Pengaruh Orientasi dari SURF dan U-SURF Terhadap Waktu Komputasi pada Pelacakan Objek

Ari Putra Nugraha [1)], Suryo Adhi Wibowo [2)], Nur Andini [3)]

[1),2),3)]Fakultas Teknik Elektro, Telkom University
Jl. Telekomunikasi Nomor 1 Bandung INDONESIA
Email : ariputranugraha03@gmail.com

Abstrak. *Pelacakan objek mangalami banyak kemajuan tiap tahunnya dan melahirkan banyak metode yang semakin membaik. Banyak peneliti yang menggeluti salah satu bidang computer vision ini untuk memberikan manfaat yang baik bagi kehidupan manusia dalam bidang Internet of Things. Ekstraksi fitur sendiri diperlukan dalam proses pelacakan objek. Salah satu metode ekstraksi fitur yang bersifat scale dan rotation invariant, yakni Speeded-Up Robust Feature (SURF). SURF sendiri memiliki versi lain yang hanya bersifat scale invariant, yaitu Upright-SURF(U-SURF), sehingga dalam ekstraksi fiturnya tidak menghitung orientasi atau rotasi dari suatu citra. Dalam pengimplementasiannya pada pelacakan objek, SURF akan melakukan ekstraksi fitur dari dua frame dan mencocokkannya. Lalu, Random Sample Consensus (RANSAC) membagi fitur menjadi inlier dan outlier kemudian melakukan estimasi transformasi untuk mendapatkan lokasi objek target. Pada paper ini, kami membandingkan SURF dengan U-SURF dalam pelacakan objek dengan mempertimbangkan waktu komputasi dan jumlah frame yang gagal dilacak dalam 2 sekuen yang terdiri atas 440 frame. Berdasarkan penelitian yang telah dilakukan, didapatkan hasil bahwa U-SURF dengan nilai parameter Metric Threshold sebesar 100 mendapatkan hasil yang terbaik dengan 20 frame yang tidak terdeteksi dan waktu komputasi selama 37.6 detik.*

***Kata kunci**: ekstraksi fitur, pelacakan objek, scale-invariant, rotation-invariant.*

1. Pendahuluan

Machine learning merupakan salah satu cabang ilmu dari *artificial intelligence* yang berkembang pesat di banyak negara. *Machine learning* sendiri dalam arsitekturnya terdapat ekstraksi fitur dan klasifikasi yang berada pada satu sistem yang sama, dimana ekstraksi fitur merupakan salah satu proses yang penting dalam pelacakan objek untuk menghemat komputasi kompleksitas dan meningkatkan akurasi [1]. Pelacakan objek merupakan salah satu ilmu yang penting dalam *computer vision* dan saat ini telah banyak diimplementasikan dalam dunia teknologi, seperti keamanan, automasi, maupun dunia medis [2]. Walaupun pelacakan objek telah berkembang dengan pesat, namun beberapa masalah performansi, seperti ketidakmampuan beradaptasi terhadap skala objek menjadi salah satu hal yang patut diperhitungkan. Masalah ini dapat terjadi karena pada proses ekstraksi fiturnya yang kurang optimal dalam menghitung skala maupun rotasi dari objek target.

Dalam proses ekstraksi yang dilakukan, fitur dapat berupa titik, garis, bulat, maupun *blob*. Pada tingkatan yang lebih tinggi, fitur juga dapat direpresentasikan sebagai *landmark* (*corner*, kolom, pintu) [3]. Kemudian, untuk detektor fitur yang sering digunakan adalah detector Harris-corner [4] yang menghitung fitur berdasarkan nilai eigen. Namun, sayangnya detector ini belum *invariant* terhadap skala sehingga, untuk dapat melacak objek yang berubah-ubah terhadap skala, masih belum optimal. Mengatasi hal tersebut, diciptakanlah sebuah metode bernama *automatic scale selection* [5] yang dapat menghitung ruang skala pada fitur dalam suatu citra, tentunya tiap fitur memiliki karakteristik skalanya sendiri. Pada metode ini, Peter Majer melakukan eksperimen dengan menggabungkan matriks Hessian dengan metode Laplacian untuk dapat mendeteksi fitur yang direpresentasikan dengan struktur *blob* yang kemudian memunculkan metode-metode baru yang memiliki sifat *invariant* terhadap skala.

