LAPORAN PRAKTIKUM PEMROGRAMAN KOMPUTER

Sistem Deteksi Gerakan Menggunakan OpenCV



Nama: Fatih Jawwad Al Mumtaz

NIM : 452024611047

Program Studi : Teknik Informatika

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS DARUSSALAM GONTOR

2024

Daftar Isi

Halaman sampul i

1 Pendahuluan 1

1.1 Latar Belakang 1

1.2 Rumusan Masalah 2

1.3 Tujuan 2

1.4 Manfaat 3

2 Dasar Algoritma dan Pemrograman 4

2.1 Teori Algoritma Pemrosesan Citra 4

2.2 Algoritma Deteksi Gerakan 6

2.2.1 Pengambilan Video 7

2.2.2 Pembacaan Frame 8

2.2.3 Perhitungan Perbedaan Frame 9

2.2.4 Konversi Ke Grayscale 10

2.2.5 Pengaburan Citra 11

2.2.6 Thresholding 12

2.2.7 Dilasi 13

2.2.8 Pencarian Kontur 15

2.2.9 Penggambaran Bounding Box 16

2.2.10 Menampilkan hasil 18

2.3 Struktur Data Yang Digunakan 19

2.3.1 Struktur Data Untuk Citra 19

2.3.2 Struktur Data Untuk Kontur 20

3 Penutup 22

3.1 Kesimpulan 22

4 Daftar Pustaka 24

# Pendahuluan

## Latar Belakang

Deteksi gerakan merupakan salah satu aspek penting dalam bidang pemrosesan citra dan visi komputer. Dengan kemajuan teknologi dan meningkatnya kebutuhan akan sistem keamanan, otomatisasi, dan interaksi manusia-komputer, deteksi gerakan telah menjadi topik yang menarik perhatian banyak peneliti dan praktisi. Sistem deteksi gerakan dapat digunakan dalam berbagai aplikasi, mulai dari pengawasan keamanan, analisis perilaku, hingga interaksi dalam augmented reality dan robotik

Dalam konteks keamanan, deteksi gerakan memungkinkan pemantauan area tertentu secara real-time, memberikan peringatan dini terhadap potensi ancaman. Misalnya, kamera pengawas yang dilengkapi dengan algoritma deteksi gerakan dapat secara otomatis mendeteksi aktivitas mencurigakan dan mengirimkan notifikasi kepada pemilik atau petugas keamanan. Hal ini tidak hanya meningkatkan efisiensi sistem keamanan, tetapi juga mengurangi kebutuhan akan pengawasan manual yang intensif.

## Rumusan Masalah

1. Bagaimana cara mendeteksi gerakan secara akurat dalam video menggunakan algoritma pemrosesan citra?
2. Apa langkah-langkah utama dalam proses deteksi gerakan yang harus diimplementasikan?
3. Bagaimana pengaruh parameter pemrosesan citra, seperti thresholding dan pengaburan, terhadap hasil deteksi?
4. Apa tantangan yang dihadapi dalam deteksi gerakan, seperti noise dan perubahan latar belakang?
5. Apa saja aplikasi praktis dari teknologi deteksi gerakan dalam kehidupan sehari-hari?

## Tujuan.

Tujuan dari makalah ini adalah untuk memberikan pemahaman yang mendalam tentang implementasi deteksi gerakan menggunakan OpenCV, serta menganalisis algoritma dan struktur data yang terlibat dalam proses tersebut. Dengan menjelaskan langkah-langkah yang diperlukan dalam deteksi gerakan, mulai dari pengambilan video hingga analisis kontur, makalah ini bertujuan untuk mengedukasi pembaca mengenai teknik-teknik pemrosesan citra yang digunakan untuk mengidentifikasi pergerakan dalam video. Selain itu, makalah ini juga akan membahas tantangan yang dihadapi dalam deteksi gerakan, seperti variabilitas pencahayaan dan kompleksitas latar belakang, serta bagaimana solusi dapat diterapkan untuk meningkatkan akurasi deteksi.

Selain itu, makalah ini bertujuan untuk mengeksplorasi aplikasi praktis dari teknologi deteksi gerakan dalam berbagai bidang, termasuk keamanan, analisis perilaku, dan interaksi manusia-komputer. Dengan memberikan contoh nyata dan potensi penggunaan teknologi ini, diharapkan pembaca dapat memahami relevansi dan manfaat dari deteksi gerakan dalam kehidupan sehari-hari. Melalui analisis yang komprehensif, makalah ini diharapkan dapat memberikan kontribusi pada pengembangan lebih lanjut dalam bidang visi komputer dan pemrosesan citra, serta mendorong penelitian dan inovasi di masa depan.

## Manfaat.

Manfaat Akademis:

1. Peningkatan Pemahaman Teoritis

Proyek ini memberikan pemahaman yang lebih mendalam tentang konsep dasar pemrosesan citra dan visi komputer. Peserta dapat mempelajari berbagai algoritma dan teknik yang digunakan dalam deteksi gerakan, yang merupakan fondasi penting dalam bidang ini.

1. Pengembangan Keterampilan Teknis

Melalui implementasi praktis, peserta proyek dapat mengembangkan keterampilan teknis yang diperlukan untuk menerapkan teknologi deteksi gerakan. Ini mencakup pemrograman dengan OpenCV, analisis citra, dan pemecahan masalah yang berkaitan dengan pengolahan data visual.

1. Dasar untuk Penelitian Lanjutan

Proyek ini dapat menjadi dasar untuk penelitian lebih lanjut di bidang visi komputer. Dengan memahami teknik deteksi gerakan, peserta dapat mengeksplorasi topik-topik lanjutan, seperti pembelajaran mesin dan kecerdasan buatan, yang dapat meningkatkan akurasi dan efisiensi sistem deteksi.

Manfaat Praktis :

1. Peningkatan Keamanan

Sistem deteksi gerakan dapat digunakan untuk meningkatkan keamanan di area publik dan pribadi. Dengan kemampuan untuk mendeteksi aktivitas mencurigakan secara real-time, teknologi ini dapat memberikan peringatan dini dan membantu mencegah tindakan kriminal.

1. Aplikasi dalam Analisis Perilaku

Proyek ini juga memiliki aplikasi dalam analisis perilaku, baik manusia maupun hewan. Dengan menggunakan teknologi deteksi gerakan, peneliti dapat mengamati dan menganalisis pola perilaku, yang dapat berguna dalam berbagai studi ilmiah.

1. Interaksi Manusia-Komputer yang Lebih Baik

Teknologi deteksi gerakan dapat diterapkan dalam interaksi manusia-komputer, memungkinkan pengguna untuk berinteraksi dengan perangkat hanya melalui gerakan. Ini membuka peluang untuk menciptakan pengalaman pengguna yang lebih intuitif dan imersif, terutama dalam aplikasi augmented reality dan permainan video.

