**DETEKSI DAN PENGHITUNGAN MANUSIA PADA VIDEO MENGGUNAKAN METODE HOG DAN SVM**

**Muhammad Imron Rosadi1**

Prodi Teknik Informatika, Universitas Yudharta Pasuruan

Purwosari 67162 Pasuruan Jawa Timur

1Email : [Imron\_uyp@yahoo.com](mailto:Imron_uyp@yahoo.com)

**ABSTRAK :**

*Deteksi obyek pejalan kaki lebih sulit daripada mendeteksi obyek lain karena orang dapat menunjukkan gerak yang bervariasi. Selain itu pejalan kaki juga memakai berbagai jenis dan warna pakaian yang berbeda. Oleh karena itu diperlukan suatu metode yang robust yang dapat mendeteksi variabilitas yang tinggi tersebut. Dalam penelitian ini sistem deteksi obyek yang diusulkan menggunakan*

*Histogram of Oriented Gradient (HOG) yang digunakan untuk mereduksi dimensi dan klasifikasi menggunakan Support Vector Machine (SVM). Reduksi dimensi data ini dapat meningkatkan akurasi dan performa sistem. System ini dapat mendeteksi pejalan kaki dari depan dan belakang. Hasil uji coba terhadap metode yang dibuat ini memiliki tingkat akurasi hingga mencapai* adalah 82 %, maka tingkat kegagalannya adalah 18 %. Dan rata-rata waktu komputasi 235.939 detik

**Kata Kunci : Deteksi Pejalan Kaki, HOG, SVM**

1. **PENDAHULUAN**

Masalah deteksi obyek dapat dilihat seperti masalah klasifikasi, bagaimana membedakan obyek yang ingin dideteksi dengan obyek yang lain. Deteksi obyek pejalan kaki lebih sulit daripada mendeteksi obyek lain karena pejalan kaki dapat menunjukkan gerak yang bervariasi. Selain itu pejalan kaki juga memakai berbagai jenis dan warna pakaian yang berbeda[1]. Oleh karena itu diperlukan suatu metode

yang robust yang dapat mendeteksi variabilitas yang

tinggi tersebut. Banyak sekali sistem deteksi obyek yang dibangun focus pada deteksi wajah. Ternyata metode yang digunakan untuk deteksi wajah dapat digunakan untuk deteksi obyek pejalan kaki. Banyak sekali sistem deteksi obyek pejalan kaki menggunkan

informasi gerakan, kamera statis, atau focus pada *tracking*.

Dalam paper ini diimplementasikan sebuah system deteksi obyek pejalan kaki menggunakan HOG

*Histogram of Oriented Gradient (HOG)* sebagai metode untuk mereduksi dimensi[2] dan SVM (*Support Vector* *Machine*) sebagai metode untuk klasifikasi simana sebuah citra masuk kedalam kelas pejalan kaki atau kelas bukan pejalan kaki.Deteksi obyek pejalan kaki ini dibangun pada citra statis dan *gray level*.

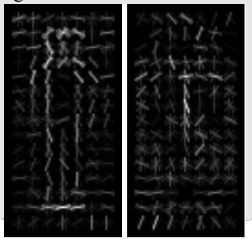
1. **Kajian Pustaka**

**2.1 *Histogram of Oriented Gradient (HOG)***

Dalam *Computer Vision* khususnya untuk kasus deteksi obyek, seringkali perlu untuk melihat nilai dari masing-masing piksel dari sebuah gambar, baik itu nilai intesitas cahaya dalam piksel, nilai komposisi warna tertentu dalam piksel maupun informasi lain yang terkandung dalam piksel. Nilai dari tiap-tiap piksel akan dibagi ke dalam kelompok *Channel* tertentu dan direpresentasikan dengan diagram batang yang biasa disebut dengan Histogram.

*Histogram of Oriented Gradient* (HOG) adalah metode yang digunakan untuk deteksi obyek, histogram berisi *channel-channel* arah/orientasi *gradient* dari piksel-piksel pada gambar, dimana penampilan serta bentuk obyek dapat diketahui melalui hasil komputasi *gradient* dari citra. Tahap awal dari metode ini adalah dengan melakukan normalisasi *gamma* pada citra. Kemudian menghitung *gradient* dari setiap piksel citra, pembagian citra ke dalam *cell*, membentuk histogram dari masing-masing *cell*, membentuk blok dari setiap histogram dan terakhir melakukan normalisasi dari setiap block.

