan atau observasi dalam sampel berkorelasi atau berbobot) kurang dari setengah jumlah partikel. Kemudian posisi terbaru dari objek dapat diperoleh berdasarkan rata-rata partikel bobot masing-masing. Namun, apabila estimasi pergerakan tidak terdeteksi, kita akan mengestimasi posisi objek pada *frame* berikutnya dan memperhalus lintasan dengan cara memproyeksikan salah satu prediksi posisi benda ke estimasi lintasan.

Sebuah proses estimasi posisi objek pada *frame* ke-7 akan dilakukan misal dengan kondisi objek tidak terdeteksi pergerakannya pada *frame* ke-5. Jika diketahui informasi posisi objek pada *frame* sebelumnya seperti pada Tabel 4.3 dengan $i > j$ maka

$$\hat{X}_{cur} = X_i + \left(X_i - X_j\right) * \frac{cur - i}{i - j}$$

$$= X_6 + (X_6 - X_4)\left(\frac{7-6}{6-4}\right)$$

$$= \begin{bmatrix}264\\125\end{bmatrix} + \left(\begin{bmatrix}264\\125\end{bmatrix} - \begin{bmatrix}265\\121\end{bmatrix}\right)\frac{1}{2}$$

$$= \begin{bmatrix}264\\125\end{bmatrix} + \begin{bmatrix}-1/2\\2\end{bmatrix}$$

$$= \begin{bmatrix} 264{,}5 \\ 127 \end{bmatrix} \approx \begin{bmatrix} 265 \\ 127 \end{bmatrix}$$

Kemudian memproyeksikan $\hat{X}_{cur}$ ke dalam lintasan terdekatnya yaitu $\hat{X}_{cur}$ dengan menghitung jarak Euclidean dari beberapa posisi objek pada *frame* sebelumnya. Proses estimasi dilanjutkan dengan

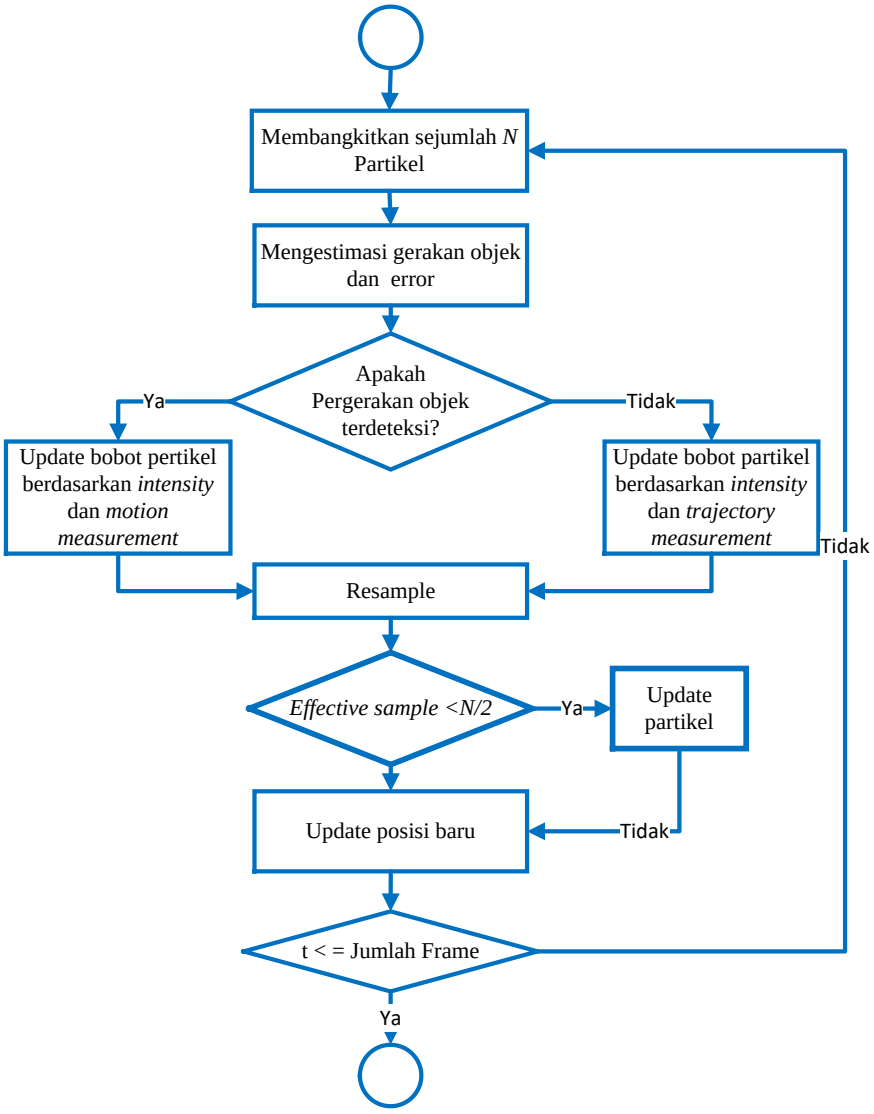
$$X_{cur} = \left(1 - \lambda_f^{t\_o}\right)\hat{X} + \tilde{X} * \lambda_f^{t\_o}$$

$$= (1 - 0{,}9^2)\begin{bmatrix} 265 \\ 127 \end{bmatrix} + \begin{bmatrix} 264 \\ 125 \end{bmatrix}(0.9)^2$$

$$= \begin{bmatrix} 50{,}35 \\ 24{,}13 \end{bmatrix} + \begin{bmatrix} 213{,}84 \\ 101{,}25 \end{bmatrix}$$

$$= \begin{bmatrix} 264{,}19 \\ 125{,}38 \end{bmatrix} \approx \begin{bmatrix} 264 \\ 125 \end{bmatrix}$$

Sehingga diperoleh posisi objek pada *frame* ke-7 dengan koordinat $x = 264; y = 125$. Proses dilanjutkan pada *frame* berikutnya, apabila sudah pada *frame* terakhir proses akan berhenti. Diagram alir dari pelacakan objek dengan metode *Adaptive Particle Filter* ditunjukkan pada Gambar 4.4.



Gambar 4.4 Diagram alir dari pelacakan dengan *Adaptive Particle Filter*

4.1.3 Estimasi Pergerakan Objek

Dalam proses pelacakan objek bergerak menggunakan *Adaptive Partikel Filter*, proses estimasi gerakan objek digunakan untuk mengetahui vektor gerakan objek. Proses estimasi gerakan menggunakan metode *Hierarchical Block Matching*. Input dalam proses estimasi berupa *frame* video dimulai pada *frame* ke-t dan *frame* $t-1$ sehingga pada saat *frame* pertama tidak dilakukan proses estimasi gerakan karena pada *frame* pertama sudah diketahui posisi objek yang dilacak melalui ROI. Selanjutnya dilakukan proses reduksi *frame* ukuran dan resolusi membentuk

beberapa level yaitu level 0, level 1 dan level 2 (selengkapnya dijelaskan pada subbab 2.4). Proses dilakukan mulai dari level 2 kemudian menentukan ukuran blok besar dan blok kecil berdasarkan posisi objek sehingga pencocokan blok tidak dilakukan pada keseluruhan *frame* namun hanya pada persekitaran objek. Pencocokan pada blok dilakukan menggunakan Mean Absolute Difference (MAD) dan persekitarannya.

Dua bagian dari *frame* yang berbeda direpresentasikan dalam dua matrik sebagai berikut

$$C = \begin{bmatrix} 245 & 246 & 248 & 245 \\ 249 & 243 & 248 & 249 \\ 240 & 248 & 247 & 247 \\ 196 & 231 & 234 & 230 \end{bmatrix} \text{ dan } R = \begin{bmatrix} 222 & 249 & 248 & 247 \\ 211 & 243 & 248 & 248 \\ 207 & 228 & 243 & 247 \\ 174 & 180 & 179 & 179 \end{bmatrix}$$

Kedua matrik tersebut akan dicari nilai kecocokan dengan MAD

$$MAD = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left|C_{ij} - R_{ij}\right|$$

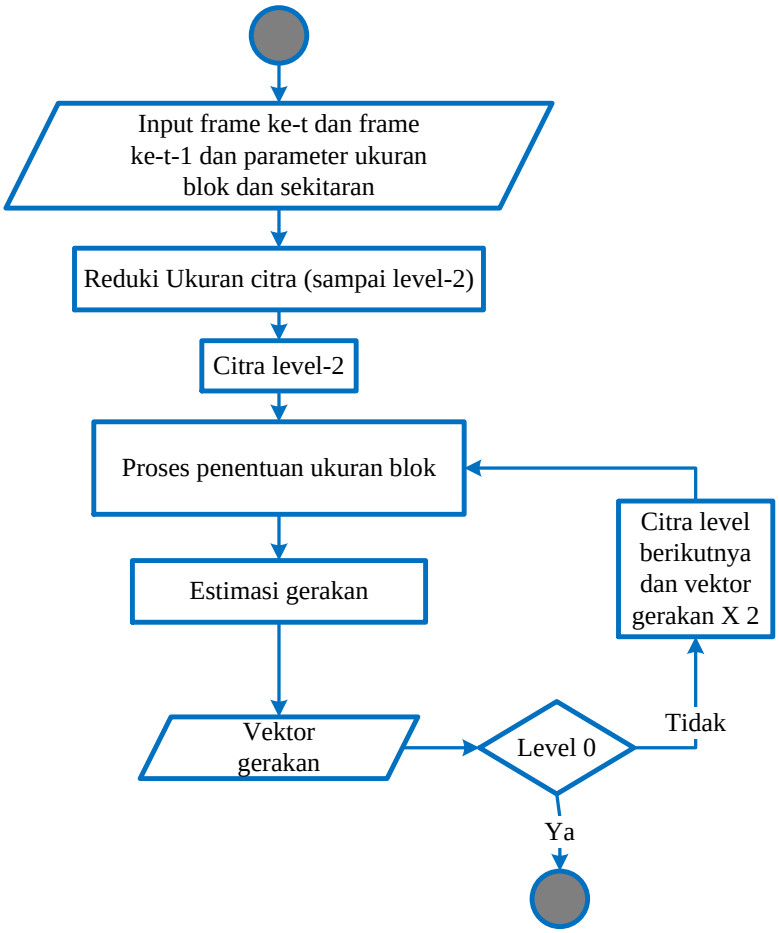
$$= \frac{1}{4^2}\begin{bmatrix} |C_{00} - R_{00}| + |C_{01} - R_{01}| + . & \cdots & +|C_{03} - R_{03}| + \\ \vdots & \ddots & \vdots \\ +|C_{30} - R_{30}| + |C_{31} - R_{31}| + & \cdots & +|C_{03} - R_{03}| \end{bmatrix}$$

$$= \frac{1}{4^2}\begin{bmatrix} |245 - 222| + |249 - 246| + . & \cdots & +|247 - 245| + \\ \vdots & \ddots & \vdots \\ +|174 - 196| + |180 - 231| + & \cdots & +|179 - 230| \end{bmatrix}$$

$$= \frac{1}{4^2}303$$

$$= 18{,}93 \approx 19$$

Semakin kecil nilai MAD maka dua bagian *frame* tersebut cenderung memiliki kesamaan dan akan dijadikan sebagai penentuan dalam mengambil nilai vektor gerakan. Proses tersebut akan terus berlanjut sampai pada level 0. Gambar 4.5 menunjukkan diagram alir proses estimasi gerakan objek dengan menggunakan metode *Hierarchical Block Matching*.



Gambar 4.5. Diagram alir estimasi gerak objek menggunakan Hierarchical *Block Matching*

4.2 Perancangan Perangkat Lunak

Untuk melakukan pengujian, metode dalam penelitian ini yang telah dijelaskan pada subbab 3.3 akan dibangun dalam sebuah bentuk perangkat lunak yang disertai proses analisis kerja dan informasi tentang perangkat yang digunakan**.** Dalam hal ini, perancangan perangkat lunak merupakan teknis yang diperlukan supaya dalam proses implementasi akan berjalan secara sistematis dan efisien. Beberapa informasi tambahan dan kebutuhan implementasinya adalah sebagai berikut.

4.2.1 Proses Kerja Perangkat Lunak

Dalam penerapannya, program pelacakan objek kecil menggunakan metode *Adaptive Particle Filter* dan teknik super-resolusi dengan menggunakan

Phased Based Imange Matching untuk registrasi dan *Projection Onto Convex Sets* untuk rekonstruksi. Berikut ini adalah proses kerja pelacakan objek kecil berbasis citra super-resolusi pada perangkat lunak yang dibangun :

1. Input video pada perangkat lunak yang telah dibangun
2. Mengubah video dari video yang diinputkan menjadi serangkaian *frame* video
3. Ekualisasi histogram pada masing-masing *frame* video untuk mendapatkan histogram yang merata. Dalam proses ini bisa disebut sebagai proses pra-pengolahan.
4. Mengambil serangkaian citra masukan dari beberapa rangkaian *frame* video untuk mengetahui nilai translasinya dalam proses registrasi.
5. Rekonstruksi citra berdasarkan nilai translasi yang telah diperoleh dalam proses registrasi citra
6. Memilih objek yang akan diamati secara manual oleh *user*.
7. Estimasi gerakan objek dengan algoritma *Hierarchical Block Matching* untuk mengetahui vektor gerakan yang akan digunakan untuk memindahkan partikel dalam metode *Adaptive Particle Filter*.
8. Menerapkan metode *Adaptive Particle Filter* untuk pelacakan objek kecil.

4.2.2 Kebutuhan Perancangan

Hal yang diperlukan untuk mengimplementasikan metode diatas adalah perangkat keras dan perangkat lunak dengan spesifikasinya yang disajikan dalam Tabel 4.2.