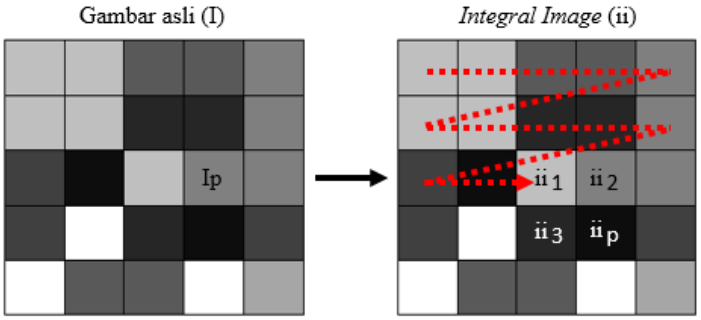
Penelitian yang terkait hal ini adalah mengenai metode *Scale Invariant Feature Transform* (SIFT) [6] menjelaskan bahwa fitur-fitur yang diekstraksi teridentifikasi dengan mencari lokasi dan skala pada suatu citra. Hal ini dilakukan dengan mengkonstruksi piramida Gaussian dan menghitung local-maxima. Lalu selanjutnya, menghitung orientasi pada setiap fitur yang didapat. Fitur-fitur ini sering disebut sebagai *keypoint* atau *interest* point. Hasil dari perhitungan ruang skala, orientasi, dan lokasi

pada tiap *keypoint* kemudian digunakan SIFT untuk dapat mengkonstruksi fitur yang invariant terhadap skala dan orientasi. Secara garis besar, SIFT memiliki 4 tahapan utama, yaitu:

1) Proses seleksi ruang skala
2) Lokalisasi *keypoint*
3) Perhitungan orientasi
4) Deskriptor *keypoint*

Pada penelitian yang dilakukan oleh Herbert Bay [7] ditemukan sebuah metode detektor-deskriptor fitur yang dinamakan Speeded-Up Robust Features (SURF). Detektor ini didasari oleh perhitungan matriks Hessian. Perbedaan dari metode ini dengan SIFT adalah penggunaan integral *image* untuk melakukan *filtering* terhadap suatu citra yang dapat menekan waktu komputasi, sehingga metode ini kemudian disebut sebagai Fast Hessian *detector*. Pada deskriptornya, SURF menghitung respons dari distribusi gelombang Haar pada tiap *keypoint*. Pada tahap ini, *integral image* juga digunakan untuk dapat melakukan komputasi fitur dengan lebih cepat.

Integral image sendiri sering disebut sebagai *summed area table*. Konsep dari integral *image* adalah dengan menghitung tiap area dari suatu citra dengan 4 akses dan 3 penjumlahan, sehingga proses ini dapat menekan waktu komputasi.



Gambar 16. Representasi dari integral *image*

Gambar 1 menunjukkan perhitungan dari *integral image* (ii) yang dapat dihitung menggunakan persamaan 1.

$$ii_p = ii_2 + ii_3 - ii_1 + I_p \quad \text{..... (1)}$$

Sedangkan, untuk detektornya berdasarkan pada matriks Hessian dengan cara menghitung determinan maksimum dengan fitur berupa struktur *blob*. Mula-mula, setiap k=(x,y) untuk citra I, matriks Hessian H(k,sigma) dengan titik k dan skala sigma, dapat didefinisikan pada persamaan 1 [8].

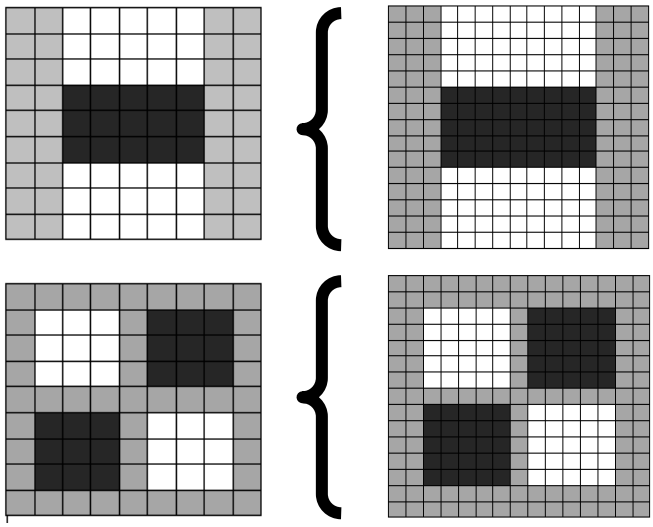
$$H(k,\sigma) = \begin{bmatrix} L_{xx}(k,\sigma) & L_{xy}(k,\sigma) \\ L_{yx}(k,\sigma) & L_{yy}(k,\sigma) \end{bmatrix} \quad \text{..... (2)}$$