Pada metode ini, fitur HOG dapat diperoleh dari membagi gambar ke dalam sel-sel berukuran n x n, lalu dikelompokkan ke dalam blok-blok berukuran 2n x 2n yang saling beririsan satu sama lain. Dari tiap sel masing-masing blok, di hitung *magnitude* dan orientasi *gradient-nya*. Nilai orientasi ini dihitung dengan menghitung konvolusi dengan matriks 𝐷𝑥= [-1 0 1] dan 𝐷y= [-1 0 1]T terlebih dahulu. Kemudian nilai orientasi tiap pikselnya dikuantisasi kedalam 9 kanal, yaitu 10°, 30°, 50°, 70°, 90°, 110°, 130°,150°, dan 170° menggunakan histogram. Kontribusi piksel terhadap tiap kanal bergantung pada nilai *gradient* *magnitude-*nya. Nilai-nilai dari seluruh kanal dari tiap sel dimasukkan kedalam vektor. Vektor inilah yang menggambarkan fitur HOG dari suatu gambar. [3]



Gambar 1. Contoh fitur HOG yang dihasilkan

**2.3 *Support Vector Machine***

SVM yang diusulkan oleh Vapnik (1995) telah dipelajari secara ekstensif untuk klasifikasi, regresi dan estimasi kepadatan. Gambar 2. adalah arsitektur SVM. SVM memetakan pola input ke ruang fitur dimensi yang lebih tinggi melalui pemetaan *non linear* berdasar teori yang dipilih. Bidang pemisah linear ini kemudian dibangun dalam ruang fitur dimensi tinggi. Dengan demikian, SVM adalah *linear classifier* di ruang parameter, tapi itu menjadi *non linear classifier* sebagai akibat dari pemetaan *non linear* dari ruang pola input ke ruang fitur dimensi tinggi. Bila data pelatihan berdimensi *m* adalah *xi* (*i* = 1, ..., M) dan masing-masing kelas labelnya adalah *yi*, di mana *yi* = 1 dan *yi* = -1 untuk kelas 1 dan 2. Jika data input terpisah secara linear di ruang fitur, maka fungsi keputusan dapat ditentukan:

*D(x) = wtg(x) + b* (2.18)

di mana *g(x)* adalah fungsi pemetaan yang memetakan *x* ke dalam ruang dimensi 1, *w* adalah vektor dimensi dan 1, dan *b* adalah skalar. Untuk memisahkan data secara linier, fungsi keputusan memenuhi kondisi berikut:

*yi(wtg(xi) + b) > 1* untuk *i* = 1, ..., M (2.19)



Gambar 2 Arsitektur SVM

Jika masalah terpisah secara linier dalam ruang fitur, maka fungsi keputusan jumlahnya tak terbatas. Di antara fungsi-fungsi tersebut, diperlukan *hyperplane* dengan margin terbesar antara dua kelas. Margin adalah jarak minimum yang memisahkan *hyperplane* terhadap data input dan ini dihasilkan dari *|****D(x)|/||w||***. Sehingga didapatkan *hyperplane* pemisah dengan margin maksimal yang optimal memisahkan *hyperplane*.

Dengan asumsi bahwa margin adalah *ρ*, kondisi berikut harus memenuhi:

**untuk *i* = 1, ..., M (2.20)

Hasil perkalian produk dari *ρ* dan ||*w*|| adalah tetap:

*ρ* ||w|| =1 (2.21)

Untuk mendapatkan *hyperplane* pemisah yang optimal dengan margin maksimal, *w* dengan ||*w*|| harus ditemukan. Mengarahkan ke pemecahan masalah optimasi berikutnya. Dengan meminimalkan

** (2.22)

dan mengikuti batasan:

*yi(wtg(xi) + b) > 1* untuk *i* = 1, ..., M (2.23)