Tabel 4.2. Kebutuhan Perancangan

Perangkat keras	1. Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz with NVIDIA 930M Graphics 2.0GHz 2. Memory 4096MB RAM
Perangkat lunak	1. Sistem operasi Microsoft Windows 10 Pro 64-bit (10.0, Build 10586)

	2. Tools menggunakan MATLAB R2015a termasuk diantaranya desain antar muka dan *toolbox* dalam pengolahannya.

4.2.3 Perancangan Proses Akuisisi Data

Tahap perancangan proses akuisisi data dalam perangkat lunak membutuhkan tiga tahapan proses akuisisi data yang akan digunakan untuk kesesuaian pada perangkat lunak yang akan diuji. Ketiganya meliputi proses akuisisi data masukan, data proses, dan data keluaran. Data masukan dapat dikategorikan sebagai input dari sebuah program perangkat lunak berupa video objek bergerak. Data proses merupakan data yang berisi parameter-parameter yang dibutuhkan dalam metode *Adaptive Particle Filter* yang akan mempengaruhi nilai pergerakan sebuah benda bergerak dalam video. Sedangkan data keluaran adalah data masukan yang telah diproses dalam beberapa tahap yang menghasilkan sebuah video pelacakan objek bergerak.

a) Data Masukan

Data masukan dalam tahap ini berupa video pergerakan objek kecil dengan karakteristik yang berbeda-beda seperti video olah raga, video surveilance dan beberapa video pergerakan objek kecil lainnya. File video yang diambil memiliki spesifikasi sebagai berikut

a. Video diambil dengan kecepatan minimal 25 *frame per second*.
b. Video memiliki resolusi minimal $360x640$ piksel.
c. Video memiliki ekstensi *.avi

b) Data Proses

Dalam data proses, input dari data masukan akan digunakan dalam proses pengolahan. Data proses ini diperoleh dari hasil pengolahan data yang sesuai dengan beberapa tahapan algoritma atau metode yang telah disusun. Tabel 4.2 menjelaskan tahapan dari data proses.

Tabel 4.3. Tabel data proses

No	Tahapan	*Input*	*Output*
1.	*Input* video	Video	*Frame* Citra yang disertakan proses *cropping*
2	Konversi video ke *frame*	Video	*Frame*
3	Pra-pengolahan	*Frame* dari proses konversi video ke *frame*	*Frame* Citra dengan histogram yang lebih merata
4	Super-resolusi	*Frame* hasil ekuilisasi histogram	*Frame* citra super-resolusi
5	Pilih Area ROI Objek	*Frame* pertama dari video	Citra ROI dan sebuah titik pusat objek (*centroid*)
6	Proses Hierarchical *Block Matching*	*Frame* ke-t dan frame ke-$t-1$	Vektor gerakan (*motion vector*) objek
7	Proses Pelacakan Objek dengan metode *Adaptive Particle Filter*	*Frame* Citra ke- $t-1$ dan *frame-*t serta vektor gerakan objek	Posisi baru dari objek yang dilacak berdasarkan titik pusat objek (*centroid*) yang diperoleh.

c) Data Keluaran

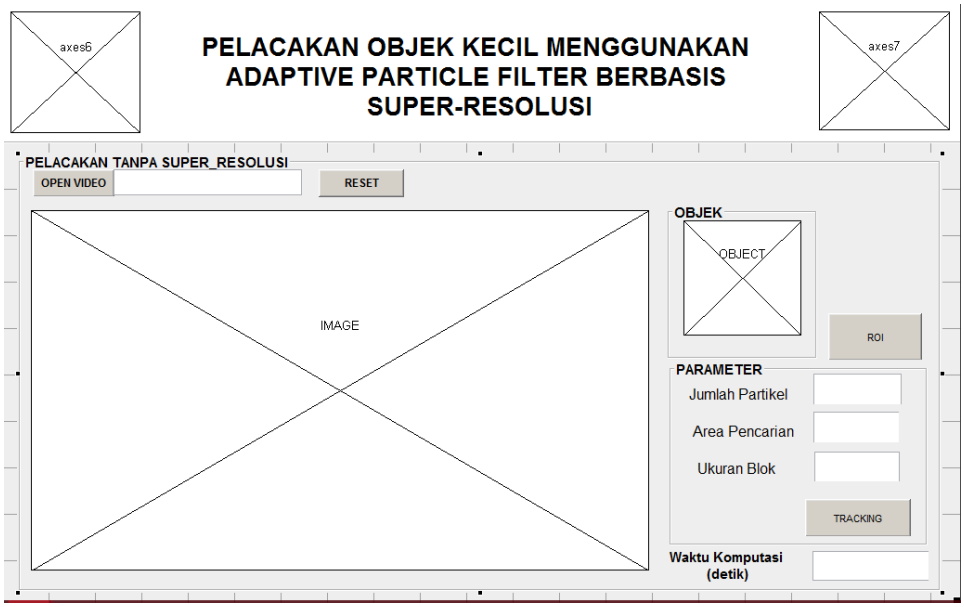
Data keluaran menghasilkan sebuah output berupa *frame* dan video hasil pelacakan objek yang disertakan penandaan posisi objek yang bergerak (*bounding box*) di setiap *frame* video.

4.2.4 Perancangan Antar Muka

Halaman utama dalam program pada perangkat lunak yang dibangun ditunjukkan pada Gambar 4.6. Pada halaman ini, seorang pengguna dapat melakukan beberapa proses yang meliputi proses pelacakan objek tanpa super-resolusi, pelacakan berbasis super-resolusi dan pengolahan video (konversi video ke *frame* dan proses super-resolusi).

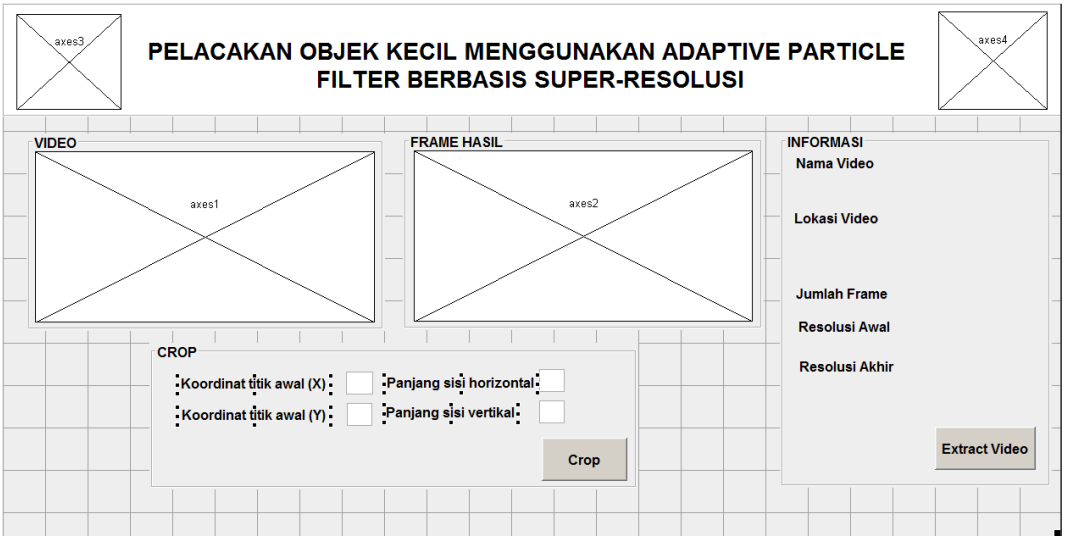
Gambar 4.6. Antar muka halaman utama

Desain antar muka dibutuhkan sebagai media interaksi antara pengguna dengan perangkat lunak. Desain ini dibuat semenarik mungkin agar pengguna mudah mengoperasikan perangkat lunak yang dibangun. Secara umum, perancangan antar muka dari halaman pelacakan dapat ditunjukkan seperti pada Gambar 4.7

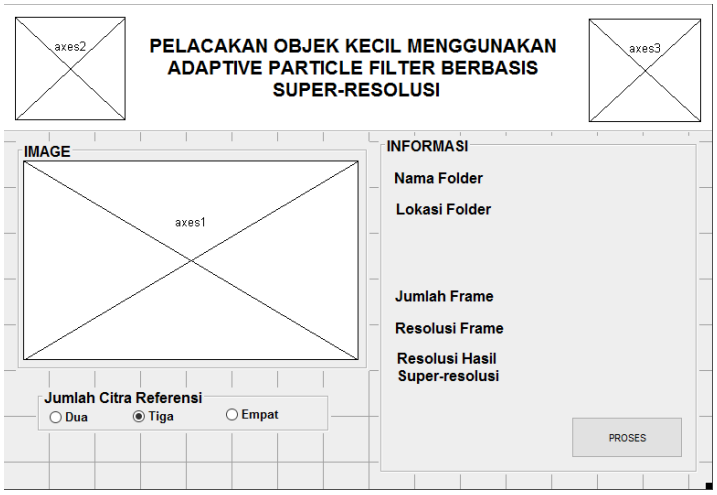


Gambar 4.7. Perancangan halaman pelacakan

Selanjutnya, perancangan antar muka untuk konversi video ke *frame* yang disertai *cropping* ditunjukkan dalam Gambar 4.8 dan perancangan untuk proses super-resolusi Gambar 4.9



Gambar 4.8. Perancangan antar muka untuk proses konversi video ke *frame*



Gambar 4.9. Perancangan antar muka proses konversi video ke *frame*

4.3 Implementasi Pelacakan Objek berbasis Super-resolusi

Terdapat beberapa tahapan dalam mengimplementasikan pelacakan objek kecil menggunakan *Adaptive Particle Filter* berbasis super-resolusi diantaranya adalah proses konversi video ke *frame*, input data, pra-pengolahan, super-resolusi, pemilihan objek dengan ROI, pelacakan objek kecil dengan menggunakan *Adaptive Particle Filter* berbasis super-resolusi dengan estimasi gerakan objek menggunakan *Hierarchical Block Matching*.

4.3.1 Implementasi Proses Konversi Video ke *Frame*

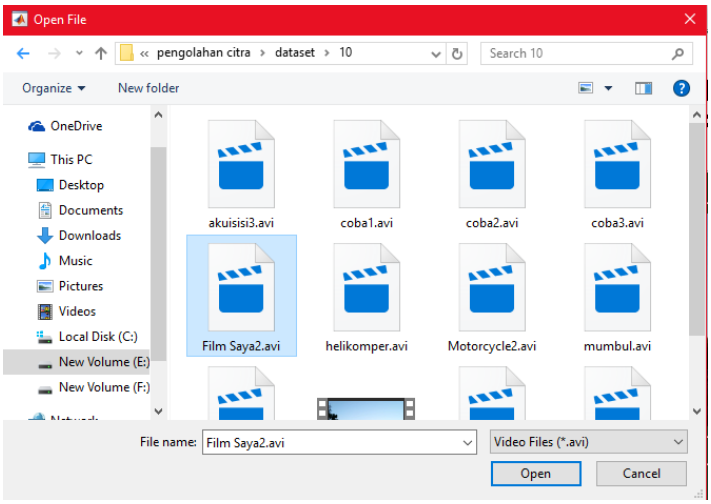
Proses ini akan menentukan video yang nantinya digunakan sebagai data masukan untuk dikonversi menjadi gambar atau *frame*. Sebelum melangkah jauh dalam proses ini, terlebih dahulu buat folder dalam partisi hardisk **(E:\)** dalam komputer dengan nama ‘Extract’. Penjabaran tentang proses ini adalah sebagai berikut

Fungsi	: Konversi video (berektensi .avi) ke frame yang disertakan proses *cropping*.
Jenis Input	: Video (.avi) dan nilai yang telah diberikan
Output	: Frame (.jpg)
Deskripsi	: Konversi video yang dipilih (berekstensi .avi) ke frame disertai proses *cropping*. Kemudian hasilnya akan disimpan dalam sebuah folder tertentu untuk proses selanjutnya.

Kode program berserta langkah-langkah untuk konversi video ke *frame* adalah sebagai berikut.