# Dasar Algoritma dan Pemrograman

## Teori Algoritma Pemrosesan Citra.

Pemrosesan gambar adalah cabang dari ilmu komputer dan teknik yang berfokus pada analisis dan manipulasi gambar digital. Tujuan utama dari pemrosesan gambar adalah untuk meningkatkan kualitas gambar, mengekstrak informasi yang relevan, dan mempersiapkan gambar untuk analisis lebih lanjut. Proses ini melibatkan berbagai teknik, mulai dari pengolahan dasar seperti pengubahan kontras dan pencahayaan, hingga metode yang lebih kompleks seperti segmentasi, deteksi tepi, dan pengenalan pola. Dengan kemajuan teknologi, pemrosesan gambar telah menjadi komponen penting dalam berbagai aplikasi, termasuk pengawasan, medis, dan interaksi manusia-komputer. Cara pemrosesan gambar mengisolasi bidang warna terpisah dari suatu gambar dan juga menerapkan pendekatan pemrosesan sinyal standar pada bidang tersebut.[[1]](#footnote-1)

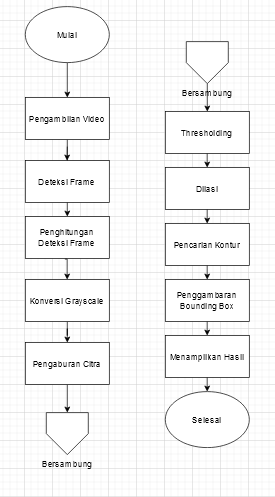
Salah satu konsep dasar dalam pemrosesan gambar adalah representasi gambar itu sendiri. Gambar digital biasanya direpresentasikan sebagai matriks dua dimensi, di mana setiap elemen matriks (pixel) mewakili nilai intensitas cahaya pada titik tertentu. Pemrosesan gambar dapat dilakukan pada level pixel, di mana operasi matematis diterapkan untuk mengubah nilai-nilai pixel guna mencapai tujuan tertentu. Misalnya, dalam pengolahan citra grayscale, setiap pixel diubah menjadi nilai tunggal yang mewakili intensitas cahaya, sedangkan dalam citra berwarna, setiap pixel biasanya terdiri dari tiga komponen warna (merah, hijau, dan biru) yang digabungkan untuk membentuk warna akhir.

Teknik pemrosesan gambar dapat dibagi menjadi dua kategori utama: pemrosesan gambar spasial dan pemrosesan gambar frekuensi. Pemrosesan gambar spasial berfokus pada manipulasi langsung pixel dalam domain gambar, seperti pengaburan, penajaman, dan pengubahan ukuran. Di sisi lain, pemrosesan gambar frekuensi melibatkan transformasi gambar ke domain frekuensi menggunakan teknik seperti Transformasi Fourier, yang memungkinkan analisis komponen frekuensi dalam gambar. Dengan memahami kedua pendekatan ini, para peneliti dan praktisi dapat mengembangkan algoritma yang lebih efektif untuk berbagai aplikasi, mulai dari deteksi objek hingga pengenalan wajah, yang semakin penting dalam era digital saat ini.

## Algoritma Deteksi Gerakan.

Algoritma deteksi gerakan umumnya berfokus pada analisis perbedaan antara frame video yang diambil secara berurutan untuk mengidentifikasi perubahan yang menunjukkan adanya pergerakan. Proses ini dimulai dengan pengambilan dua frame berturut-turut dari video.

Selanjutnya, algoritma menghitung perbedaan antara kedua frame tersebut menggunakan metode seperti pengurangan absolut (absolute difference). Hasil dari perhitungan ini kemudian diproses lebih lanjut dengan mengonversi citra menjadi grayscale, mengaburkan citra untuk mengurangi noise, dan menerapkan thresholding untuk menghasilkan citra biner. Citra biner ini menyoroti area yang mengalami perubahan signifikan, yang dianggap sebagai gerakan.



Setelah mendapatkan citra biner, langkah selanjutnya adalah melakukan dilasi untuk memperkuat area yang terdeteksi sebagai gerakan, diikuti dengan pencarian kontur untuk mengidentifikasi bentuk dan ukuran objek yang bergerak. Dengan menggunakan fungsi seperti cv.findContours(), algoritma dapat menemukan kontur dari objek yang terdeteksi dan menggambar bounding box di sekitar objek tersebut.

Selain itu, algoritma dapat dilengkapi dengan logika tambahan untuk mengabaikan gerakan kecil yang tidak signifikan, seperti perubahan akibat noise atau pergerakan latar belakang. Dengan pendekatan ini, algoritma deteksi gerakan dapat memberikan informasi yang berguna untuk berbagai aplikasi, mulai dari sistem keamanan hingga analisis perilaku.

## 2.2.1 Teori Pengambilan Video

Pengambilan video adalah proses merekam gambar bergerak dalam bentuk rangkaian frame yang diambil secara berurutan pada interval waktu tertentu. Setiap frame dalam video adalah gambar statis yang, ketika diputar dengan kecepatan tertentu, menciptakan ilusi gerakan.

Proses ini melibatkan penggunaan perangkat keras seperti kamera, yang dapat menangkap cahaya dan mengubahnya menjadi sinyal digital. Kecepatan pengambilan, atau frame rate, biasanya diukur dalam frame per detik (fps) dan berpengaruh besar terhadap kualitas video. Frame rate yang umum digunakan dalam industri film adalah 24 fps, sementara video televisi sering menggunakan 30 fps atau 60 fps untuk memberikan gerakan yang lebih halus.

Dalam konteks pemrosesan citra dan analisis video, pengambilan video tidak hanya sekadar merekam gambar, tetapi juga melibatkan teknik untuk memastikan bahwa data yang dihasilkan berkualitas tinggi dan relevan untuk analisis lebih lanjut. Faktor-faktor seperti resolusi, pencahayaan, dan stabilitas kamera sangat penting dalam menentukan kualitas video yang diambil. Selain itu, pengambilan video dapat dilakukan dalam berbagai format dan kompresi untuk mengoptimalkan penyimpanan dan transmisi data.

Dengan kemajuan teknologi, pengambilan video kini juga dapat dilakukan menggunakan perangkat mobile dan drone, yang semakin memperluas aplikasi dan penggunaan video dalam berbagai bidang, termasuk keamanan, hiburan, dan penelitian ilmiah.

## 2.2.2 Pembacaan Frame.

Pembacaan frame dalam konteks pemrosesan video adalah proses ekstraksi gambar statis dari aliran video yang direkam. Setiap frame diambil pada interval waktu tertentu, yang dikenal sebagai frame rate, yang diukur dalam frame per detik (fps). Frame rate yang umum digunakan dalam video adalah 24 fps untuk film, 30 fps untuk televisi, dan 60 fps untuk video berkualitas tinggi. Pembacaan frame yang tepat sangat penting untuk analisis video, karena kecepatan pengambilan frame mempengaruhi detail gerakan yang dapat ditangkap. Jika frame rate terlalu rendah, gerakan cepat mungkin tidak terdeteksi dengan baik, sedangkan frame rate yang terlalu tinggi dapat menghasilkan data yang berlebihan dan memerlukan lebih banyak sumber daya untuk pemrosesan.

Secara matematis, jika adalah waktu dalam detik dan adalah frame rate dalam fps, maka waktu antara dua frame berturut-turut dapat dihitung dengan rumus:

Di mana adalah interval waktu antara dua frame. Misalnya, untuk frame rate 30 fps, interval waktu antara frame adalah:

Dengan menggunakan rumus ini, sistem dapat menentukan kapan harus mengambil frame berikutnya untuk memastikan bahwa gerakan dalam video direkam dengan akurat. Selain itu, pembacaan frame juga melibatkan pengolahan citra untuk memastikan bahwa setiap frame yang diambil memiliki kualitas yang baik, yang dapat melibatkan teknik seperti pengaturan pencahayaan dan fokus kamera.