Dimana L_{xx}(k,sigma) dan L_{xy}(k,sigma) adalah konvolusi dari turunan kedua Gaussian $\frac{\delta^2 g(\sigma)}{\delta x^2}$ dan $\frac{\delta^2 g(\sigma)}{\delta x \delta y}$ untuk citra I. Dari persamaan 1, didapatkan hasil dari turunan kedua Gaussian yang digunakan untuk *box filter*. Lalu, dari hasil ini akan dicari determinan dengan menggunakan persamaan 2 [8].

$$\det(H = D_{xx}D_{yy} - (0.9D_{xy})^2 \quad \text{..... (3)}$$

Dimana D_{xx} dan D_{yy} menunjukkan hasil *box filter* dari persamaan 1 pada arah x dan y. Begitu juga untuk D_{xy} sebagai hasil *box filter* pada arah xy.

Untuk menghitung ruang skala pada suatu citra, kebanyakan dari metode ekstraksi fitur akan menghitung skala menggunakan Gaussian *filter* dengan melakukan sub-sampel pada citra secara berulang-ulang untuk mendapatkan level piramida yang diinginkan. Daripada melakukan sub-sampel dan menghitung skala pada tiap citra yang di sub-sampel, SURF cenderung menghitung skala dengan memperlebar ukuran dari *box filter*, sehingga citra tidak perlu di sub-sampel dan dapat menekan waktu komputasi. Gambar 2 menunjukkan bagaimana ukuran *box filter* diperbesar untuk memperoleh ruang skala yang diinginkan. Gambar pertama (atas) menunjukkan *box filter* pada arah x, sedangkan gambar kedua (bawah) menunjukkan *box filter* pada arah xy.



Gambar 17. Ukuran *box filter* yang diperbesar[8]

Sedangkan, untuk mendapat sifat invariant terhadap rotasi, maka SURF menghitung orientasi dari tiap keypoint [9]. Lalu, SURF menghitung respons dari gelombang Haar pada arah x dan y dari radius 6s disekitar keypoint, dimana s merupakan skala dari suatu keypoint. Kemudian, respons dari gelombang Haar dihitung bobotnya menggunakan Gaussian dan dicari orientasi dominannya dengan melakukan estimasi total dari seluruh respons gelombang Haar. Pada SURF sendiri, dengan melakukan perhitungan terhadap orientasi dari suatu citra membuat SURF menjadi *invariant* terhadap rotasi. Namun, terdapat versi lain dari SURF, yakni Upright SURF (U-SURF) yang tidak *invariant* terhadap rotasi. Sehingga U-SURF hanya *invariant* terhadap skala saja. Dengan mengorbankan sifat *rotation-invariant* ini, maka waktu komputasi dapat ditekan. Pada algoritma SURF, terdapat suatu parameter yang disebut dengan *Metric Threshold*, dimana parameter ini merupakan ambang batas untuk nilai fitur yang mempengaruhi jumlah fitur yang diekstrak dari SURF [10]. Semakin besar nilai parameter, maka jumlah fitur yang diesktrak akan semakin sedikit, begitu juga sebaliknya.

Penelitian yang dilakukan oleh Silica Kole [11] mengenai Random Sample Consensus (RANSAC) mendapatkan hasil bahwa fitur cocok antara kedua citra tidak sepenuhnya sesuai. Maka dari itu fitur-fitur yang cocok tersebut dibagi menjadi dua jenis, yakni *inlier* dan *outlier*. Keluaran dari RANSAC adalah berupa garis dari hasil transformasi disekitar *keypoint* [12].

2. Pembahasan

Pengujian dilakukan pada 2 sekuen yang memiliki total 440 *frame* menggunakan aplikasi komputasi numerikal. Untuk dapat melakukan pelacakan terhadap objek yang bergerak. Maka, pada frame pertama dilakukan *crop* pada objek yang akan dilacak. Lalu, hasil *crop* yang berisikan objek akan dilakukan ekstraksi fitur lalu dicocokkan dengan fitur pada *frame* 2 hingga *frame* akhir secara keseluruhan tiap piksel pada *frame* tersebut. Secara garis besar, proses algoritma ini dapat dilihat dari gambar 3.