Proses dimulai dengan memilih video berektensi *.avi yang telah tersimpan dalam komputer dengan klik menu **File** kemudian pilih **Open Video.** Gambar 4.10 adalah penampilan antar muka pengambilan video.

```
[filename, pathname] = uigetfile( ...
{'*.avi','Video Files (*.avi)'},...
'Open File','../../../TestVideos/');
if ~isequal(filename,0)
fullname = fullfile(pathname, filename);
vid=fullname;
else
return
end
```



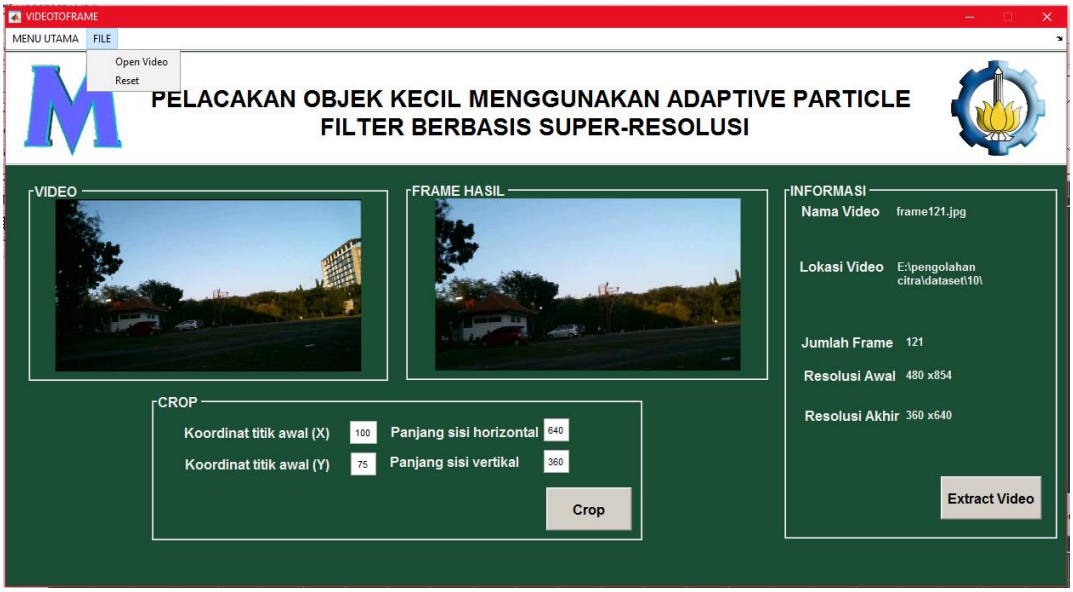
Gambar 4.10. Antar muka *input* video

Tahapan berikutnya mengambil nilai yang diinputkan yaitu kooordinat awal (x dan y) dari video tersebut beserta perpanjangannya dari sisi vertikal dan perpanjangannya dari sisi horizontal.

```
video = vision.VideoFileReader(vid);
roi1= get(handles.roi1,'String');
get_roi(1) = str2num (roi1)-1;
roi2 = get(handles.roi2,'String');
get_roi(2) = str2num (roi2)-1;
roi3 = get(handles.roi3,'String');
get_roi(3) = str2num (roi3)-1;
roi4 = get(handles.roi4,'String');
get_roi(4) = str2num (roi4)-1;
```

Selanjutnya, video akan di konversi menjadi beberapa *frame* yang disertai proses *cropping* dari nilai yang diinputkan tersebut. Proses *cropping* bertujuan untuk memotong ukuran video yang fokus pada daerah yang dilalui oleh objek yang bergerak sehingga objek yang akan dilacak menjadi lebih tampak dan waktu proses pelacakan jadi lebih cepat. Gambar 4.11 menunjukkan antar muka secara keseluruhan pada proses konversi video ke *frame*.

```
img=1;
while ~isDone(video)
frame=step(video);
frame2=(imcrop(frame,get_roi));
filename=strcat('frame',num2str(img),'.jpg');
workingDir = 'E:\Extract';
fullname= fullfile(workingDir,filename);
imwrite(frame2,fullname);
img=img+1;
end
```



Gambar 4.11. Antar muka proses konversi video ke *frame*

Hasil dari proses ini akan disimpan dalam sebuah folder yang bernama ‘Extract’ yang telah dibuat sebelum proses ini dijalankan.

4.3.2 Proses Pra-pengolahan Video

Pra-pengolahan video dilakukan untuk mengatasi masalah dari pengaruh pencahayaan. Penjabaran tentang proses pemilihan video adalah sebagai berikut**.**

Fungsi	: Pemerataan histogram pada semua frame hasil konversi dari video (.avi) yang disertai proses *cropping*.
Jenis Input	: Folder (kumpulan frame)
Output	: Frame baru dengan histogram yang merata.
Deskripsi	: Pemerataan histogram (ekualisasi histogram) dilakukan pada semua frame video sebelum melakukan proses super-resolusi.

Kode program untuk memenuhi kondisi seperti diatas adalah sebagai berikut

```
for i=1:nframe
    Frame=rgb2gray(frame{i});
    img {num} = histeq(Frame);
    num = num+1;
end
```

4.3.3 Super-resolusi

Super-resolusi dalam tahap ini adalah proses untuk memperoleh citra super-resolusi yang lebih tinggi. Citra super-resolusi diperoleh dari serangkaian citra resolusi rendah dalam hal ini diambil dari *frame* video yang telah melalui proses pra-pengolahan. Serangkaian *frame* tersebut akan diambil sebagai masukan atau citra observasi dalam proses registrasi. Proses ini dijalankan sebelum merekonstruksi super-resolusi untuk mendapatkan nilai parameter translasi. Nilai tersebut akan digunakan untuk proses rekonstruksi citra super-resolusi. Gambar 4.12 adalah tampilan antar muka untuk proses super-resolusi. Penjabaran tentang proses pemilihan video adalah sebagai berikut**.**

Fungsi	:Peningkatan resolusi frame video (berekstensi .jpg)
Input	: Frame video (.jpg)
Output	: Frame video dengan resolusi lebih tinggi (.jpg)
Deskripsi	: Serangkaian citra digunakan sebagai citra masukan dalam proses registrasi menggunakan PBIM kemudian hasilnya akan digunakan dalam proses rekonstruksi menggunakan POCS.

Gambar 4.12. Antar muka proses super-resolusi

Proses registrasi mengambil beberapa buah *frame* citra referensi, misalnya mengambi tiga buah *frame* sebagai citra observasi dengan kriteria dua *frame* berikutnya dari citra yang akan ditingkatkan resolusinya. Sedangkan pada bagian akhir dari *frame* video ($lastframe$-1 atau $lastframe$-2) akan mengambil dua buah *frame* sebelumnya dari cita yang akan diproses super-resolusi. Kode program untuk proses super-resolusi ini adalah sebagai berikut

```
img=frame   % frame awal setelah proses pra-pengolahan
sequence=1:nframe;
k=nframe;
factor=2;
for i=1:nframe
    loop=1;
     if sequence(i)<k-1
         for kk=sequence(i):sequence(i)+2
              image{loop}=rgb2gray(img{kk});
              s{loop}=rgb2gray(img{kk});
              loop=loop+1;
         end
     end
     if sequence(i)==k-1&& sequence(i)==k
         for ll=sequence(i):-1:sequence(i)-2
              image{loop}=rgb2gray(img{ll});
              s{loop}=rgb2gray(img{ll});
              loop=loop+1;
         end
    end
    [delta_est, phi_est] = PBIM(image);
    imageSR{i}= (pocs(s,delta_est,factor));
end
```

Hasil dari proses super-resolusi ini akan disimpan dalam sebuah folder bernama ‘Extract’ dengan kode program sebagai berikut

```
for i=1:nFrame
  filename=strcat('Citra Super Resolusi',num2str(i),'.jpg');
  workingDir = 'E:\Extract';
  fullname= fullfile(workingDir,filename);
  imwrite(imageSR{i},fullname);
end
```

4.3.4 Proses Pengambilan Input Data *Frame* Video dan parameter

Pada proses ini akan dipilih data *frame* video dan parameter yang nantinya digunakan sebagai data masukan untuk diproses pelacakan objek bergerak. Gambar 4.13 adalah tampilan antar muka pelacakan objek kecil berbasis super-resolusi. Penjabaran tentang proses pemilihan dan input video adalah sebagai berikut

Fungsi	: Input data frame video dalam sebuah folder.
Input	: Frame (.jpg)
Output	: Frame (.jpg)
Deskrips	:Data frame video didapatkan dari proses sebelumnya meliputi hasil konversi video ke frame dan hasil super-resolusinya dimana keduanya telah tersimpan menjadi satu dalam satu folder dalam komputer.

Gambar 4.13. Antar muka pelacakan objek kecil berbasis super-resolusi yang dilanjutkan proses *input* data *frame* video.

Fungsi	: Input folder yang berisikan keseluruhan frame dari sebuah video beserta hasil super-resolusinya.
Jenis Input	: Folder (kumpulan frame)
Deskripsi	: Mengambil frame yang disimpan dalam sebuah folder

Kode program untuk memenuhi kondisi seperti diatas adalah sebagai berikut

```
imgDir = uigetdir;
fileNames = dir(imgDir);
files=dir(sprintf('%s/*.jpg',imgDir));

nframe = length(files)/2;
for in=1:nframe
   filename = sprintf('frame%1.1i.jpg', in);
   R = imread([imgDir '\' filename]);
   image{in}=(R);
end
for in=1:nframe
   filename = sprintf('Citra Super Resolusi%1.1i.jpg', in);
   R = imread([imgDir '\' filename]);
   imageSR{in}=im2double(R);
end
```

Setelah input data video dilanjutnya pengisian parameter yang meliputi

a. **Ukuran Blok** untuk menentukan ukuran blok untuk melakukan estimasi gerakan.
b. **Area Pencarian** untuk menentukan search area pada estimasi gerakan.
c. **Jumlah Partikel** untuk menentukan jumlah partikel yang dibangkitkan dalam pelacakan objek dengan metode *Adaptive Particle Filter*.

4.3.5 Proses Pemilihan Objek dengan ROI

Region of Interest (ROI) adalah bagian dari citra atau *frame* yang akan diproses dalam pelacakan objek bergerak. Pembentukan ROI diperoleh dengan membentuk *rectangle* yang mengelilingi objek. Ukuran ROI disesuaikan dengan ukuran objek yang akan dibutuhkan.

Fungsi	: Pemilihan objek disertai posisi awal objek yang akan dilacak pergerakannya pada setiap frame.
Input	: Frame pertama dari video.
Output	: Citra ROI dan *centroid* dari objek ROI.
Deskripsi	: Mendapatkan bagian citra atau objek yang akan dilacak

Proses tersebut diimplementasikan dengan kode program sebagai berikut

```
h=imrect;
posisi=wait(h);
get_roi=getPosition(h);
target_img=imcrop(img,get_roi);
```

Gambar 4.14. Bagian *frame* yang dipilih sebagai ROI

4.3.6 Pelacakan Objek Menggunakan *Adaptive Particel Filter*

Bagian ini akan memproses pelacakan objek kecil dengan metode *Adaptive Particle Filter* dari data *frame* video yang telah di-*input*kan. Penjelasan singkat proses metode *Adaptive Particle Filter* adalah sebagai berikut

Fungsi	: Mengolah citra untuk melacak objek yang bergerak.
Input	: Frame (.jpg) dan parameter yang meliputi jumlah partikel, ukuran blok dan area pencarian
Output	: Posisi baru dari objek yang dilacak dalam bentuk video (.avi)
Deskripsi	: Memproses setiap frame video untuk mengetahui posisi objek dalam setiap frame.

Penjelasan mengenai setiap tahapan dalam proses pelacakan objek kecil menggunakan metode *Adaptive Particle Filter* adalah sebagai berikut :

a) Membangkitkan sejumlah N partikel

Pada tahap awal atau pada saat memproses *frame* ke-1, partikel $X_t^{(i)}$disebarkan secara acak berdasarkan posisi centroid objek yang dikuetahui. Namun, informasi mengenai bobot partikel belum diketahui sehingga perlu menggunakan prinsip *noninformative prior* yang telah dijelaskan diatas yaitu bobot awal dinyatakan mempunyai distribusi uniform (mempunyai probabilitas yang sama)

$$\pi_t^{(i)} = 1/n$$

dimana *N* adalah parameter untuk jumlah partikel yang akan dibangkitkan pada sekitaran *centroid* objek. Kode program untuk membangkitkan sejumlah N partikel adalah sebagai berikut :

```
if t==1
X=bsxfun(@plus,trackedLocation(t,:)', randn(2,N));
else
X=bsxfun(@plus,trackedLocation(t-1,:)', randn(2,N));
end
```

b) Estimasi gerakan dari objek dan prediksi error.

Vektor gerakan objek diperoleh dengan menggunakan metode *Hierarchical Block Matching* dengan algoritma pencocokan Exhaustive Search. Penjabaran tentang estimasi gerakan adalah sebagai berikut

Fungsi	: mendapatkan vektor gerakan objek.
Input	: frame t dan $t-1$.
Output	: vektor gerakan (*motion vector*) objek.
Deskripsi	: estimasi gerakan berbasis blok dan juga memperhitungkan sekitaran objek dimana keduanya ukurannya telah ditentukan pada saat pengisian parameter. Pemilihan gerakan objek berdasarkan nilai vektor gerakan (*motion vector*) yang terbesar.

Penjelasan mengenai estimasi gerak dan proses perhitungannya terdapat pada subbab 4.1.3. Pencarian vektor gerakan dapat diimplementasikan dalam kode program dan selengkapnya dapat dilihat dalam Lampiran B.