Bila data pelatihan tidak linier dipisahkan, digunakan *slack variable* ξi ke persamaan (2.24):

*yi(wtg(xi) + b) > 1-* ξiξi> 0 untuk *i* = 1, ..., M (2.24)

*Hyperplane* pemisah yang optimal telah ditentukan sehingga maksimalisasi dari margin dan meminimalisasi dari kesalahan *training* didapatkan. Dengan meminimalkan

** (2.25)

mengikuti batasan:

*yi(wtg(xi) + b) > 1-* ξiξi> 0 untuk *i* = 1, ..., M (2.26)

di mana *C* adalah parameter yang menentukan *tradeoff* antara margin maksimum dan kesalahan klasifikasi minimum dan *ρ* adalah 1 atau 2. Jika *ρ* = 1, SVM disebut SVM dengan *soft margin* L1 (L1-SVM), dan jika *ρ* = 2, SVM dengan *soft margin* L2 (L2-SVM). Pada SVM konvensional, *hyperplane* pemisah yang optimal diperoleh dengan memecahkan masalah pemrograman kuadratik.

Fungsi kernel memungkinkan operasi yang akan dilakukan di ruang input bukan di ruang fitur dimensi tinggi. Beberapa contoh fungsi kernel adalah *K(u, v)* = *vTu* (SVM linier); *K(u, v)* = *(vTu + 1)n* (SVM polinomial derajat n); *K(u, v)* = exp(-||*u - v||2 / 2σ2*) (SVM fungsi *radial bases* – SVM RBF); *K(u, v)* = tanh*(KvTy + ο)* (*neural* SVM dua *layer*) di mana σ, ĸ, *ο* adalah konstanta [Vapnik, 1995; Cortes, 1995]. Namun, fungsi kernel yang tepat untuk suatu masalah tertentu tergantung pada data, dan sampai saat ini belum ada metode yang baik tentang cara memilih fungsi kernel.

**3. METODE PENELITIAN**

Proses deteksi obyek pejalan kaki yang diimplementasikan menggunakan metode Histogram of Oriented Gradients.(HOG) Pada metode ini juga dilakukan proses learning dengan *SVM*. Alur proses deteksi digambarkan pada gambar 3-

Input Gambar

Praproses

Hasil dan evaluasi

Ektraksi ciri (HOG)

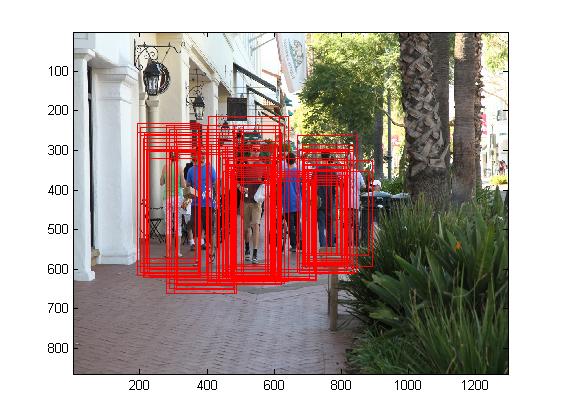
Proses deteksi (SVM)

Gambar 3. Proses deteksi obyek dengan metode HOG+SVM

1. Mulai Input gambar
2. Praproses : Normalisasi dan perbaikan citra
3. Ekstraksi Fitur : Setelah gambar di-resize menjadi satu ukuran, lalu sistem akan mengekstraksi ciri. Ekstraksi ciri dilakukan dengan cara memasukkan nilai pixel gambar ke dalam sebuah vektor atau array 1 dimensi. Sebelum dimasukkan ke dalam array, pixel diubah terlebih dahulu menjadi nilai 1 dan -1 dimana 1 mewakili warna hitam sedangkan -1 mewakili warna putih pada gambar.
4. Pengenalan Obyek: Setelah ciri didapatkan maka proses selanjutnya adalah klasifikasi obyek menggunakan algoritma SVM. Proses ini berfungsi untuk mengenali pejalan kaki. Untuk dapat mengenali pejalan kaki, sistem perlu dilatih (training) terlebih dahulu dengan algoritma SVM supaya dapat mengenali masing-masing pejalan kaki. Proses pelatihan akan menghasilkan bobot untuk tiap pejalan kaki, kemudian bobot tersebut digunakan untuk pengujian terhadap input pejalan kaki.