## 2.2.3 Penghitungan Perbedaan Frame

Metode Frame differencemerupakan salah satu metode yang digunakan untuk mendeteksi objek yang bergerak dalam sebuah video. Metode ini didasarkan pada perbedaan antara dua Frame berturut-turut dalam video

Untuk menemukan perbedaan atau perubahan yang terjadi[[2]](#footnote-2). Perhitungan perbedaan antara dua frame dalam deteksi gerakan adalah langkah kunci untuk mengidentifikasi perubahan yang menunjukkan adanya pergerakan. Misalkan kita memiliki dua frame video, dan yang masing-masing merupakan citra dua dimensi dengan ukuran Setiap pixel dalam frame memiliki nilai intensitas yang dapat dinyatakan sebagai (x, y) dan (x, y) , di mana (x, y) adalah koordinat pixel. Untuk menghitung perbedaan antara dua frame, kita menggunakan rumus:

Rumus ini memberikan nilai perbedaan pada setiap pixel, yang menunjukkan seberapa besar perubahan intensitas cahaya antara frame pertama dan kedua.

Algoritma frame differences merupakan salah satu algoritma yang digunakan dalam mendeteksi gerak melalui sebuah citra[[3]](#footnote-3). Setelah menghitung perbedaan untuk setiap pixel, kita dapat memperoleh gambaran keseluruhan tentang seberapa besar perubahan yang terjadi antara kedua frame dengan menjumlahkan semua nilai perbedaan di seluruh pixel. Total perbedaan dapat dihitung dengan rumus:

Di mana adalah total perbedaan antara kedua frame, dan *m* serta *n* adalah dimensi dari frame.

Jika nilai melebihi ambang batas tertentu, maka dapat disimpulkan bahwa ada pergerakan yang signifikan antara kedua frame tersebut, yang menjadi indikator penting dalam proses deteksi gerakan.

## 2.2.4 Konversi Ke *Grayscale*

Konversi citra berwarna menjadi grayscale adalah langkah penting dalam pemrosesan citra, karena mengurangi kompleksitas data dengan mengubah citra dari format berwarna menjadi satu saluran intensitas.

Proses ini dilakukan dengan menggabungkan komponen warna merah, hijau, dan biru (RGB) menjadi satu nilai intensitas. Salah satu metode yang umum digunakan untuk konversi ini adalah dengan menggunakan rumus yang mempertimbangkan sensitivitas mata manusia terhadap berbagai warna. Rumus yang sering digunakan adalah:

Di mana *Y* adalah nilai intensitas grayscale, dan *R* , *G* , dan *B* adalah nilai intensitas dari komponen merah, hijau, dan biru pada pixel tertentu. Koefisien dalam rumus ini mencerminkan kontribusi relatif dari masing-masing warna terhadap persepsi manusia, di mana warna hijau memiliki pengaruh terbesar, diikuti oleh merah dan biru.

Setelah konversi dilakukan, setiap pixel dalam citra berwarna akan digantikan oleh nilai grayscale yang dihasilkan dari rumus tersebut. Proses ini tidak hanya mengurangi jumlah data yang perlu diproses, tetapi juga memfokuskan analisis pada intensitas cahaya, yang sangat berguna dalam berbagai aplikasi pemrosesan citra, seperti deteksi tepi dan segmentasi.

Dengan menggunakan citra grayscale, algoritma pemrosesan citra dapat bekerja lebih efisien dan efektif, karena mengurangi kompleksitas komputasi yang diperlukan untuk memproses citra berwarna.

## 2.2.5 Pengaburan Citra

Pengaburan citra adalah teknik pemrosesan citra yang digunakan untuk mengurangi noise dan detail halus dalam gambar, sehingga menghasilkan citra yang lebih halus dan lebih mudah untuk dianalisis. Salah satu metode pengaburan yang paling umum adalah Gaussian blur, yang menggunakan fungsi distribusi Gaussian untuk memberikan bobot yang lebih besar pada pixel yang lebih dekat dengan pusat kernel. Proses ini dilakukan dengan mengalikan setiap pixel dalam citra dengan nilai dari kernel Gaussian, yang didefinisikan oleh rumus:

Di mana adalah nilai dari fungsi Gaussian pada titik dan adalah deviasi standar yang mengontrol tingkat pengaburan. Semakin besar nilai , semakin besar area pengaruh dari pixel di sekitar pusat, yang menghasilkan efek pengaburan yang lebih kuat.

Setelah kernel Gaussian dihitung, pengaburan citra dilakukan dengan cara konvolusi antara citra asli *( I )* dan kernel Gaussian *( G ).* Proses konvolusi dapat dinyatakan dengan rumus:

Di mana adalah nilai pixel pada citra yang telah di-blur, dan  *k* adalah ukuran kernel. Dengan menggunakan teknik pengaburan ini, citra yang dihasilkan akan memiliki detail yang lebih sedikit, sehingga membantu dalam mengurangi noise dan meningkatkan kinerja algoritma pemrosesan citra selanjutnya, seperti deteksi tepi dan segmentasi.

## 2.2.6 Penjabaran Thresholding

Thresholding adalah teknik pemrosesan citra yang digunakan untuk mengubah citra grayscale menjadi citra biner dengan cara menetapkan ambang batas tertentu. Proses ini sangat berguna dalam segmentasi citra, di mana tujuan utamanya adalah memisahkan objek dari latar belakang. Operasi pengambangan (thresholding) digunakan untuk mengubah citra dengan format skala keabuan ke citra biner, yang hanya memiliki 2 buah nilai (0 atau 1).[[4]](#footnote-4)

Dalam thresholding, setiap pixel dalam citra grayscale dibandingkan dengan nilai ambang . Jika nilai pixel lebih besar dari , pixel tersebut akan diubah menjadi 1 (putih), yang menunjukkan keberadaan objek, sedangkan jika nilai pixel lebih kecil atau sama dengan , pixel tersebut akan diubah menjadi 0 (hitam), yang menunjukkan latar belakang. Secara matematis, proses ini dapat dinyatakan dengan rumus:

Di mana adalah nilai biner pada posisi setelah thresholding, dan ( adalah nilai intensitas pixel pada citra grayscale.

Thresholding dapat dilakukan dengan berbagai metode, termasuk thresholding global, di mana satu nilai ambang digunakan untuk seluruh citra, dan thresholding adaptif, di mana nilai ambang ditentukan berdasarkan karakteristik lokal dari citra. Dalam thresholding adaptif, nilai ambang *T* dapat dihitung berdasarkan rata-rata intensitas pixel dalam jendela lokal di sekitar pixel yang sedang dianalisis. Rumus untuk menghitung nilai ambang adaptif dapat dinyatakan sebagai:

Di mana *N* adalah jumlah pixel dalam jendela *W* , dan adalah koordinat pixel dalam jendela tersebut. Dengan cara ini, thresholding dapat disesuaikan untuk kondisi pencahayaan yang bervariasi dalam citra, sehingga meningkatkan akurasi segmentasi objek dan menghasilkan hasil yang lebih baik dalam analisis citra.