Suatu citra digital pada video saat *frame* ke-t dapat dinyatakan sebagai $f(x,y,t)$, dimana *x* dan *y* merupakan titik koordinat spasial dan t adalah urutan *frame*. Jika turunan parsial dari $f(x,y,t)$ terhadap x,y,t dinyatakan sebagai

$$I_x = \frac{\delta f(x,y,t)}{\delta x} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x,y,t) - f(x,y,t)}{\Delta x}$$

$$I_y = \frac{\delta f(x,y,t)}{\delta y} = \lim_{\Delta y \to 0} \frac{f(x,y+\Delta y,t) - f(x,y,t)}{\Delta y}$$

$$I_t = \frac{\delta f(x,y,t)}{\delta t} = \lim_{\Delta t \to 0} \frac{f(x,y,t+\Delta t) - f(x,y,t)}{\Delta t}$$

maka untuk kondisi diskrit $\Delta t \to 0$ dapat bernilai 1 piksel sehingga persamaan diatas dapat ditulis kembali menjadi

$$I_x = \frac{\delta f(x,y,t)}{\delta x} = f(x+1,y,t) - f(x,y,t)$$

$$I_y = \frac{\delta f(x,y,t)}{\delta y} = f(x,y+1,t) - f(x,y,t)$$

$$I_t = \frac{\delta f(x,y,t)}{\delta t} = f(x,y,t+1) - f(x,y,t)$$

Sebuah prediksi error dari estimasi gerakan dinyatakan sebagai

$$\mu_t = \left| uI_x + vI_y + I_t \right|$$

dimana u merupakan estimasi gerakan terhadap horizontal (sumbu-x) dan v untuk arah vertikal (sumbu-y) Huang [4]. Kode program untuk menghitung prediksi error adalah sebagai berikut:

```
pixel = round(trackedLocation(t-1,:));
Ix = prevFrame(pixel(2),pixel(1)+1)-
prevFrame(pixel(2),pixel(1));
Iy = prevFrame(pixel(2)+1,pixel(1))-
prevFrame(pixel(2),pixel(1));
It = frame2(pixel(2),pixel(1)) -
prevFrame(pixel(2),pixel(1));
miu = abs((u*Ix)+(v*Iy)+It);
```

Setelah prediksi error diperoleh, ganti partikel $\left\{\left(X_{t-1}^{(i)}, \pi_{t-1}^{(i)}\right) \middle| i = 1, \ldots, N\right\}$ dengan $X_t^{(i)} \sim N(X_{t-1}^{(i)} + V_t, \mu_t)$ atau seperti pada persamaan 2.1 dimana V_t adalah vektor gerakan objek yang meliputi nilai u dan v. Namun, Jika $V_t = 0$ ($u = 0$ dan $v = 0$) maka $\mu_t = \mu_{maks}$. Kode program selengkapnya dapat dilihat pada Lampiran A.

c) *Update* bobot partikel

Seperti pada penjelasan diatas, partikel telah di*update* dengan partikel baru sehingga bobot yang lama perlu diperbarui. Perubahan bobot partikel dilakukan dengan mempertimbangkan *intensity measurement*, *motion measurement* dan *trajectory measurement*. *Intensity measurement* diasumsikan *independent* dari *motion measurement* atau *trajectory measurement*. Apabila objek terdeteksi gerakannya maka bobot partikel dihitung berdasarkan *intensity measurement* dan *motion measurement*. Apabila objek tidak terdeteksi gerakannya maka bobot partikel dihitung berdasarkan *intensity measurement* dan *trajectory measurement*. Jadi, *motion measuremen* dan *trajectory measuremen* tidak akan diproses dalam satu waktu dan keadaan yang sama.

Intensity measurement dihitung berdasarkan kemiripan antara target dan kandidat partikel dengan menggunakan *SSD correlation surface*. Output dari *SSD correlation surface* berupa J yaitu banyaknya kandidat dari sekumpulan piksel yang sesuai dalam neib untuk setiap partikel, standar deviasi dan nilai SSD r untuk setiap partikel. Secara umum, kode program untuk *intensity measurement* adalah sebagai berikut

```
[ J , stdIT , r ] = SSD (frame2,template, Xt ,N , mbSize);
a = 0; b = 1;
q0 = 0.5;
C = 1/sum(r);
for i= 1:N
qj = (1-0.5)/J(i);
p = qj*(normrnd(r(i),stdIT(i),[1 J(i)]));
P1(i)= (q0*unifrnd(a,b))+(C*sum(p));
end
```

Kode program selengkapnya dari fungsi untuk menghitung SSD setiap partikel disajikan pada Lampiran C.

Motion measurement dihitung berdasarkan selisih antara perubahan posisi partikel dan rata-rata perubahan posisi objek. Kode program untuk pengukuran gerak sebagai berikut:

```
%average object speed
delta2 = 0;
for o=-30:-1
if(t+o<=0)
    Xs = [0 0];
else
     Xs = trackedLocation(t+o,:);
end
if(t+o-1<=0)
  Xs1 = [0 0];
else
  Xs1 = trackedLocation(t+o-1,:);
end
delta2_0 = Xs - Xs1;
delta2 = delta2 + delta2_0;
end
delta2 = delta2/30 ;
%particle speed
for i=1:N
delta1 =[ abs(X(1,i)-Xt(1,i)) abs(X(2,i)-Xt(2,i))];
d = ((delta1(1)- delta2(1))^2+ (delta1(2)-delta2(2))^2);
P2(i)= (1/sqrt(2*pi))*(exp(-1/2 *(d/50).^2));
end
```

Trajectory measurement dihitung berdasarkan kedekatan partikel dengan lintasan atau posisi objek yang diperoleh dari *frame* sebelumnya dan dinotasikan dalam fungsi polinomial dengan order 2. Pada pengukuran lintasan juga terdapat λ_f yaitu rasio *forgotten* dengan menggunakan $\lambda_f = 0{,}9$ berdasarkan paper Huang [4]. Sedangkan, banyaknya *frame* pada saat objek tidak terdeteksi dinotasikan t_o (dalam program dinotasikan dengan occ). Kode program untuk *trajectory measurement* adalah sebagai berikut:

```
occ = occ+1;
koef_poly = polyfit
(trackedLocation(:,1),trackedLocation(:,2),2);
for i=1:N
poly = [ Xt(2,i)^2 Xt(2,i) 1];
d_trj = abs (Xt(2,i)-(sum(koef_poly.*poly)));
P2(i)= (1/sqrt(2*pi))*(exp(-1/2 *((d_trj/(0.9^occ))/1).^2));
end
```

d) *Resampling*

Proses *resampling* bertujuan untuk memilih bobot partikel yang tinggi sebagai pertimbangan untuk menghasikan output berupa posisi objek yang dilacak. *Resampling* dilakukan jika nilai *Effective Sample Size* kurang dari setengah jumlah partikel dan menggantikan himpunan partikel beserta bobotnya sebelumnya $\left\{\left(X_t^{(i)}, \pi_t^{(i)}\right) \middle| i = 1, \dots, N\right\}$ menjadi $\left\{\left(\tilde{X}_t^{(i)}, \frac{1}{N}\right) \middle| i = 1, \dots, N\right\}$. Implementasi tersebut secara umum dapat dibuat dengan kode program sebagai berikut

```
if Neff < Nt
          disp('Resampling ...\n')
          [index] = resample(w);
           Xt = Xt(:,index);
           w=ones(1,N)./N;
end
```

Kode program selengkapnya proses *resampling* disajikan pada Lampiran D.

e) *Update* posisi objek

Posisi terbaru dari objek diperoleh dengan menghitung nilai rata-rata dari nilai seluruh partikel berdasarkan bobotnya. Kode program untuk update posisi objek adalah sebagai berikut:

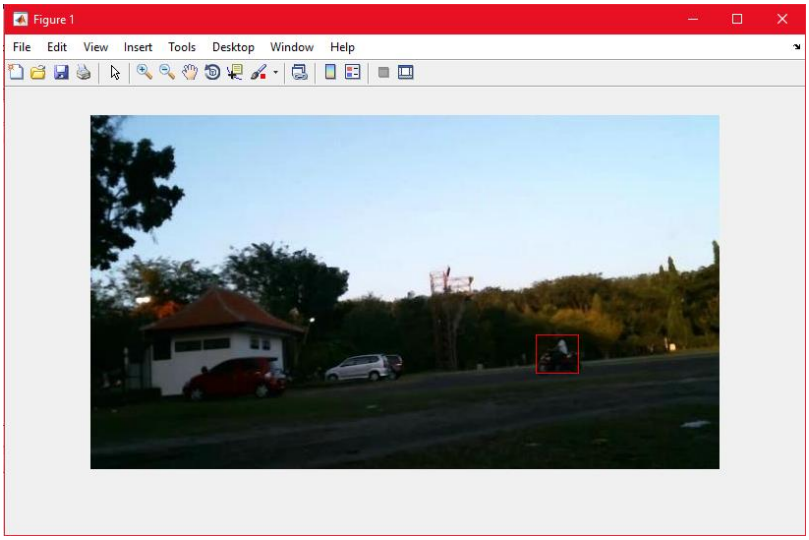
```
Detect (j,:) = (sum(bsxfun(@times,Xt,w),2)/sum(w))';
trackedLocation(t,:) = Detect(j,:);
```

Apabila pada *frame* $t-1$ (atau *frame cur* -1) tidak terdeteksi gerakan objek yang dilacak maka estimasi dilakukan dengan cara memilih satu atau lebih partikel dengan bobot maksimum dan hitung rata-rata partikel. Selanjutnya adalah memperhalus lintasan dengan memproyeksikan salah satu prediksi posisi

benda ke estimasi lintasan untuk mempertahankan kehalusan gerakannya. Posisi terbaru hasil penghalusan dapat ditentukan dengan $X_{cur} = \left(1 - \lambda_f^{t\_o}\right)\hat{X}_{cur} + \tilde{X}_{cur}\lambda_f^{t\_o}$ dengan menentukan satu prediksi posisi objek $\hat{X}_{cur} = X_i + \left(X_i - X_j\right) * \frac{cur-i}{i-j}$ untuk $i > j$. Kemudian mencari titik terdekat dari $\hat{X}_{cur}$ yaitu $\tilde{X}_{cur}$ menggunakan jarak Euclidean. Implementasi penghalusan dapat ditunjukkan dalam kode program adalah sebagai berikut :

```
cur= Detect(j,:)+((Detect(j,:)-Detect(j-1,:)) * ((t-
point(j,1))/(point(j,1)-point(j-1,1))))
point_trj = trackedLocation(1:end-1,:);
D = pdist2(point_trj,cur,'euclidean');
closest = find(D == min(D(:)));
closest_point = trackedLocation(closest(1),:);
trackedLocation(t,:) =((1-
0.9^occ)*cur)+(closest_point*(0.9^occ));
```

Output dari beberapa proses diatas berupa hasil pelacakan objek dapat dilihat pada Gambar 4.15 sebagai berikut



Gambar 4.15. Tampilan antar muka hasil pelacakan objek.

BAB V
PENGUJIAN DAN PEMBAHASAN

Bab ini akan membahas sebuah kajian tentang pengujian metode yang digunakan terhadap perangkat lunak yang telah dibangun. Pengujian perangkat lunak meliputi pengujian hasil pelacakan dari data video uji coba dalam penyimpanan komputer. Pengujian meliputi waktu komputasi, jumlah *frame* yang dapat dilacak secara tepat oleh algoritma yang diusulkan beserta pengaruh kinerja super-resolusi terhadapat ketepatan pelacakan.

5.1 Data Uji Coba

Uji coba pada program dalam penelitian ini menggunakan video berekstensi .avi yang diperoleh dari situs online maupun pengambilan video oleh peneliti dengan menggunakan alat perekam video seperti webcam dan kamera handphone. Video yang digunakan sebanyak lima video diamana masing-masing video memiliki jumlah *frame* dan karateristik yang berbeda-beda. Daftar input video uji coba tersebut antara lain disajikan dalam Tabel 5.1.

Tabel 5.1 Data Video

No	**Nama video**	***Screenshoot* Video**	**Jumlah *Frame***
1	Motorcycle.avi		63

2	Bicycle.avi		50
3	Helikopter.avi		59
4	Paralayang.avi		60
5	Surveilance.avi		60

Dalam akuisisi video tersebut, video Helikopter.avi, Motorcycle.avi dan Surveillance.avi memiliki resolusi yang besar. Video Helikopter Motorcycle.avi dan Surveillance memiliki resolusi 1920×1080 piksel dan video Motorcycle memiliki resolusi 864×480 piksel. Namun, dalam proses penerapannya pada

pelacakan video-video tersebut akan dipotong (*cropping*) dalam ukuran 640×360 piksel untuk memperlancar waktu proses komputasi. Sedangkan video lainnya, seperti paralayang.avi dan bicycle memiliki ukuran video 640×360 yang diperoleh dari situs Youtube. Video tersebut mewakili keadaan seperti pada video Motorcycle.avi dimana objek bergerak dalam intesitas rendah (dalam bayang-bayang sebuah gedung) yang menyebabkan kemiripan intensitas antara objek dengan background. Video Helikopter.avi diakuisisi dengan kondisi objek sangat jauh dengan kamera beserta video lainnya dengan objek kecil lainnya.

5.2 Pengujian Pemilihan Objek

Pengujian pemilihan Objek bertujuan untuk mengetahui citra objek yang akan dilacak melalui ROI. Gambar beserta ukuran dari setiap citra objek yang akan dilacak dapat ditunjukan pada Gambar 5.1.

17×16 piksel

12×37 piksel

44×47 piksel

23×15 piksel

28×15 piksel

Gambar 5.1 Citra Objek yang dilacak

Selain karakteristik video dimana telah telah dijelaskan diatas, objek juga memiliki karakterisik tersendiri seperti pada video Paralayang.avi, bicycle.avi dan gardener.avi yang menunjukkan ukuran objek yang relatif kecil. Objek pada video Motorcycle memiliki bagian kecil intensitas yang berbeda berbeda dengan backgroundnya (ditunjukkan dengan warna putih) dan begitu juga untuk video Helikopter.avi dimana bagian kecil dari objek memiliki intensitas yang berbeda dengan backgroundnya (hal ini ditunjukkan dengan warna gelap pada sisi objek).

5.3 Pengujian terhadap Pelacakan Objek

Pengujian pelacakan objek menggunakan lima video yang telah ditunjukan pada Gambar 5.1. Penelitian ini memberikan tiga kriteria tingkat ketepatan dalam pelacakan objek yang ditunjukkan dalam Gambar 5.2.