**4. HASIL DAN PEMBAHASAN**

Setelah semua percobaan telah selesai dilakukan maka perlunya untuk diadakan evalusi tentang program yang digunakan. Dari evaluasi tersebut didapatkan kelebihan serta kekurangan dari program yang sedang diujicoba. Tujuan evaluasi atau pengujian ini adalah untuk mengetahui tingkat akurasi dari program aplikasi identifikasi pejalan kaki ini. Berikut sample hasil pengujian program :



Gambar 4. Citra Ujicoba



Gambar 5 Citra hasil ujicoba

Tabel 1 hasil akurasi uji coba

|  |  |  |
| --- | --- | --- |
| **Data Training** | **Akurasi** | **Waktu (Detik)** |
| **1** | 100% | 241.58 |
| **2** | 100% | 240.70 |
| **3** | 0% | 231.61 |
| **4** | 75% | 242.95 |
| **5** | 66.67% | 226.69 |
| **6** | 100% | 244.57 |
| **7** | 100% | 242.95 |
| **8** | 100% | 225.39 |
| **9** | 100% | 230.33 |
| **10** | 75% | 232.62 |
| **Rata-rata** | **82%** | **235.939** |

Dari hasil yang terlihat pada tabel 1 tersebut dapat dianalisis bahwa penambahan jumlah data training pejalan kaki maupun data training bukan pejalan kaki tidak berpengaruh terhadap tingkat akurasi. Maka dapat dihitung tingkat akurasi keseluruhan dari aplikasi sebagai berikut :

Tingkat akurasi keseluruhan aplikasi ini adalah 82 %, maka tingkat kegagalannya adalah 18 %. Dan rata-rata waktu komputasi 235.939 detik

Berdasarkan data sampel yang telah dilakukan pengujian, kegagalan pendeteksian disebabkan oleh beberapa hal yaitu:

1. Kondisi pejalan kaki yang kurang bagus.
2. Pengaruh cahaya yang tidak merata.
3. Dipengaruhi objek yang karaternya sama dengan pejalan kaki
4. Sulit membedakan karakter yang miring.
5. Jarak antar pejalan kaki yang terlalu rapat satu sama lain. .

**5. KESIMPULAN**

Sesuai dengan pembahasan mengenai identifikasi pejalan kaki menggunakan algoritma HOG dan SVM ini, maka kesimpulan yang dapat diambil oleh penulis adalah sebagai berikut:

1. Aplikasi identifikasi pejalan kaki ini dapat mengakses informasi jumlah pejalan kaki secara otomatis.
2. Aplikasi identifikasi pejalan kaki ini dapat mengidentifikasi jumlah pejalan kaki dengan tingkat akurasi program mencapai 82 %, maka tingkat kegagalannya adalah 18 %. Dan rata-rata waktu komputasi 235.939 detik
3. **SARAN**

Penelitian ini masih jauh dari sempurna dan masih perlu banyak perbaikan dan pengembangan supaya menjadi lebih baik lagi. Berikut ini saran untuk pengembang penelitian ini :

1. Menambahkan metode untuk menghilangkan noise yang besar pada gambar supaya gambar yang dihasilkan dapat mudah dikenali oleh sistem, seperti metode morfologis matematis.
2. Aspek pencahayaan harus diperhatikan sebaik-baiknya sehingga menghasilkan pencahayaan yang merata.
3. Menggunakan metode pengenalan wajah manusia agar pendeteksian lebih akurat.

**KAJIAN PUSTKA**

1. Malago´n-Borja, Luis., Fuentes, Olac.2007. *Object detection using image reconstruction with PCA*. Image and Vision Computing.
2. R. Gunn, Steve.1998. *Support Vector Machines for Classification and Regression.* Southampton
3. Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. *Computer Vision and Pattern Recognition*.
4. Sembiring, Krisantus. 2007. *Penerapan Teknik Support Vector Machine untuk Pendeteksian Intrusi pada Jaringan*. Bandung:Indonesia