## 2.2.7 Penjabaran Dilasi

Dilasi adalah teknik pemrosesan citra yang digunakan untuk memperluas area objek dalam citra biner. Proses ini melibatkan penambahan pixel ke batas objek yang terdeteksi, sehingga membuat objek terlihat lebih besar dan lebih jelas.

Dilasi sering digunakan dalam berbagai aplikasi, seperti pengolahan citra medis, pengenalan pola, dan deteksi objek, di mana memperkuat fitur objek dapat membantu dalam analisis lebih lanjut.

Dengan memperluas objek, dilasi juga dapat membantu mengisi celah atau lubang kecil dalam objek, yang dapat meningkatkan akurasi dalam penghitungan atau pengenalan objek.

Secara matematis, dilasi dapat dinyatakan melalui operasi konvolusi antara citra biner dan struktur elemen. Struktur elemen adalah matriks kecil yang menentukan bentuk dan ukuran area yang akan diperluas. Jika *( B )* adalah citra biner dan *( S )* adalah struktur elemen, maka dilasi *( D )* dari citra *( B )* dapat dihitung dengan rumus:

Di mana adalah nilai pixel pada posisi setelah dilasi, dan *(i, j)* adalah koordinat relatif dari struktur elemen *S* . Dengan rumus ini, setiap pixel dalam citra hasil dilasi akan memiliki nilai maksimum dari pixel yang terpengaruh oleh struktur elemen, sehingga memperluas area objek.

Dilasi dapat diterapkan berulang kali untuk mencapai efek yang diinginkan, dan kombinasi dengan operasi lain seperti erosi dapat digunakan untuk mencapai hasil yang lebih kompleks.

Misalnya, setelah dilasi, erosi dapat diterapkan untuk mengurangi ukuran objek kembali, tetapi dengan mempertahankan fitur yang lebih besar. Proses ini sering disebut sebagai "morfologi" dalam pemrosesan citra, di mana berbagai operasi morfologi seperti dilasi dan erosi digunakan untuk memanipulasi bentuk dan struktur objek dalam citra. Dengan menggunakan teknik ini, analisis citra dapat dilakukan dengan lebih efektif, terutama dalam konteks di mana detail objek sangat penting.

## 2.2.8 Pencarian Kontur

Pencarian kontur adalah proses penting dalam pemrosesan citra yang digunakan untuk mendeteksi dan mengidentifikasi batas atau tepi objek dalam citra. Kontur adalah garis yang menghubungkan titik-titik dengan nilai intensitas yang sama, dan pencarian kontur memungkinkan kita untuk menemukan bentuk dan ukuran objek yang ada dalam citra.

Proses ini sering digunakan dalam berbagai aplikasi, seperti pengenalan objek, analisis bentuk, dan pengolahan citra medis. Dengan mendeteksi kontur, kita dapat memperoleh informasi yang lebih mendalam tentang struktur objek, yang dapat digunakan untuk analisis lebih lanjut atau pengambilan keputusan.

Secara matematis, pencarian kontur dapat dilakukan dengan menggunakan algoritma seperti Moore-Neighbor Tracing atau Suzuki-Abe. Salah satu pendekatan yang umum digunakan adalah dengan memanfaatkan citra biner yang dihasilkan dari proses thresholding. Dalam konteks ini, kita dapat menggunakan fungsi kontur yang mendeteksi batas antara area dengan nilai 1 (objek) dan 0 (latar belakang). Misalkan adalah himpunan titik yang membentuk kontur, maka kontur dapat dinyatakan sebagai:

Di mana adalah citra biner yang menunjukkan objek. Dengan rumus ini, kita dapat mengidentifikasi titik-titik yang membentuk kontur dengan memeriksa pixel yang memiliki nilai 1 dan dikelilingi oleh pixel dengan nilai 0.

Setelah kontur terdeteksi, kita dapat menghitung berbagai parameter geometris, seperti panjang kontur, luas area yang dibatasi oleh kontur, dan momen kontur. Panjang kontur dapat dihitung dengan menjumlahkan jarak antara titik-titik kontur yang berurutan. Jika adalah kontur yang terdiri dari titik, maka panjang kontur dapat dinyatakan sebagai:

Di mana adalah jarak antara dua titik kontur berturut-turut, yang dapat dihitung menggunakan rumus jarak Euclidean:

Dengan menggunakan teknik pencarian kontur dan rumus matematis ini, kita dapat memperoleh informasi yang berharga tentang bentuk dan ukuran objek dalam citra, yang sangat berguna dalam berbagai aplikasi pemrosesan citra dan analisis visual.

## 2.2.9 Penggambaran Bounding Box

Bounding box adalah kotak persegi panjang yang digunakan untuk membatasi atau mengelilingi objek yang terdeteksi dalam citra. Dalam konteks deteksi gerakan dan pemrosesan citra, bounding box memberikan cara yang efisien untuk menandai lokasi dan ukuran objek yang terdeteksi, sehingga memudahkan analisis lebih lanjut.

Ketika kontur objek terdeteksi, bounding box dapat dihitung dengan menemukan koordinat minimum dan maksimum dari kontur tersebut. Dengan cara ini, kita dapat menggambar kotak di sekitar objek yang bergerak, yang membantu dalam visualisasi dan pengenalan objek.

Proses untuk menghitung bounding box dimulai setelah kontur objek terdeteksi. Untuk menghitung bounding box, kita perlu menemukan nilai minimum dan maksimum dari koordinat ( x ) dan ( y ) dari semua titik kontur. Koordinat bounding box dapat dinyatakan sebagai:

Di mana adalah sudut kiri atas dari bounding box, dan adalah sudut kanan bawah dari bounding box.

Rumus Matematis untuk Bounding Box

Setelah mendapatkan koordinat minimum dan maksimum, kita dapat menggambar bounding box dengan menggunakan dua titik tersebut. Panjang dan lebar dari bounding box dapat dihitung sebagai berikut:

Dengan menggunakan informasi ini, kita dapat menggambar bounding box di sekitar objek yang terdeteksi dalam citra. Bounding box tidak hanya memberikan informasi tentang lokasi objek, tetapi juga dapat digunakan untuk menghitung area objek dan melakukan analisis lebih lanjut, seperti pelacakan objek dari frame ke frame dalam video. Dengan demikian, bounding box adalah alat yang sangat berguna dalam pemrosesan citra dan visi komputer.