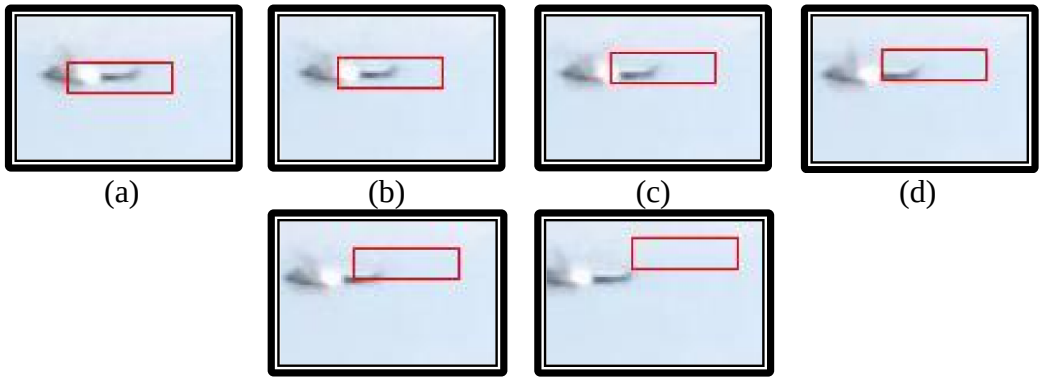
(a) Presisi (b) Kurang Presisi (c) Tidak Presisi

Gambar 5.2 . Tiga kriteria tingkat ketepatan dalam pelacakan objek

Dari gambar diatas, Sebuah pelacakan dikatakan presisi jika sebagian besar objek masih terdapat dalam area *bounding box.* Suatu pelacakan dikatakan kurang presisi jika dalam area *bounding box* hanya terdapat sebagian kecil dari objek yang dilacak dan dikatakan tidak presisi jika tidak ada objek yang dilacak dalam area *bounding box*

Pengujian pelacakan menggunakan metode *Adaptive Particle Filter* dengan parameter ukuran blok dan area pencarian yang digunakan dalam proses estimasi pergerakan objek. Masing-masing data video uji coba memiliki parameter ukuran blok dan area pencarian yang berbeda. Hal ini disebabkan karakteristik data video uji coba memiliki karakteristik yang berbeda-beda, seperti ukuran objek dan kecepatan perpindahan objek yang akan dilacak. Parameter ukuran blok tergantung terhadap ukuran besar kecilnya objek. Sementara itu, kondisi objek yang bergerak cepat dibutuhkan area pencarian yang besar. Dari beberapa kali proses pengujian, Video Motorcycle, Paralayang dan Bicycle menggunakan parameter ukuran blok sebesar 15 dan parameter area pencarian sebesar 15. Video Surveillance menggunakan parameter ukuran blok sebesar 21 dan parameter area pencarian sebesar 20. Sedangkan video Helikopter menggunakan parameter ukuran blok sebesar 30 dan parameter area pencarian sebesar 25.

Hasil pelacakan objek dari kelima data video uji coba akan dibandingkan berdasarkan *frame* yang telah di proses super-resolusi maupun tanpa super-resolusi. Tujuannya adalah melihat pengaruh perubahan tingkat presisi pelacakan objek berdasarkan citra super-resolusi. Pemberian parameter jumlah partikel juga digunakan dalam pengujian untuk menganalisa pengaruh parameter tersebut terhadap pelacakan. Gambar 5.3 mengilustrasikan sebuah hasil pelacakan dalam video Helikopter.



Gambar 5.3. Ilustrasi hasil pelacakan

Pada Gambar 5.3, bagian (a), (b) dan (c) menunjukkan hasil pelacakan yang presisi berdasarkan kriteria yang telah diberikan seperti dalam Gambar 5.2. Pelacakan objek kurang presisi ditunjukkan pada Gambar 5.3. bagian (d), (e) dan pelacakan menjadi tidak presisi ditunjukkan pada Gambar 5.3 bagian (f). Untuk perhitungan selajutnya pada keseluruhan data video uji coba, hasil pelacakan objek tanpa proses super-resolusi dengan parameter jumah partikel yang berbeda beserta waktu komputasinya disajikan dalam Tabel 5.3.1.

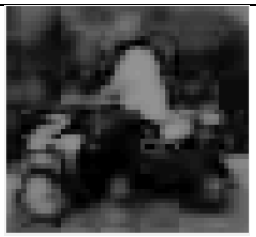
Tabel 5.2 Hasil Pelacakan objek tanpa super-resolusi

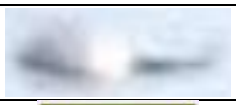
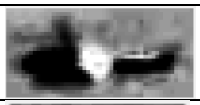
Jumlah Partikel	Nama Video	Total hasil pelacakan (*frame*)			Rata-rata waktu komputasi per-*frame* (detik)
		Tidak Presisi	Kurang Presisi	Presisi	
15	Motorcycle.avi	46	6	11	0,110019
	Bicycle.avi	16	15	19	0,099109

	Helikopter.avi	23	6	30	0,11667
	Paralayang.avi	16	13	31	0,078969
	Surveilance.avi	55	2	3	0,125799
50	Motorcycle.avi	38	4	21	0,217332
	Bicycle.avi	13	23	14	0,109011
	Helikopter.avi	30	2	27	0,189964
	Paralayang.avi	0	12	48	0,102185
	Surveilance.avi	55	2	3	0,139125
75	Motorcycle.avi	36	6	21	0,299556
	Bicycle.avi	17	13	20	0,185231
	Helikopter.avi	18	10	31	0,259764
	Paralayang.avi	0	14	46	0,120957
	Surveilance.avi	55	2	3	0,160729

Kemudian hasil pelacakan dengan nilai parameter yang sama akan dibandingkan dengan pelacakan berbasis citra super-resolusi untuk mengetahui pengaruhnya terhadap hasil pelacakan. Citra super-resolusi berdasarkan serangkaian citra resolusi rendah yang dibangun dengan proses registrasi citra menggunakan *Phased Based Image Matching* (PBIM) dan rekonstruksi citra menggunakan *Projection onto Convex Set* (POCS). Tabel 5.3 menunjukkan citra objek yang telah dipilih melalui ROI beserta hasil citra super-resolusinya.

Tabel 5.3 Tampilan objek citra dan hasil citra super-resolusi

Nama Video	Objek Citra	Citra objek *grayscale* resolusi rendah	Citra objek grayscale super-resolusi
Motorcycle.avi			
Bicycle.avi			

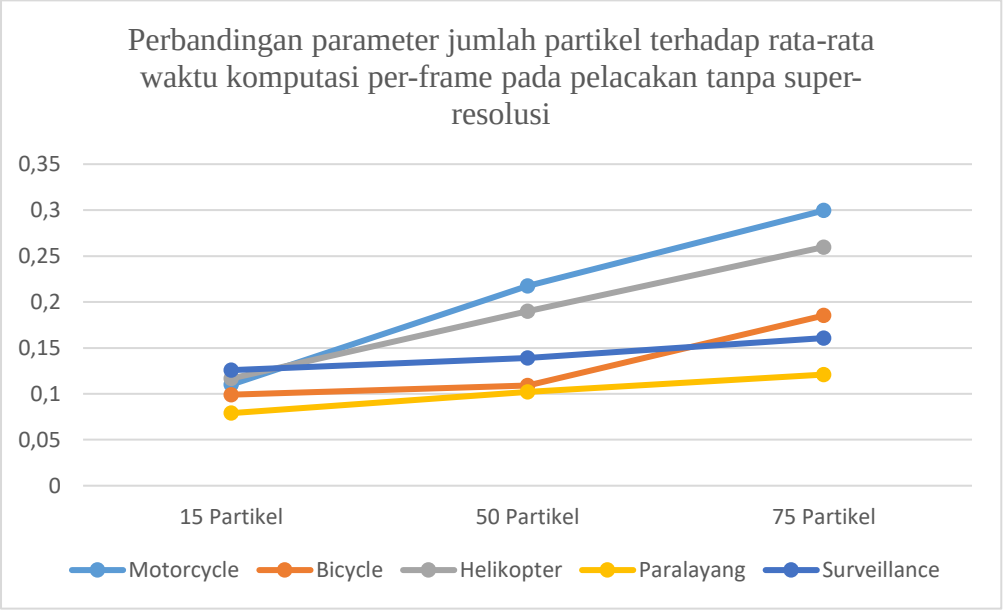
Helikopter.avi			
Paralayang.avi			
Surveillance.avi			

Selanjutnya, hasil citra super-resolusi akan diimplementasikan dalam proses pelacakan. Tabel 5.3.2 menyajikan hasil pelacakan objek berbasis citra super-resolusi dengan parameter jumah partikel yang berbeda beserta waktu komputasinya.

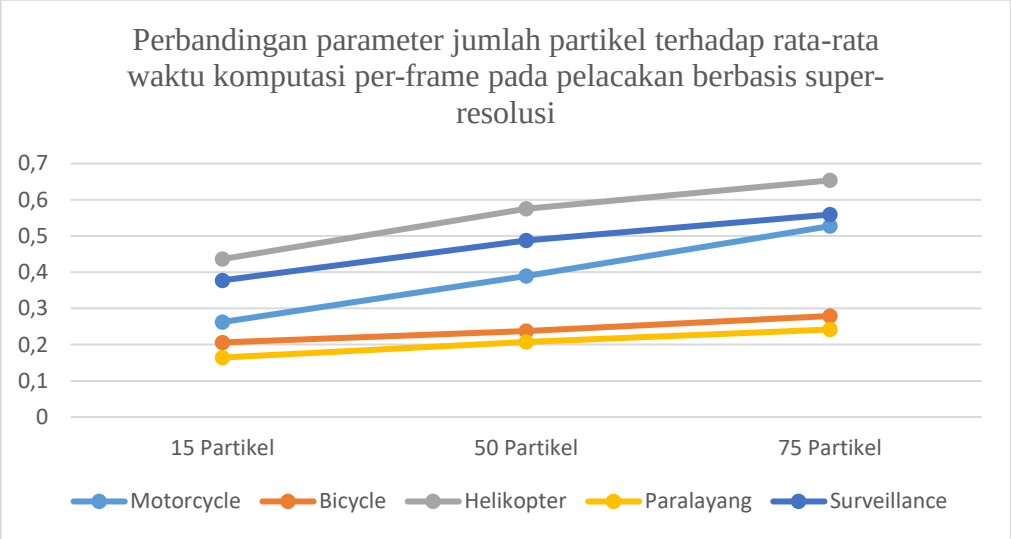
Tabel 5.4 Hasil pelacakan objek berbasis citra super-resolusi

Jumlah Partikel	Nama Video	Total hasil pelacakan (*frame*)			Rata-rata waktu komputasi per-*frame* (detik)
		Tidak Presisi	Kurang Presisi	Presisi	
15	Motorcycle.avi	3	6	54	0,262321
	Bicycle.avi	0	0	50	0,20606
	Helikopter.avi	21	8	30	0,436205
	Paralayang.avi	0	8	52	0,164418
	Surveilance.avi	9	9	42	0,377487
50	Motorcycle.avi	3	6	54	0,38989
	Bicycle.avi	0	0	50	0,237736
	Helikopter.avi	22	1	36	0,575466
	Paralayang.avi	0	0	60	0,207267
	Surveilance.avi	9	4	47	0,488043
75	Motorcycle.avi	3	6	54	0,527143
	Bicycle.avi	0	0	50	0,279024
	Helikopter.avi	22	1	36	0,653502
	Paralayang.avi	0	0	60	0,241707
	Surveilance.avi	17	6	37	0,559162

Dari beberapa hasil diatas dalam Tabel 5.3 dan Tabel 5.4, parameter jumlah partikel mempengaruhi tingkat presisi dalam proses pelacakan. Semakin besar nilai paramater jumlah partikel maka tingkat akurasi pelacakan cenderung lebih tinggi tergantung dari ukuran besarnya objek yang dilacak. Pada pelacakan berbasis super-resolusi, pemberian nilai parameter dapat menggunakan jumlah partikel yang lebih kecil karena akan ditemukan hasil yang sama pada nilai parameter jumlah partikel yang lebih tinggi. Namun, pemberian nilai parameter jumlah partikel yang terlalu besar atau kecil akan mengurangi akurasi pelacakan seperti yang terjadi pada video Surveillance. Berikut ini, sebuah grafik yang menampilkan waktu komputasi dari pelacakan dengan parameter yang berbeda

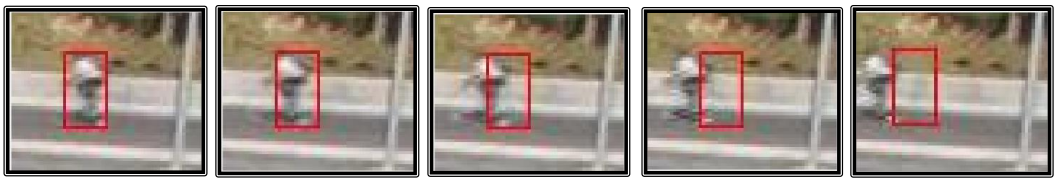


Gambar 5.4. Perbandingan parameter jumlah partikel terhadap rata-rata waktu komputasi per-*frame* pada pelacakan tanpa super-resolusi



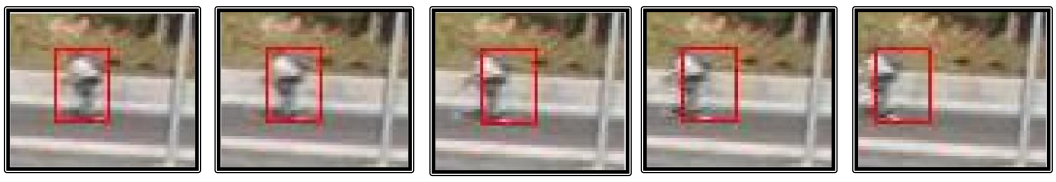
Gambar 5.5. Perbandingan parameter jumlah partikel terhadap rata-rata waktu komputasi per-*frame* pada pelacakan berbasis super-resolusi