Gambar 18. Diagram blok algoritma

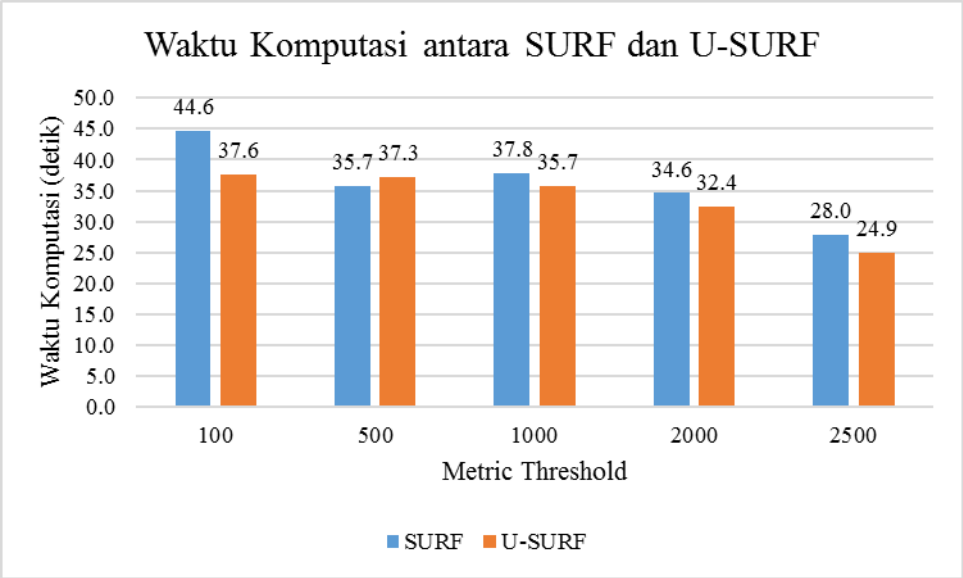
Laptop yang digunakan untuk melakukan pengujian memiliki spesifikasi Intel Core i7, nVidia GeForce GT755M dengan RAM 8GB. Nilai parameter dari *Metric Threshold* yang diuji adalah 100, 500, 1000, 2000, dan 2500 yang dapat dilihat pada Tabel 1.

Tabel 10. Hasil pengujian terhadap 2 sekuen yang terdiri atas 440 *frame.*

No	Metric Threshold	Jumlah frame yang tidak terdeteksi	
		SURF	U-SURF
1	100	17	20
2	500	19	27
3	1000	32	38
4	2000	59	56
5	2500	106	125

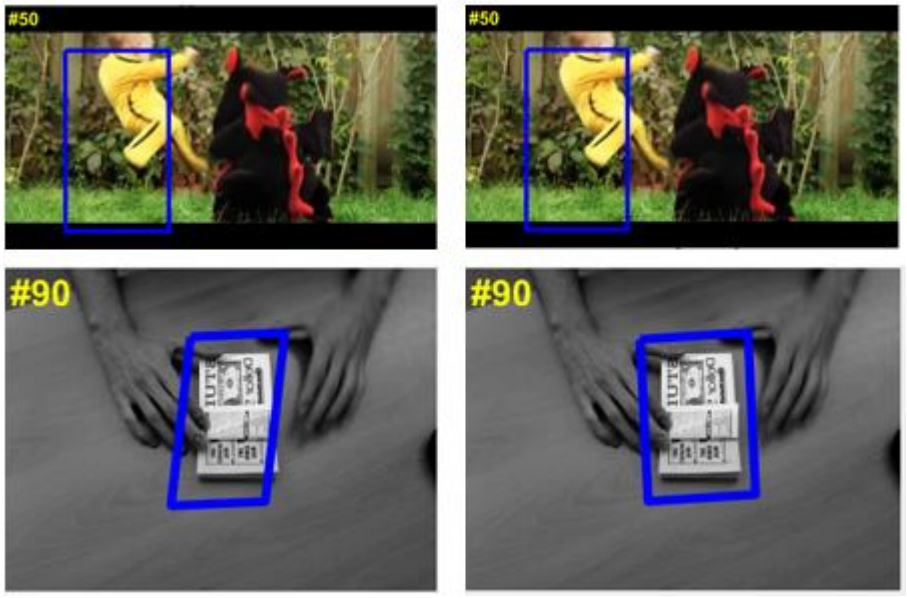
Pada tabel 1 menunjukkan bahwa *Metric Threshold* berpengaruh pula terhadap performansi pelacakan objek, dimana semakin tinggi nilai dari *Metric Threshold*, maka jumlah *frame* yang gagal untuk dilacak semakin banyak. Begitu juga sebaliknya, semakin rendah nilai dari *Metric Threshold*, maka jumlah frame yang gagal untuk dilacak semakin sedikit. Ini terjadi karena, parameter ini menunjukkan toleransi terhadap seberapa besar fitur yang dapat diekstraksi. Jika semakin banyak fitur yang diekstraksi, maka kemungkinan gagal akan menjadi kecil dan begitu juga sebaliknya. Jumlah *frame* yang tidak terdeteksi tersebut merupakan jumlah frame yang diakumulasi dari sekuen 1 dan sekuen 2. Sekuen 1 terdiri atas 113 *frame* dan sekuen 2 terdiri atas 327 *frame*.