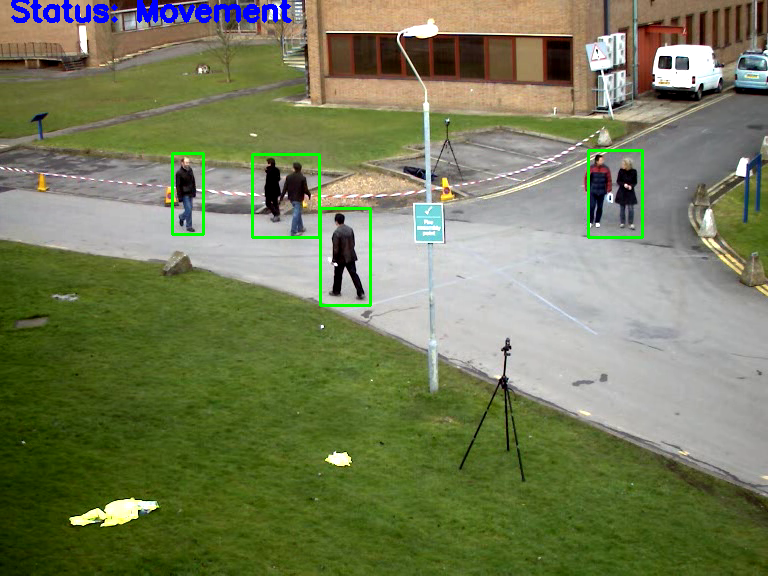
## 2.2.10 Menampilkan Hasil

**Deteksi gerakan dalam video adalah proses yang digunakan untuk mengidentifikasi dan menandai pergerakan objek yang terjadi dalam rangkaian frame video. Teknik ini sangat penting dalam berbagai aplikasi, seperti sistem keamanan, analisis perilaku, dan pengenalan objek.**

# Setelah pergerakan terdeteksi, program akan menandai objek yang bergerak dengan menggunakan teknik seperti bounding box. Bounding box adalah kotak persegi panjang yang digambar di sekitar objek yang terdeteksi, memberikan visualisasi yang jelas tentang lokasi dan ukuran objek tersebut.

# Dengan cara ini, pengguna dapat dengan mudah melihat pergerakan yang terjadi dalam video. Selain itu, informasi tambahan seperti arah dan kecepatan pergerakan objek juga dapat dihitung dan ditampilkan, memberikan konteks yang lebih dalam tentang aktivitas yang sedang berlangsung.

# Deteksi gerakan tidak hanya berguna untuk memantau aktivitas dalam video, tetapi juga dapat digunakan untuk mengaktifkan tindakan tertentu berdasarkan pergerakan yang terdeteksi. Misalnya, dalam sistem keamanan, deteksi gerakan dapat memicu alarm atau merekam video ketika objek bergerak di area yang diawasi. Dengan demikian, teknologi deteksi gerakan memainkan peran penting dalam meningkatkan keamanan, efisiensi, dan analisis dalam berbagai bidang, menjadikannya alat yang sangat berharga dalam pemrosesan video dan visi komputer.



### Looping

Looping adalah struktur kontrol yang memungkinkan eksekusi berulang dari sekelompok instruksi selama kondisi tertentu terpenuhi. Ada beberapa jenis looping, seperti for dan while.

Contoh penggunaan looping adalah menghitung jumlah angka dari 1 hingga 10. Dalam algoritma ini, kita dapat menggunakan loop for untuk iterasi. Pertama, kita inisialisasi variabel total dengan nilai 0. Kemudian, kita menggunakan loop for untuk iterasi dari 1 hingga 10. Di dalam loop, kita menambahkan setiap angka ke variabel total. Setelah loop selesai, kita dapat mencetak hasil total.

Looping sangat berguna dalam situasi di mana kita perlu melakukan operasi yang sama berulang kali, seperti memproses elemen dalam daftar atau array. Misalnya, jika kita memiliki daftar nama dan ingin mencetak setiap nama, kita dapat menggunakan loop untuk iterasi melalui daftar tersebut.

Looping juga dapat digunakan untuk mengulangi proses hingga kondisi tertentu terpenuhi, seperti dalam loop while. Dalam hal ini, kita dapat memeriksa kondisi di awal setiap iterasi dan menghentikan loop jika kondisi tidak lagi terpenuhi.

Secara keseluruhan, looping adalah alat yang sangat kuat dalam pemrograman yang memungkinkan kita untuk menghemat waktu dan usaha dengan mengotomatiskan proses yang berulang.

### If-Else

Struktur kontrol if-else adalah salah satu cara untuk membuat keputusan dalam pemrograman. Dengan menggunakan if-else, kita dapat mengeksekusi blok kode tertentu berdasarkan kondisi yang diberikan.

Misalnya, kita ingin menentukan apakah seorang siswa lulus atau tidak berdasarkan nilai ujian. Pertama, kita dapat meminta input nilai siswa. Kemudian, kita menggunakan pernyataan if untuk memeriksa apakah nilai tersebut lebih besar atau sama dengan 60. Jika kondisi ini benar, kita mencetak "Selamat, Anda lulus!" Jika tidak, kita menggunakan else untuk mencetak "Maaf, Anda tidak lulus."

Struktur if-else juga dapat diperluas dengan menggunakan elif (else if) untuk memeriksa beberapa kondisi. Misalnya, kita dapat menambahkan kategori nilai, seperti "Baik" untuk nilai di atas 80 dan "Cukup" untuk nilai di antara 60 dan 79. Dengan cara ini, kita dapat memberikan umpan balik yang lebih spesifik kepada siswa berdasarkan kinerja mereka.

Penggunaan if-else sangat penting dalam pemrograman karena memungkinkan kita untuk membuat keputusan berdasarkan input atau kondisi tertentu. Ini membantu dalam mengontrol alur program dan memberikan respons yang sesuai terhadap berbagai situasi.

### Try-Except

Struktur kontrol try-except digunakan untuk menangani pengecualian atau kesalahan yang mungkin terjadi selama eksekusi program. Dengan menggunakan try-except, kita dapat mencegah program dari crash dan memberikan penanganan kesalahan yang lebih baik.

Misalnya, kita ingin meminta pengguna untuk memasukkan angka dan kemudian membagi angka 100 dengan angka yang dimasukkan. Kita dapat menggunakan blok try untuk mencoba melakukan pembagian. Jika pengguna memasukkan 0, ini akan menyebabkan kesalahan pembagian dengan nol. Dalam hal ini, kita dapat menangkap kesalahan tersebut dengan blok except dan mencetak pesan yang sesuai, seperti "Kesalahan: Tidak dapat membagi dengan nol."

Struktur try-except sangat berguna dalam situasi di mana kita tidak dapat memprediksi semua kemungkinan kesalahan yang mungkin terjadi. Misalnya, saat membaca file, file mungkin tidak ada atau tidak dapat diakses. Dengan menggunakan try-except, kita dapat menangani situasi ini dengan cara yang lebih elegan, memberikan umpan balik kepada pengguna tanpa menghentikan program secara tiba-tiba.

Secara keseluruhan, try-except adalah alat penting dalam pemrograman yang membantu kita menangani kesalahan dengan cara yang terstruktur dan aman, meningkatkan keandalan dan pengalaman pengguna dari aplikasi yang kita buat.

## 2.3 Struktur Data Yang Digunakan

## 2.3.1 Struktur Data untuk Citra

Dalam pemrosesan citra, struktur data yang digunakan untuk merepresentasikan citra sangat penting karena mempengaruhi cara citra diolah dan dianalisis. Salah satu struktur data yang paling umum digunakan adalah matriks dua dimensi, di mana setiap elemen dalam matriks mewakili nilai intensitas pixel dari citra.

Untuk citra grayscale, setiap elemen dalam matriks adalah nilai tunggal yang menunjukkan kecerahan pixel, biasanya dalam rentang 0 hingga 255, di mana 0 mewakili warna hitam dan 255 mewakili warna putih. Dengan menggunakan matriks, operasi matematis seperti pengurangan, penjumlahan, dan konvolusi dapat dilakukan dengan mudah untuk berbagai teknik pemrosesan citra.