Hal lain yang dapat dilihat dalam Tabel 5.2 dan Tabel 5.4 adalah perbedaan jumlah *frame* yang dapat dilacak lebih tepat pada pelacakan berbasis citra super-resolusi dibandingkan pelacakan objek tanpa proses super-resolusi. Sebagai contoh perbandingan hasil pelacakan dalam video Surveillance, Sebuah visualisasi pelacakan dalam sebuah *scene* yang sama ditunjukkan dalam Gambar 5.6



Frame 1 *Frame* 2 *Frame* 3 *Frame* 4 *Frame* 5

(a)



Frame 1 *Frame* 2 *Frame* 3 *Frame* 4 *Frame* 5

(b)

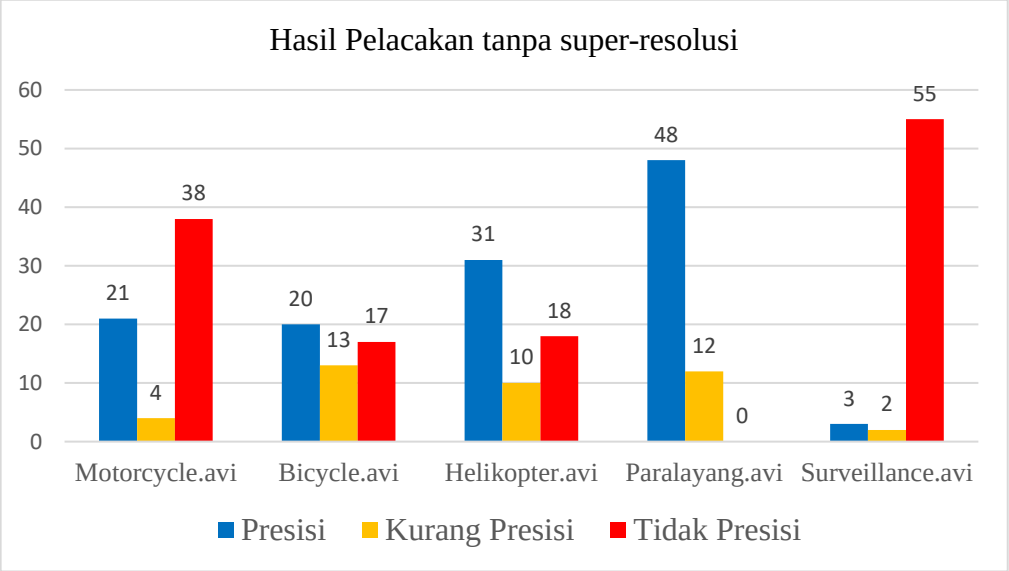
Gambar 5.6. Hasil visualisasi pelacakan. (a) Hasil pelacakan tanpa proses super-resolusi. Hasil pelacakan berbasis super-resolusi

Gambar 5.6 menampilkan sebuah hasil pelacakan dengan *scene* yang sama dari video Surveillance. Hasil pelacakan tanpa super-resolusi ditampilkan dalam Gambar 5.6.(a) sedangkan hasil pelacakan berbasis super-resolusi ditampilkan dalam Gambar 5.6.(b). Hasil visualisasi pelacakan pada Gambar5.4.a menunjukkan pelacakan tanpa super-resolusi mengalami kondisi kriteria kurang presisi pada *frame* 4 dan tidak presisi pada gambar bagian *frame* 5. Namun, hal tersebut tidak dialami dalam pelacakan berbasis super-resolusi bahkan keseluruhan *frame* pada Gambar5.4.(b) objek dapat dilacak dengan tepat.

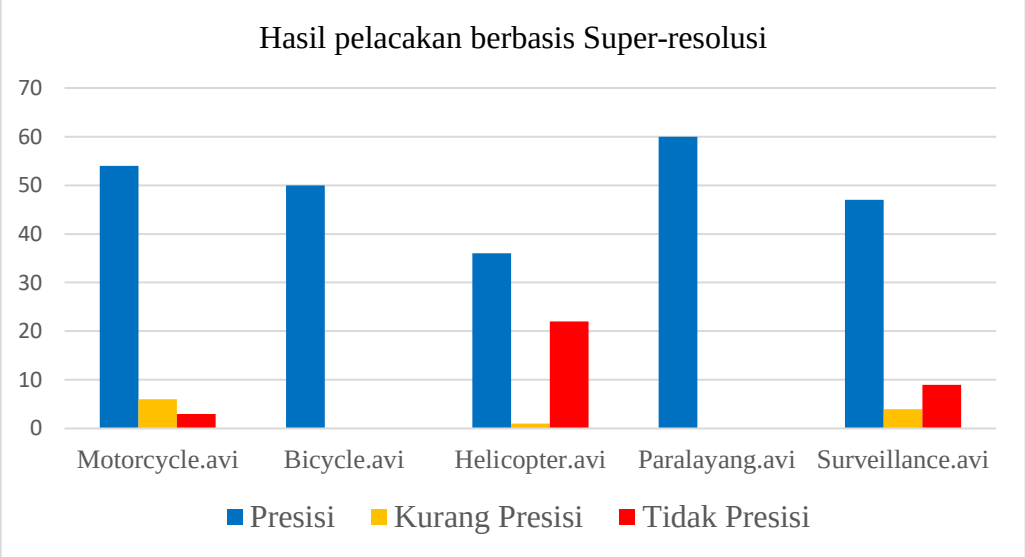
Perhitungan selanjutnya adalah hasil pelacakan dari keseluruhan data video uji coba, baik pelacakan berbasis super-resolusi maupun pelacakan tanpa super-resolusi, dengan parameter jumlah parameter yang efisien dari segi waktu komputasi dapat dilihat pada Tabel 5.5. Kemudian bentuk visualisasi dari Tabel 5.5 dalam sebuah grafik dapat dilihat pada Gambar 5.7 dan Gambar 5.8.

Tabel 5.5. Perbandingan hasil pelacakan (*frame*)

No.	**Nama Video**	**Pelacakan tanpa super-resolusi**			**Pelacakan berbasis citra super-resolusi**		
		Tidak presisi	Kurang presisi	Presisi	Tidak Presisi	Kurang presisi	Presisi
1	Motorcycle.avi	38	4	21	3	6	54
2	Bicycle.avi	17	13	20	0	0	50
3	Helikopter.avi	18	10	31	22	1	36
4	Paralayang.avi	0	12	48	0	0	60
5	Surveillance.avi	55	2	3	9	4	47



Gambar 5.7 Grafik tingkat akurasi pelacakan objek tanpa super-resolusi.



Gambar 5.8. Grafik tingkat akurasi pelacakan objek berbasis super-resolusi

Peningkatan presisi pada pelacakan objek berbasis citra super-resolusi terhadap pelacakan tanpa super-resolusi terjadi jika semakin banyaknya objek terlacak secara tepat pada video dan semakin berkurangnya hasil pelacakan objek dengan kriteria kurang presisi maupun tidak presisi.

Untuk mengetahui lebih jauh pengaruh super-resolsi terhadap proses pelacakan, proses pelacakan akan diuji berdasarkan citra super-resolusi dengan

jumlah citra referensi yang berbeda. Jumlah citra referensi citra menentukan kualitas dari citra super-resolusi dimana nilai translasi yang diperoleh dari proses registrasi citra digunakan untuk menempatkan piksel-piksel yang belum diketahui dalam citra super-resolusi. Tabel 5.6 menunjukkan hasil pelacakan berdasarkan super-resolusi dengan jumlah citra referensi yang berbeda.

Tabel 5.6. Hasil pelacakan berdasarkan jumlah citra referensi

Jumlah referensi super-resolusi	Nama Video	Total hasil pelacakan (*frame*)		
		Tidak Presisi	Kurang presisi	Presisi
Dua Citra	Motorcycle.avi	3	6	54
	Bicycle.avi	0	0	50
	Helikopter.avi	22	1	36
	Paralayang.avi	0	0	60
	Surveilance.avi	9	4	47
Tiga Citra	Motorcycle.avi	3	6	54
	Bicycle.avi	0	0	50
	Helikopter.avi	22	1	36
	Paralayang.avi	0	0	60
	Surveilance.avi	9	4	47
Empat Citra	Motorcycle.avi	3	6	54
	Bicycle.avi	0	0	50
	Helikopter.avi	22	1	36
	Paralayang.avi	0	0	60
	Surveilance.avi	9	4	47

Tabel diatas menunjukkan bahwa penambahan jumlah citra referensi dalam proses registrasi super-resolusi tidak mempengaruhi tingkat akurasi proses pelacakan.

5.4 Pembahasan Hasil Pengujian

Pembahasan hasil pengujian membahas hasil pengujian dari proses pelacakan objek kecil pada video yang telah tersedia. Untuk mengetahui persentase akurasi pelacakan objek kecil dengan maupun tanpa proses super-resolusi maka sebuah perhitungan persentase akurasi prototipe perangkat lunak akan diuji dengan menggunakan Persamaan 5.1.

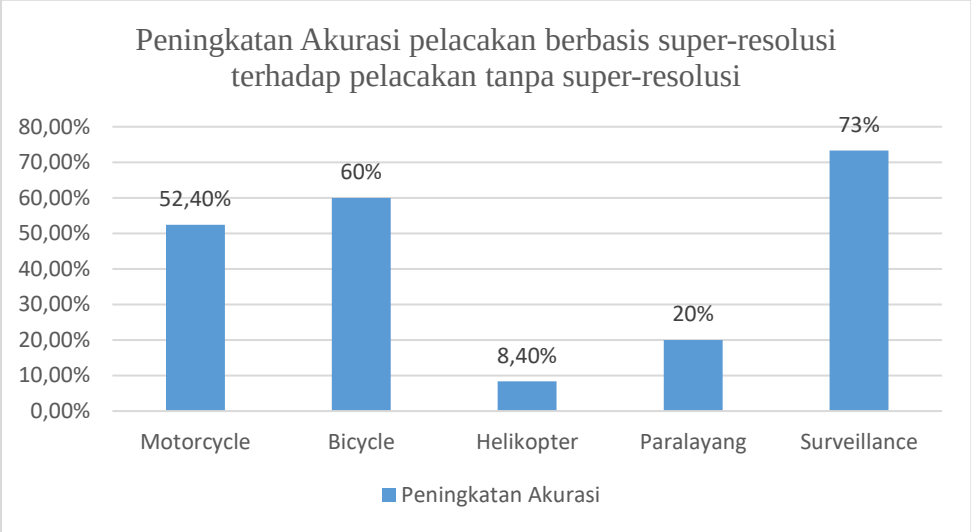
$$PA = \frac{JT}{JS} x\ 100\% \quad (5.1)$$

dengan PA adalah persentase akurasi, JT adalah jumlah *frame* ketika objek dapat dilacak secara tepat, dan JS adalah jumlah dari seluruh *frame* video. Dari hasil perbandingan pelacakan seperti pada tabel 5.3.5, persentase akurasi pelacakan yang dibandingkan ditunjukan pada tabel 5.7

Tabel 5.7. Perbandingan hasil persentase akurasi pelacakan objek

No	**Nama Video**	**Jumlah *Frame***	**Jumlah objek terlacak (presisi)**	**Persentase akurasi**
Pelacakan tanpa super-resolusi				
1	Motorcycle.avi	63	21	33,3%
2	Bicycle.avi	50	20	40%
3	Helikopter.avi	59	31	52,5%
4	Paralayang.avi	60	48	80%
5	Surveillance.avi	60	2	5%
Pelacakan berbasis super-resolusi				
1	Motorcycle.avi	63	54	85,7%
2	Bicycle.avi	50	50	100%
3	Helikopter.avi	59	36	61%
4	Paralayang.avi	60	60	100%
5	Surveillance.avi	60	47	78,3%

Kemudian dari tabel 5.7 diatas, peningkatan akurasi pelacakan dapat diketahui dengan menghitung selisih tingkat akurasi hasil pelacakan tanpa super-resolusi terhadap pelacakan berbasis super-resolusi. Persentase peningkatan akurasi pelacakan dapat ditunjukkan dalam Gambar 5.9



Gambar 5.9. Peningkatan akurasi pelacakan berbasis super-resolusi terhadap pelacakan tanpa super-resolusi

Peningkatan akurasi pelacakan objek pada video Motorcycle, Bicycle dan Surveillance masing-masing sebesar 52,4 %, 60% dan 73,3 %. Hal ini juga terjadi pada video paralang dengan peningkatan sebesar 20%. Peningkatan presisi paling rendah terjadi pada video Helikopter yaitu, hanya mencapai 8,4 %.

5.4 Pembahasan Penyebab Kecilnya Persentase Akurasi

Meskipun metode pelacakan objek telah diintegrasikan dengan super-resolusi, masih terdapat beberapa kendala yang terjadi dalam proses pelacakan untuk melacak objek lebih tepat. Beberapa hal tersebut diantaranya adalah sebagai berikut :

1. Ketidaksesuaian ukuran blok dan area pencarian yang digunakan untuk mengestimasi gerakan objek.
2. Objek memiliki kesamaan warna dengan background seperti kasus yang terjadi pada video Bicycle dan Surveillance yang menyebabkan sulitnya mengetahui vektor gerakan oleh estimasi pergerakan objek.
3. Adannya pergerakan kamera saat akuisisi video.

   Seperti yang terjadi *frame* ke-33 dan *frame* ke-34 pada video Helikopter yaoitu terjadi getaran pada kamera pada saat akuisisi video sehingga menyebabkan kesalahan estimasi gerak objek.