Untuk citra berwarna, struktur data yang lebih kompleks diperlukan. Citra berwarna biasanya direpresentasikan dalam format RGB (Red, Green, Blue), di mana setiap pixel terdiri dari tiga komponen warna. Dalam hal ini, citra dapat direpresentasikan sebagai tiga matriks dua dimensi, masing-masing untuk komponen merah, hijau, dan biru.

Setiap pixel dalam citra berwarna dapat diakses dengan menggunakan indeks dua dimensi untuk posisi pixel dan tiga indeks tambahan untuk komponen warna. Dengan cara ini, pemrosesan citra berwarna dapat dilakukan dengan mempertimbangkan interaksi antara komponen warna, yang sangat penting dalam aplikasi seperti pengenalan objek dan pengolahan gambar.

Selain matriks, struktur data lain yang sering digunakan dalam pemrosesan citra adalah array multidimensi dan struktur data berbasis objek. Array multidimensi memungkinkan representasi citra dengan lebih dari dua dimensi, yang berguna untuk citra dengan lebih banyak saluran warna atau untuk citra volumetrik.

Di sisi lain, struktur data berbasis objek, seperti kelas dalam pemrograman berorientasi objek, dapat digunakan untuk menyimpan informasi tambahan tentang citra, seperti metadata, ukuran, dan format. Dengan menggunakan struktur data yang tepat, pemrosesan citra dapat dilakukan dengan lebih efisien dan terorganisir, memungkinkan pengembangan aplikasi yang lebih kompleks dan canggih dalam bidang visi komputer.

## 2.3.2 Struktur Data dalam Program

### String

String adalah tipe data yang digunakan untuk menyimpan urutan karakter. Dalam banyak bahasa pemrograman, string dikelilingi oleh tanda kutip tunggal (' ') atau tanda kutip ganda (" "). String dapat berisi huruf, angka, simbol, dan spasi. Misalnya, "Hello, World!" adalah sebuah string yang terdiri dari huruf dan tanda baca.

Salah satu fitur utama dari string adalah kemampuannya untuk melakukan berbagai operasi. Kita dapat menggabungkan (concatenate) dua string menggunakan operator +, seperti "Hello" + " World" yang menghasilkan "Hello World". Selain itu, kita dapat mengakses karakter tertentu dalam string menggunakan indeks, di mana indeks pertama biasanya dimulai dari 0. Misalnya, dalam string "Python", karakter pertama adalah 'P' (indeks 0).

String juga memiliki banyak metode bawaan yang memungkinkan kita untuk memanipulasi dan memformat teks. Contohnya, metode .upper() mengubah semua karakter dalam string menjadi huruf kapital, sedangkan .lower() mengubahnya menjadi huruf kecil. Kita juga dapat menggunakan metode .split() untuk memecah string menjadi daftar berdasarkan pemisah tertentu, seperti spasi atau koma.

Penggunaan string sangat luas dalam pemrograman, mulai dari menyimpan nama pengguna, pesan, hingga data yang lebih kompleks seperti JSON. String juga sering digunakan dalam pengolahan teks, pengembangan web, dan aplikasi berbasis data. Dengan kemampuannya untuk menyimpan dan memanipulasi teks, string menjadi salah satu struktur data yang paling penting dan sering digunakan dalam pemrograman.

### Integer

Integer adalah tipe data yang digunakan untuk menyimpan bilangan bulat, baik positif maupun negatif, tanpa desimal. Dalam banyak bahasa pemrograman, integer dapat digunakan untuk melakukan berbagai operasi aritmatika, seperti penjumlahan, pengurangan, perkalian, dan pembagian. Contoh integer termasuk -5, 0, dan 42.

Salah satu keunggulan integer adalah kemampuannya untuk digunakan dalam perhitungan matematis. Misalnya, kita dapat menyimpan nilai dalam variabel integer dan melakukan operasi seperti a + b, di mana a dan b adalah integer. Hasil dari operasi ini juga akan menjadi integer. Integer juga dapat digunakan dalam pengulangan (looping) dan pengkondisian (if-else) untuk mengontrol alur program.

Integer memiliki batasan tertentu tergantung pada bahasa pemrograman dan arsitektur sistem. Misalnya, dalam banyak bahasa, integer 32-bit dapat menyimpan nilai dari -2.147.483.648 hingga 2.147.483.647. Jika kita mencoba menyimpan nilai di luar rentang ini, kita mungkin mengalami overflow, yang dapat menyebabkan kesalahan atau perilaku yang tidak terduga.

Penggunaan integer sangat umum dalam pemrograman, terutama dalam aplikasi yang memerlukan perhitungan matematis, penghitungan jumlah, atau pengindeksan. Integer juga sering digunakan dalam algoritma dan struktur data, seperti dalam penghitungan elemen dalam array atau dalam pengulangan untuk iterasi. Dengan kemampuannya untuk menyimpan dan memanipulasi bilangan bulat, integer menjadi salah satu struktur data dasar yang penting dalam pemrograman.

### Array

Array adalah struktur data yang digunakan untuk menyimpan sekumpulan elemen yang memiliki tipe data yang sama. Elemen dalam array diakses menggunakan indeks, di mana indeks pertama biasanya dimulai dari 0. Array memungkinkan kita untuk menyimpan dan mengelola data secara efisien, terutama ketika kita perlu bekerja dengan koleksi data yang terorganisir.

Misalnya, kita dapat membuat array untuk menyimpan daftar angka, seperti [1, 2, 3, 4, 5]. Dalam contoh ini, kita memiliki array yang berisi lima elemen integer. Kita dapat mengakses elemen tertentu dengan menggunakan indeks, seperti array[0] yang akan mengembalikan nilai 1. Array juga memungkinkan kita untuk melakukan operasi seperti iterasi, di mana kita dapat menggunakan loop untuk mengakses dan memanipulasi setiap elemen dalam array.

Salah satu keuntungan utama dari array adalah kemampuannya untuk menyimpan data dalam urutan yang teratur, yang memudahkan pengelolaan dan pencarian data. Namun, array memiliki batasan, seperti ukuran tetap setelah dideklarasikan. Jika kita perlu menambah atau mengurangi elemen, kita harus membuat array baru atau menggunakan struktur data lain seperti list (daftar) dalam bahasa pemrograman tertentu.

Array sering digunakan dalam berbagai aplikasi, mulai dari pengolahan data, algoritma pencarian, hingga pemrograman grafis. Dengan kemampuannya untuk menyimpan dan mengelola data.

## 2.3.2 Struktur Data untuk Kontur

Dalam pemrosesan citra, struktur data yang digunakan untuk merepresentasikan kontur sangat penting untuk mendeteksi dan menganalisis bentuk objek dalam citra. Kontur biasanya direpresentasikan sebagai daftar atau array dari titik-titik yang membentuk batas objek.

Setiap titik dalam kontur dapat dinyatakan dengan koordinat yang menunjukkan posisi pixel dalam citra. Dengan menggunakan struktur data seperti daftar terhubung atau array dinamis, kita dapat menyimpan dan mengelola titik-titik kontur dengan efisien, memungkinkan akses dan manipulasi yang cepat selama proses analisis.

Salah satu pendekatan umum untuk menyimpan kontur adalah dengan menggunakan array dua dimensi, di mana setiap elemen dalam array mewakili titik kontur. Dalam hal ini, setiap titik dapat disimpan sebagai pasangan koordinat, misalnya, dalam format .

Struktur data ini memungkinkan kita untuk dengan mudah menghitung panjang kontur, luas area yang dibatasi oleh kontur, dan parameter geometris lainnya. Selain itu, dengan menyimpan kontur dalam format array, kita dapat dengan mudah menerapkan algoritma pemrosesan lebih lanjut, seperti penghalusan kontur atau penghitungan momen kontur.

Selain array, struktur data berbasis objek juga dapat digunakan untuk merepresentasikan kontur, terutama dalam konteks pemrograman berorientasi objek. Dalam pendekatan ini, kita dapat membuat kelas yang menyimpan informasi tentang kontur, termasuk daftar titik, panjang kontur, dan area yang dibatasi.

Dengan cara ini, kita dapat mengelompokkan fungsi-fungsi yang berkaitan dengan kontur, seperti metode untuk menghitung panjang, area, atau untuk menggambar kontur pada citra. Struktur data berbasis objek memberikan fleksibilitas dan kemudahan dalam pengembangan aplikasi yang lebih kompleks, memungkinkan integrasi yang lebih baik dengan komponen lain dalam sistem pemrosesan citra.

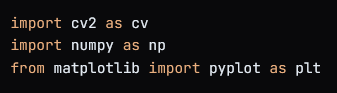
# Implementasi Proyek

## Lingkungan Pemrograman

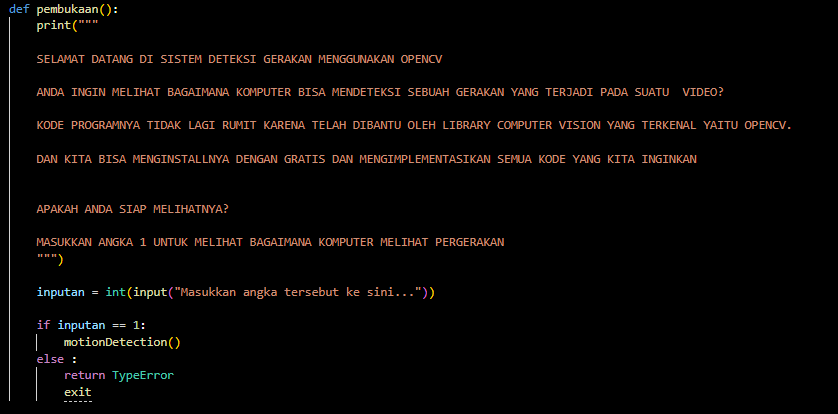
Lingkungan yang digunakan untuk membuat proyek deteksi gerakan ini adalah IDE yang bernama Visual Studio Code yang umumnya digunakan oleh para programmer dan para developer untuk membuat kode program didalamnya.

Kemudian tool yang digunakan dalam proyek ini adalah library python yaitu numpy untuk pemrosesan matematis yang terjadi dalam program ini, kemudian ada matplotlib yang digunakan dalam visualisas sederhana dalam deteksi gerakan ini dan yang terakhir ada opencv yaitu library opensource yang umum digunakan untuk computer vision.

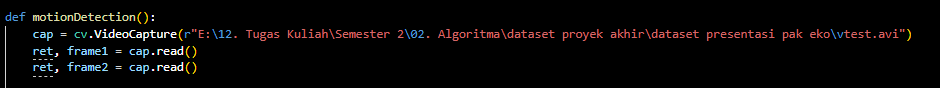
## Penjelasan Kode Program



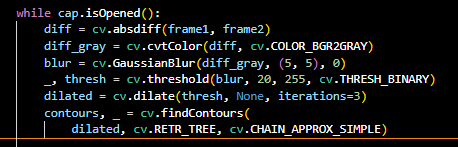
* **cv2**: Merupakan library OpenCV yang digunakan untuk pemrosesan gambar dan video. Library ini menyediakan berbagai fungsi untuk analisis citra dan video.
* **numpy**: Digunakan untuk operasi array dan matriks, meskipun dalam kode ini tidak digunakan secara eksplisit.
* **pyplot**: Dari library Matplotlib yang umumnta ==



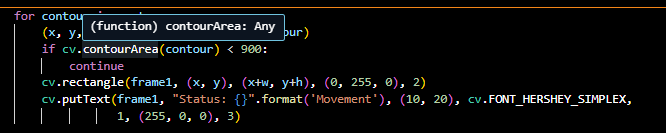
* Fungsi ini menampilkan pesan sambutan kepada pengguna dan menjelaskan tujuan program.
* Pengguna diminta untuk memasukkan angka 1 untuk melanjutkan ke fungsi deteksi gerakan.
* Jika input tidak sesuai, fungsi akan mengembalikan TypeError dan keluar dari program.



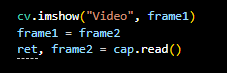
* Fungsi ini bertanggung jawab untuk mendeteksi gerakan dalam video.
* **cv.VideoCapture()**: Membuka file video yang ditentukan. Path video harus sesuai dengan lokasi file di sistem Anda.
* **cap.read()**: Membaca dua frame pertama dari video untuk perbandingan gerakan.



* **cv.absdiff()**: Menghitung perbedaan antara dua frame untuk mendeteksi perubahan.
* **cv.cvtColor()**: Mengubah gambar dari format warna (BGR) ke grayscale untuk memudahkan pemrosesan.
* **cv.GaussianBlur()**: Menghaluskan gambar untuk mengurangi noise yang dapat mengganggu deteksi.
* **cv.threshold()**: Mengubah gambar menjadi biner (hitam-putih) berdasarkan ambang batas yang ditentukan.
* **cv.dilate()**: Memperbesar area putih dalam gambar biner untuk mengisi celah dan membuat kontur lebih jelas.
* **cv.findContours()**: Mencari kontur dalam gambar yang telah diproses.



* Looping melalui setiap kontur yang ditemukan.
* **cv.boundingRect()**: Menghitung kotak pembatas untuk setiap kontur yang terdeteksi.
* Jika area kontur kurang dari 900 piksel, kontur tersebut diabaikan untuk menghindari noise kecil.
* **cv.rectangle()**: Menggambar kotak hijau di sekitar area yang terdeteksi bergerak.
* **cv.putText()**: Menambahkan teks "Status: Movement" di atas video untuk menunjukkan bahwa gerakan terdeteksi.



* **cv.imshow()**: Menampilkan frame video yang sedang diproses dalam jendela dengan judul "Video".
* Frame pertama diperbarui dengan frame kedua untuk iterasi berikutnya, sehingga proses deteksi gerakan dapat berlanjut.