BAB VI
PENUTUP

Bab ini menjelaskan tentang beberapa kesimpulan yang dapat diambil berdasarkan penelitian yang telah dilaksanakan. Disamping itu, terdapat beberapa saran yang diberikan untuk digunakan dalam pengembangan penelitian ini lebih lanjut.

6.1 Kesimpulan

Berdasarkan pengujian dan pembahasan dari penelitian yang telah dilakukan terhadap prototipe perangkat lunak untuk pelacakan objek kecil menggunakan metode *Adaptive Particle Filter* berbasis super-resolusi, maka dapat diambil beberapa kesimpulan diantaranya adalah sebagai berikut:

1. Penelitian ini telah berhasil menerapkan teknik super-resolusi *multi-frame* berdasarkan serangkaian *frame* video termasuk diantaranya beberapa tahap untuk mengetahui nilai translasi citra pada proses registra menggunakan *Phased Based Image Matching* (PBIM) dan pembangunan ulang citra pada proses rekonstruksi menggunakan *Projection onto Convex Sets* (POCS)
2. Penelitian ini juga telah berhasil menerapkan metode *Adaptive Particle Filter* berdasarkan serangkaian citra super-resolusi untuk melakukan pelacakan pada objek kecil dari video yang tersedia kemudian mengimplementasikannya ke dalam software MATLAB berupa prototype perangkat lunak. Proses pelacakan dilakukan dengan melalui beberapa tahapan: Input video, *convert* video ke *frame*, preprosessing video, super-resolusi, pemilihan ROI, estimasi gerak dengan *Hierarchical Block Matching* dan pelacakan objek kecil menggunakan metode *Adaptive Particle Filter*.
3. Dengan mengimplementasikan citra super-resolusi pada proses pelacakan, peningkatan akurasi terjadi dalam proses pelacakan. Peningkatan signifikan terdapat pada video Motorcycle, Bicycle dan Surveillance masing-masing sebesar 52,4%, 60% dan 73,3 %. Hal ini juga terjadi pada

video paralang dengan peningkatan sebesar 20%. Peningkatan presisi pada video Helikopter hanya mencapai 8,4 % karena adanya faktor-faktor yang menyebabkan kesalahan estimasi dalam proses pelacakan seperti adanya getaran pada saat akuisisi video.

4. Parameter jumlah partikel mempengaruhi tingkat akurasi pelacakan. Semakin tinggi nilai parameter jumlah partikel maka tingkat akurasi pelacakan cenderung lebih tinggi tergantung dari objek yang akan dilacak. Namun, penambahan jumlah citra referensi dalam proses registrasi super-resolusi tidak mempengaruhi tingkat pelacakan lebih jauh.

6.2 Saran

Berdasarkan hasil yang telah diperoleh pada penelitian ini, ada beberapa hal yang dapat dijadikan sebagai bahan masukan untuk perbaikan kinerja dan pengembangan selanjutnya diantaranya sebagai berikut:

1. Akuisisi video dengan kondisi lain seperti akuisisi dengan kondisi translasi maju (zoom out) atau mundur (zoom in) dan rotasi.
2. Karena video dalam penelitian ini hanya sebatas pada kasus video dengan objek yang rigid atau tidak mengalami penskalaan dan rotasi, penelitian selanjutnya dapat menggunakan metode estimasi gerakan objek Scale Invariant Feature Transform (SIFT) yang tahan terhadap faktor penskalaan dan rotasi.

DAFTAR PUSTAKA

[1] Barjatya, A., (2004),"Block Matching Algorithms For Motion Estimation", *Student Member*, IEEE, DIP 6620 Spring.

[2] Davieshy, D., Palmeth, P., dan Mirmehdio, M. (1998), " Detection And Tracking of Very Small Low Contrast Object", eds. Mark Nixon dan John Carters, *Proceedings of the British Machine Conference*, BMVA Press, hal 60.1-60.10

[3] Fan, C., Zhu, J., Gong, J and Kuang, C. (2006),"POCS Super-esolution Sequence Image Reconstruction Based on Improvement Approach of Keren Registration Method". *Sixth International Conference on Intelligent Systems Design and Applications* (ISDA'06), Jinan, vol. 2, 2006.

[4] Hao, S., Lin, L., Weiping Z., and Limin, L. (2009), "Location and Super-resolution Enhancement of License Plates Based on Video Sequences", *International Conference on Information Science and Engineering* (ICISE), hal 1319 – 1322

[5] Huang, Y., Llach, J. (2008), "Tracking the small object through Clutter with Adaptive Particle Filter", *Audio, Language and Image Processing*, hal 357 – 362

[6] Lefevre, S., Foissotte, T dan Vincent, N, (2003),"Detection and Tracking of Small Objects by Details Removal", *Proceedings of 4th European Workshop on Image Analysis for Multimedia Interactive Services*, University of London, Queen Mary, hal 137-140.

[7] Mise, O dan Breckon, T.P. (2013), "Super-Resolution Imaging Applied to Moving Targets in High Dynamic Scenes". *Proceedings Of SPIE-Emerging*

Technologies in Security and Defence and Quantum Security II and Unmanned Sensor Systems X, Eds. Edward M. Carapezza, et al., Vol. 8899

[8] Nam, K.M., Kim, J.S dan Park., R.H, (1995) "A Fast Hierarchical Motion Vector Estimation Algorithm Using Mean Pyramid", IEEE Transactions on Circuits and Systems for Video Technology ,Vol. 5, Issue. 4.

[9] Panchal, N., Limbasiya, B., dan Prajapati, A. (2013)," Survey On Multi-Frame Image Super-Resolution", *International Journal Of Scientific & Technology Research*, Vol 2, hal 11, 2013.

[10] Rupesh, K.R. (2013), *A Survey on Object Detection and Tracking Algorithms*, Tesis., National Institute of Technology Rourkela, Rourkela.

[11] Setiyono, B., Hariadi, M., Purnomo, H.M. (2012),"Survei of Superresolution using Phased Based Image Matching"**,** *Jurnal Of Theoretical and Applied Information technology*, Vol.43, No.2

[12] Stark, H dan Oskoui, P., (1989)," High-resolution Image Recovery from Image-plane Arrays Using Convex Projections", *JOSA* A, Optical Society of America, Vol.6, No.11, hal 1715-1726.

[13] Stauffer, C dan Grimson, W.E.L., (1999), "Adaptive Background Mixture Models for Real-time Tracking", *Computer Vision and Pattern Recognition*, IEEE Computer Society Conference, Vol.2 , hal 252.

[14] Yilmaz, A., Javed, O., and Shah, M. (2006), "Object Tracking: A survey", *ACM Computing* Survei, Vol 38, No 4, Article 13.

LAMPIRAN A

A. Kode Program Pelacakan dengan Metode *Adaptive Particle Filter*.

```
function [trackedLocation] = APF2 ( jmlframe, roi, mbSize , search,
particle , frame, frameSR)

for i=1:jmlframe
    t = t+1;
    frame1= frameSR{t};
    frame2= frame1;
    citra = frame{t
      if t==1
           roi=2*[roi(1) roi(2) roi(3) roi(4)];
           template = imcrop(frame2,roi
           cent = [roi(1)+uint32(roi(3)/2)
roi(2)+uint32(roi(4)/2)];
           trackedLocation(t,:)=cent
           Detect(j,:)=trackedLocation(t,:);
           %% generate particle at time t-1
           N = particle;%number of particle
           ON = ones(1 , N);
           X3 = zeros(2, N);
           cte = 1/N;
           cteN = cte(1 , ON);
           w = cteN;
           X =
bsxfun(@plus,trackedLocation(t,:)',sqrt(varians)*randn(2,N));
           Xt = X;
           label = 'initial';
           track = [(trackedLocation(t,1)-(roi(3)/2))+1
(trackedLocation(t,2)-(roi(4)/2))+1];
           regionTrack = [track roi(3) roi(4)]/2 ;
           annotateTrackedObject(regionTrack);
           point(j,1) = t;
           prevFrame = frame2;
      else
           X = bsxfun(@plus,trackedLocation(t-
1,:)',sqrt(varians)*randn(2,N));
          [motion] =
Hierarchical(prevFrame,frame2,2*mbSize,2*search,trackedLocation(t-
1,:));
           motion = motion(2:end,2:end);
           c =(find(motion==max(motion(:))));
           u = real(motion(c(1)))
           v = imag(motion(c(1
           Xt = repmat (X,1,1);
           if (u~=0 || v~=0)
        for i=1:N
            Ix = prevFrame(pixel(2,i),pixel(1,i)+1)-
prevFrame(pixel(2,i),pixel(1,i));
            Iy = prevFrame(pixel(2,i)+1,pixel(1,i))-
prevFrame(pixel(2,i),pixel(1,i));
```

```
            It = frame2(pixel(2,i),pixel(1,i)) -
prevFrame(pixel(2,i),pixel(1,i));
            miu(i) = abs((u*Ix)+(v*Iy)+It);
            Xt(1,i)= normrnd(X(1,i)+u,miu(i));
            Xt(2,i)= normrnd(X(2,i)+v,miu(i));
        end
    else
        for i=1:N
            Ix = prevFrame(pixel(2,i),pixel(1,i)+1)-
prevFrame(pixel(2,i),pixel(1,i));
            Iy = prevFrame(pixel(2,i)+1,pixel(1,i))-
prevFrame(pixel(2,i),pixel(1,i));
            It = frame2(pixel(2,i),pixel(1,i)) -
prevFrame(pixel(2,i),pixel(1,i));
            miu(i) = abs((u*Ix)+(v*Iy)+It);
        end
        Xt(1,:)= normrnd(X(1,i)+u,max(miu));
        Xt(2,:)= normrnd(X(2,i)+v,max(miu));
    end
          [ J , stdIT , r ] = SSD (frame2,template, Xt ,N ,
mbSize);
          a = 0; b = 1;
          q0 = 0.5;
          C = 1/sum(r);
          for i= 1:N
             qj = (1-0.5)/J(i);
             p = qj*(normrnd(r(i),stdIT(i),[1 J(i)]));
             P1(i)= (q0*unifrnd(a,b))+(C*sum(p));
          end
          if (u~=0 || v~=0)
          %average object speed
            delta2 = 0;
            for o=-30:-1
             if(t+o<=0)
               Xs = [0 0];
             else
               Xs = trackedLocation(t+o,:);
             end
             if(t+o-1<=0)
               Xs1 = [0 0];
             else
               Xs1 = trackedLocation(t+o-1,:);
             end
             delta2_0 = Xs - Xs1;
             delta2 = delta2 + delta2_0;
            end
            delta2 = delta2/30 ;

            %particle speed
            for i=1:N
               delta1 =[ abs(X(1,i)-Xt(1,i)) abs(X(2,i)-Xt(2,i))];
               d = ((delta1(1)- delta2(1))^2+ (delta1(2)-
delta2(2))^2);
```

```
            P2(i)= (1/sqrt(2*pi))*(exp(-1/2 *(d/(1.^2))));
         end
         w = P1.*P2;
         j=j+1;
         point(j,1)= t;
      % Normalize weight vector
         wk = w./sum(w);
      % Calculate effective sample size
         Neff = 1/sum(wk.^2);
         resample_percentaje = 0.50;
         Nt = resample_percentaje*N;
         if Neff < Nt
            disp('Resampling ...\n')
            [index] = resample(w);
            Xt = Xt(:,index);
            w=ones(1,N)./N;
         end
% weighted average
            Detect (j,:) = (sum(bsxfun(@times,Xt,w),2)/sum(w))';
            trackedLocation(t,:) = Detect(j,:);
            track = [(trackedLocation(t,1)-(roi(3)/2))+1
      (trackedLocation(t,2)-(roi(4)/2))+1];
           regionTrack = [track roi(3) roi(4)]/2 ;
           annotateTrackedObject(regionTrack);
           prevFrame = frame2
      else
         fprintf('no motion')
         occ = occ+1;
         koef_poly = polyfit
      trackedLocation(:,1),trackedLocation(:,2),2);
         for i=1:N
         poly = [ Xt(1,i)^2 Xt(1,i) 1];
         d_trj = abs (Xt(2,i)-(sum(koef_poly.*poly)));
         P2(i)= (1/sqrt(2*pi))*(exp(-1/2
      *((d_trj/(0.9^occ))/1).^2));
         end
         w = P1.*P2 ;
         %% Normalize weight vector
         wk = w./sum(w);
         %% Calculate effective sample size
         Neff = 1/sum(wk.^2);
         resample_percentaje = 0.50;
         Nt = resample_percentaje*N;
            if Neff < Nt
               disp('Resampling ...\n')
               [index] = resample(w);
                Xt = Xt(:,index);
                w=ones(1,N)./N;
            end
         max_weight = find(w==max(w(:)));
         num = size(max_weight, 2);
          if(num > 1)
             for l=1:num
```

```
                    X_baru(1,l) = Xt(1, max_weight(l));
                    X_baru(2,l) = Xt(2, max_weight(l));
                end
                maks = w(1,max_weight(1));
                trackedLocation(t,:)
            =(sum(bsxfun(@times,X_baru,maks),2)/num*maks)';
             else
                trackedLocation(t,1) = Xt(1,max_weight);
                trackedLocation(t,2) = Xt(2,max_weight);
             end
            if t==2 || t==3
                cur = trackedLocation(t-1,:);
                closest_point = cur;
                trackedLocation(t,:) =((1-
            0.9^occ)*cur)+(closest_point*(0.9^occ));
                      j=j+1;
                 point(j,1)= t;
                 Detect(j,:)= trackedLocation(t-1,:);
            else
                cur= Detect(j,:)+((Detect(j,:)-Detect(j-1,:)) *
            ((t-point(j,1))/(point(j,1)-point(j-1,1))))
                point_trj = trackedLocation(1:end-1,:);
            D = pdist2(point_trj,cur,'euclidean');
                closest = find(D == min(D(:)));
                closest_point = trackedLocation(closest(1),:);
                trackedLocation(t,:) =((1-
            0.9^occ)*cur)+(closest_point*(0.9^occ));
            end
            track = [(trackedLocation(t,1)-(roi(3)/2))+1
            (trackedLocation(t,2)-(roi(4)/2))+1];
            regionTrack = [track roi(3) roi(4)]/2 ;
            annotateTrackedObject(regionTrack);
            prevFrame = frame2;
             end
      end
 end
function annotateTrackedObject(regionTrack)
       if (isnan(track))
          videoPlayer.step(citra);
       else
        Imf = insertShape(citra, 'Rectangle', regionTrack, 'Color',
'red');
        imshow(Imf);
        writeVideo(writerObj,Imf);
       end
end
 close(writerObj);
end
```

LAMPIRAN C

C. Kode Program Sum of Square Difference (SSD)

```
function [ J , stdIT , r ] = SSD( frame, template, particle, N,
neib)
J = 1;
stdIT = 1;
r = 1;
particle = round(particle);
    for(l=1:N)
        for (i= -neib :1:-1)
            if (particle(1,l)+i > 0 && particle(2,l)+i > 0 &&
particle(1,l)-i < size(frame,2) && particle(2,l)-i < size(frame,1))
                input_image = frame (particle(2,l)+i:particle(2,l)-
i,particle(1,l)+i:particle(1,l)-i);
                [N_SSD, var, SSD]=SSDscore(template,input_image);
                [x,y] = find(N_SSD>0.89);
                J(l) = length(x);
                stdIT(l) = var;
                [x,y]=find(N_SSD==max(N_SSD(:)))  ;
                r(l) = SSD(x(1),y(1));
                break
            end
        end
    end

function [N_SSD, stdT , SSD] = SSDscore (T,I)
 if(nargin<3), IdataIn=[]; end
[N_SSD, stdT , SSD]=template_match(T,I,IdataIn);

function [N_SSD, stdT , SSD]=template_match(T,I,IdataIn)
T_size = size(T); I_size = size(I);
outsize = I_size + T_size-1;
FT = fft2(rot90(T,2),outsize(1),outsize(2));
Idata.FI = fft2(I,outsize(1),outsize(2));
Icorr = double(real(ifft2(Idata.FI.* FT)));
Idata.LocalQSumI= double(local_sum(I.*I,T_size));

QSumT = sum(T(:).^2);

% SSD antara template dan citra
SSD=Idata.LocalQSumI+QSumT-2*Icorr;
 IXI=size(Idata.LocalQSumI);
 T2=size(QSumT);
 TI=size(Icorr);
% Normalisasi pada range 0..1
N_SSD=SSD-min(SSD(:));
if N_SSD~=0
N_SSD=1-(N_SSD./max(N_SSD(:)));
else
N_SSD=1-(N_SSD./10);
end
```

LAMPIRAN C (Lanjutan)

```
% Menghapus Padding
N_SSD=unpadarray(N_SSD,size(I));
Idata.LocalSumI= local_sum(I,T_size);

% STD=std(T(:))
T=double(T(:));
stdT=double(sqrt(numel(T)-1))*std(T(:));

function B=unpadarray(A,Bsize)
Bstart=ceil((size(A)-Bsize)/2)+1;
Bend=Bstart+Bsize-1;
B=A(Bstart(1):Bend(1),Bstart(2):Bend(2));

function local_sum_I= local_sum(I,T_size)
% menambahkan padding
B = padarray(I,T_size);
    s = cumsum(B,1);
    c = s(1+T_size(1):end-1,:)-s(1:end-T_size(1)-1,:);
    s = cumsum(c,2);
    local_sum_I= s(:,1+T_size(2):end-1)-s(:,1:end-T_size(2)-1);
```

LAMPIRAN D

D. Kode Fungsi Resampling

```
function [ indx ] = resample(w)
N = length(w);
M = length(w);
w = w / sum(w);
indx = zeros(1, N);
Ns = floor(N.* w);
R = sum(Ns);
i = 1;
j = 0;
while j < M
    j = j + 1;
  cnt = 1;
  while cnt <= Ns(j)
    indx(i) = j;
    i = i + 1; cnt = cnt + 1;
  end;
end;
N_rdn = N - R;
Ws =(N*w - Ns)/N_rdn;
Q = cumsum(Ws);
while(i <= N)
    sampl = rand;
    j = 1;
    while(Q(j) < sampl),
        j = j + 1;
    end;
    indx(i) = j;
    i = i + 1;
end
```

LAMPIRAN D (Lanjutan)

LAMPIRAN E

E. Kode Program *Phased Based Image Matching* (PBIM)

```
function [delta_est, phi_est] = PBIM(im)
for imnr = 2:length(im)
    im0{1} = im{1};
    im1{1} = im{imnr};
 [output shift] = dftregistration(fft2(im0{1}),fft2(im1{1}),1000);
    phi_est(imnr) = output(1,2);
    delta_est(imnr,:) = output(3:4);
    x =phi_est(imnr);
    x1 = output(1,2);
    y=delta_est(imnr);
    y1 = output(3:4);
end

function [output Greg] = dftregistration(buf1ft,buf2ft,usfac)
if usfac == 0,
    CCmax = sum(sum(buf1ft.*conj(buf2ft)));
    rfzero = sum(abs(buf1ft(:)).^2);
    rgzero = sum(abs(buf2ft(:)).^2);
    error = 1.0 - CCmax.*conj(CCmax)/(rgzero*rfzero);
    error = sqrt(abs(error));
    diffphase=atan2(imag(CCmax),real(CCmax));
    output=[error,diffphase];
elseif usfac == 1,
    [m,n]=size(buf1ft);
    CC = ifft2(buf1ft.*conj(buf2ft));
    [max1,loc1] = max(CC);
    [max2,loc2] = max(max1);
    rloc=loc1(loc2);
    cloc=loc2;
    CCmax=CC(rloc,cloc);
    rfzero = sum(abs(buf1ft(:)).^2)/(m*n);
    rgzero = sum(abs(buf2ft(:)).^2)/(m*n);
    error = 1.0 - CCmax.*conj(CCmax)/(rgzero(1,1)*rfzero(1,1));
    error = sqrt(abs(error));
    diffphase=atan2(imag(CCmax),real(CCmax));
    md2 = fix(m/2);
    nd2 = fix(n/2);
    if rloc > md2
        row_shift = rloc - m - 1;
    else
        row_shift = rloc - 1;
    end

    if cloc > nd2
        col_shift = cloc - n - 1;
    else
        col_shift = cloc - 1;
    end
    output=[error,diffphase,row_shift,col_shift];
else
```

```
    [m,n]=size(buf1ft);
    mlarge=m*2;
    nlarge=n*2;
    CC=zeros(mlarge,nlarge);
    CC(m+1-fix(m/2):m+1+fix((m-1)/2),n+1-fix(n/2):n+1+fix((n-1)/2))
= ...
        fftshift(buf1ft).*conj(fftshift(buf2ft));
    CC = ifft2(ifftshift(CC));
    [max1,loc1] = max(CC);
    [max2,loc2] = max(max1);
    rloc=loc1(loc2);cloc=loc2;
    CCmax=CC(rloc,cloc);
    [m,n] = size(CC);
    md2 = fix(m/2);
    nd2 = fix(n/2);

    if rloc > md2
        row_shift = rloc - m - 1;
    else
        row_shift = rloc - 1;
    end
    if cloc > nd2
        col_shift = cloc - n - 1;
    else
        col_shift = cloc - 1;
    end
    row_shift=row_shift/2;
    col_shift=col_shift/2;

    if usfac > 2,
        row_shift = round(row_shift*usfac)/usfac;
        col_shift = round(col_shift*usfac)/usfac;
        dftshift = fix(ceil(usfac*1.5)/2);
        CC =
conj(dftups(buf2ft.*conj(buf1ft),ceil(usfac*1.5),ceil(usfac*1.5),usf
ac,...
            dftshift-row_shift*usfac,dftshift-
col_shift*usfac))/(md2*nd2*usfac^2);

        [max1,loc1] = max(CC);
        [max2,loc2] = max(max1);
        rloc = loc1(loc2); cloc = loc2;
        CCmax = CC(rloc,cloc);
        rg00 =
dftups(buf1ft.*conj(buf1ft),1,1,usfac)/(md2*nd2*usfac^2);
        rf00 =
dftups(buf2ft.*conj(buf2ft),1,1,usfac)/(md2*nd2*usfac^2);
        rloc = rloc - dftshift - 1;
        cloc = cloc - dftshift - 1;
        row_shift = row_shift + rloc/usfac;
        col_shift = col_shift + cloc/usfac;
```

```
    else
        rg00 = sum(sum( buf1ft.*conj(buf1ft) ))/m/n;
        rf00 = sum(sum( buf2ft.*conj(buf2ft) ))/m/n;
    end
    error = 1.0 - CCmax.*conj(CCmax)/(rg00*rf00);
    error = sqrt(abs(error));
    diffphase=atan2(imag(CCmax),real(CCmax));

    if md2 == 1,
        row_shift = 0;
    end
    if nd2 == 1,
        col_shift = 0;
    end
    output=[error,diffphase,row_shift,col_shift];

end

if (nargout > 1)&&(usfac > 0),
    [nr,nc]=size(buf2ft);
    Nr = ifftshift([-fix(nr/2):ceil(nr/2)-1]);
    Nc = ifftshift([-fix(nc/2):ceil(nc/2)-1]);
    [Nc,Nr] = meshgrid(Nc,Nr);
    Greg = buf2ft.*exp(i*2*pi*(-row_shift*Nr/nr-col_shift*Nc/nc));
    Greg = Greg*exp(i*diffphase);
elseif (nargout > 1)&&(usfac == 0)
    Greg = buf2ft*exp(i*diffphase);
end
return
```

LAMPIRAN E (Lanjutan)

LAMPIRAN F

F. Kode Program Rekontruksi Citra dengan Projection Onto Convex Sets (POCS)

```
function y = POCS1(s,delta_est,factor)

lamda=0.01;
max_iter = 50;
temp = upsample(upsample(s{1}, factor)', factor)';
y = zeros(size(temp));
coord = find(temp);
y(coord) = temp(coord);
Citra_rev=imresize(s{1},factor,'bicubic');
X=fft2(Citra_rev);
X_prev=X;
for i = 2:length(s)
    temp = upsample(upsample(s{i}, factor)', factor)';
    temp = shift(temp, round(delta_est(i, 2)*factor),
round(delta_est(i, 1)*factor)); % Call shift function
    coord = find(temp);
    y(coord) = temp(coord);
end
y_prev=y;
E=[];
iter=1;
blur =[.25 0 1 0 .25;...
        0  1 2 1  0;...
        1  2 4 2  1;...
        0  1 2 1  0;...
       .25 0 1 0 .25];
blur = blur / sum(blur(:));
G=fft2(y);
while iter < max_iter
    X = imfilter(X, blur);
    R=(G-X)/norm(blur);
    temp2=lamda*(imfilter(R,blur));
    X=X_prev+temp2;
    X_prev=X
    delta= norm(y-y_prev)/norm(y);
    E=[E; iter delta];
   iter = iter+1;
   if iter>3
     if abs(E(iter-3,2)-delta) <1e-4
        break
     end
   end
end
G=ifft2(X);
y = uint8(G);
```

LAMPIRAN F (Lanjutan)

BIODATA PENULIS

Penulis yang memiliki nama lengkap Yabunayya Habibi, lahir di Blitar pada tanggal 26 April 1991. Pendidikan formal yang pernah ditempuh yaitu SD Negeri Kalipang 01 Kecamatan Sutojayan-Kabupaten Blitar, SMP Negeri 1 Sutojayan Kecamatan Sutojayan-Kabupaten Blitar dan dilanjutkan ke SMA Darul Ulum 1 Peterongan-Kabupaten Jombang dalam ruang lingkup pondok pesantren. Setelah lulus SMA, penulis melanjutkan studi di jurusan S1 Matematika Universitas Airlangga (UNAIR) Surabaya. Selama kuliah S1 penulis aktif dalam kegiatan organisasi kepanitiaan dan bidang keilmuan khususnya dalam bidang ilmu komputer dan riset operasi. Kajian tentang sistem manajemen basis data dengan MySQL dan aplikasinya juga dipelajari pada kuliah S1 sebagai mata kuliah pilihan. Bahasa Pemrograman yang pernah penulis pelajari adalah C, C++, Visual Basic, Java, dan beberapa pengembangannya dalam program perangkat lunak komputasi seperti Matlab dan Mathematica. Karena ketertarikan dalam bidang tersebut dan pengembangannya pada jenjang S1, penulis mengambil topik skripsi tentang optimasi penjadwalan dengan algoritma metaheuristik.

Penulis melanjutkan studi S2 di jurusan Matematika Institut Teknologi Sepuluh Nopember Surabaya pada tahun 2014 semester genap. Pada perkuliahan di S2 penulis tertarik dengan mata kuliah Analisis dan Pengolahan Citra karena merupakan kajian baru bagi penulis yang menjadi latar belakang penulis untuk mengambil topik Tesis tentang pelacakan objek berbasis citra super-resolusi. Selama penulisan Tesis ini penulis tidak lepas dari beberapa kekurangan sehingga kritik, saran, dan pertanyaan yang berhubungan dengan Tesis ini yang dapat dikirimkan melalui *e-mail* ke habibiexebppt@gmail.com.