* **cv.waitKey(50)**: Menunggu selama 50 milidetik untuk input dari keyboard. Jika tombol Esc (kode 27) ditekan, loop akan dihentikan.
* Setelah loop selesai, video akan dilepaskan dan semua jendela yang dibuka akan ditutup dengan:

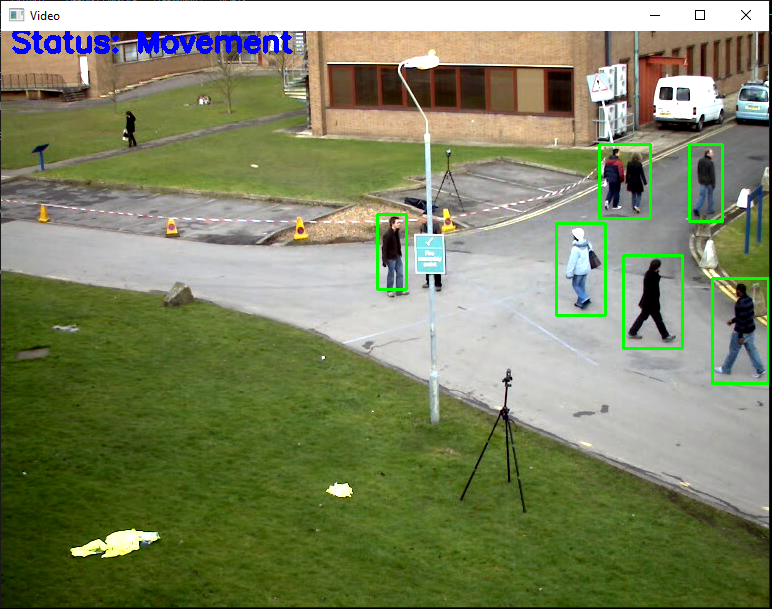




* Bagian ini memastikan bahwa fungsi pembukaan() hanya akan dipanggil jika skrip dijalankan sebagai program utama, bukan saat diimpor sebagai modul.

## Hasil Implementasi Program

Hasil dari deteksi gerakan ditampilkan dalam jendela video, di mana area yang terdeteksi bergerak ditandai dengan kotak hijau. Selain itu, teks "Status: Movement" ditampilkan di sudut atas video untuk memberikan informasi visual kepada pengguna bahwa gerakan telah terdeteksi. Pengguna dapat melihat secara langsung bagaimana sistem merespons perubahan dalam video, memberikan pengalaman interaktif yang menarik.



Selama pengujian, sistem berhasil mendeteksi berbagai jenis gerakan, mulai dari pergerakan lambat hingga cepat. Namun, ada beberapa tantangan yang dihadapi, seperti deteksi gerakan yang tidak diinginkan akibat perubahan pencahayaan atau objek yang bergerak di latar belakang. Untuk mengatasi masalah ini, parameter seperti ambang batas dan ukuran kontur dapat disesuaikan untuk meningkatkan akurasi deteksi.

Implementasi sistem deteksi gerakan menggunakan OpenCV menunjukkan hasil yang memuaskan. Dengan menggunakan teknik pemrosesan citra yang tepat, sistem ini dapat mendeteksi dan menandai gerakan dalam video secara real-time. Hasil ini membuka peluang untuk pengembangan lebih lanjut, seperti integrasi dengan sistem keamanan, analisis perilaku, dan aplikasi interaktif lainnya. Dengan penyesuaian dan pengembangan lebih lanjut, sistem ini dapat menjadi alat yang berguna dalam berbagai bidang, termasuk keamanan, pemantauan, dan interaksi manusia-komputer.

Secara keseluruhan, proyek ini tidak hanya memberikan pemahaman yang lebih baik tentang deteksi gerakan, tetapi juga menunjukkan potensi besar dari teknologi computer vision dalam kehidupan sehari-hari.

# Penutup

## Kesimpulan

Proyek deteksi gerakan ini telah berhasil menunjukkan bagaimana teknik pemrosesan citra dapat diterapkan untuk mengidentifikasi dan menganalisis pergerakan objek dalam video.

Dengan menggunakan serangkaian langkah, mulai dari pengambilan frame hingga penerapan algoritma deteksi gerakan, kami dapat menghasilkan visualisasi yang jelas dan informatif tentang objek yang bergerak. Proses ini tidak hanya meningkatkan pemahaman kita tentang dinamika dalam video, tetapi juga memberikan alat yang berguna untuk berbagai aplikasi, seperti sistem keamanan, analisis perilaku, dan pengenalan objek.

Selama proyek ini, kami juga mengeksplorasi berbagai teknik pemrosesan citra, termasuk konversi ke grayscale, thresholding, pencarian kontur, dan penerapan bounding box. Setiap langkah dalam proses ini berkontribusi pada hasil akhir, memungkinkan kami untuk menandai pergerakan dengan akurasi yang tinggi.

Dengan memanfaatkan struktur data yang tepat, kami dapat menyimpan dan mengelola informasi kontur dengan efisien, yang sangat penting untuk analisis lebih lanjut. Hasil dari proyek ini menunjukkan potensi besar dari teknologi deteksi gerakan dalam meningkatkan keamanan dan efisiensi dalam berbagai bidang.

Kesimpulan dari proyek ini adalah bahwa deteksi gerakan merupakan alat yang sangat efektif dalam pemrosesan citra, yang dapat digunakan untuk berbagai aplikasi praktis. Dengan mengimplementasikan teknik-teknik yang telah dibahas, kami dapat menciptakan sistem yang tidak hanya mendeteksi pergerakan, tetapi juga memberikan informasi yang berguna tentang objek yang terdeteksi.

Manfaat dari proyek ini sangat luas, termasuk peningkatan keamanan melalui sistem pengawasan otomatis, analisis perilaku dalam konteks penelitian, dan pengenalan objek dalam aplikasi industri. Dengan potensi untuk diintegrasikan dengan teknologi pembelajaran mesin, proyek ini membuka jalan bagi pengembangan solusi yang lebih canggih dan inovatif, yang dapat memberikan dampak positif dalam berbagai sektor, mulai dari keamanan hingga analisis data.

# Referensi

Pawar A. (2021). A Survey of Image Processing and Identification Techniques Journal Publication of International Research for Engineering and Management (JOIREM), Vol.10 .

2 R. A. Yuha, M. Al Fiqri, Ashari, R. Pratama, and M. Harahap, “Deteksi Gerakan pada Kamera CCTV dengan Algoritma Frame Difference dan Frame Substraction,” in Seminar Nasional Aptikom (Semnastik) 2019, Aptikom, 2019, pp. 503–511.

3 Ihsan Zul M. , Widyawan , Nugroho L. (2012) Deteksi Gerak dengan Menggunakan Metode Frame Differences pada IP Camera. Proceeding CITEE.

4 Firdausy K., Daryono, Yudhana A. (2008) Webcam Untuk Sistem Pemantauan Menggunakan Metode Deteksi Gerakan. Seminar Nasional Aplikasi Teknologi Informasi 2008.

1. Pawar A. (2021). A Survey of Image Processing and Identification Techniques. Journal Publication of International Research for Engineering and Management (JOIREM), Vol.10 . [↑](#footnote-ref-1)
2. R. A. Yuha, M. Al Fiqri, Ashari, R. Pratama, and M. Harahap, “Deteksi Gerakan pada Kamera CCTV dengan Algoritma Frame Difference dan Frame Substraction,” in Seminar Nasional Aptikom (Semnastik) 2019, Aptikom, 2019, pp. 503–511. [↑](#footnote-ref-2)
3. 3 Ihsan Zul M. , Widyawan , Nugroho L. (2012) Deteksi Gerak dengan Menggunakan Metode Frame Differences pada IP Camera. Proceeding CITEE. [↑](#footnote-ref-3)
4. Firdausy K., Daryono, Yudhana A. (2008) Webcam Untuk Sistem Pemantauan Menggunakan Metode Deteksi Gerakan. Seminar Nasional Aplikasi Teknologi Informasi 2008. [↑](#footnote-ref-4)