ISSN 2085-4218
2 Februari 2019

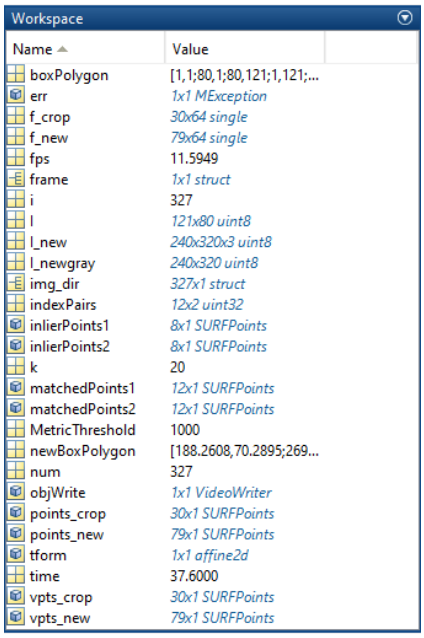


Gambar 19. Waktu Komputasi antara SURF dan U-SURF terhadap nilai *Metric Threshold*

Dari hasil pengujian yang telah dilakukan, waktu komputasi antara SURF dan U-SURF berbeda diantara 2 sekuen. Gambar 4 menunjukkan 5 nilai parameter *Metric* Threshold yang diuji, yaitu 100, 500, 1000, 2000, 2500. Semakin nilai besar nilai parameter, maka waktu komputasi akan semakin meningkat, namun mengorbankan performansinya, dalam hal ini semakin besar nilai parameternya, maka frame yang gagal dilacak akan semakin banyak. Kedua metode ini memiliki perhitungan yang berbeda. Pada SURF, skala dan orientasi dihitung tiap *frame.* Sedangkan, pada U-SURF hanya skala saja dihitung tanpa menghitung orientasi, sehingga U-SURF memproses citra lebih cepat dibanding SURF. Hal ini mempengaruhi hasil dari pelacakan objek yang dapat dilihat pada Gambar 4.



Gambar 20. Hasil estimasi lokasi objek target dari SURF(kiri) dan U-SURF(kanan)



Gambar 21. Jumlah variabel yang digunakan dalam algoritma.

Pengujian yang dilakukan memiliki beberapa variable dalam prosesnya. Pada Gambar 6 menunjukkan jumlah variabel yang digunakan dalam algoritma. Gambar 6 merupakan hasil analisa dari U-SURF pada nilai *Metric Threshold* sebesar 100 dari 2 sekuen, dimana variable k merupakan jumlah *frame* yang gagal dilacak, sedangkan *time* merupakan variabel yang merepresentasikan waktu komputasi.

3. Kesimpulan

Berdasarkan hasil pengujian yang dilakukan pada 3 sekuen, maka dapat disimpulkan bahwa:

1) Waktu komputasi dari U-SURF lebih cepat dibandingkan dengan SURF karena pada U-SURF tidak perlu menghitung orientasi dari fitur, sehingga dapat menekan waktu komputasi.
2) Nilai parameter metric threshold juga berperan dalam mempengaruhi waktu komputasi. Semakin besar nilai parameter maka semakin cepat waktu komputasi. Begitu juga sebaliknya, semakin kecil nilai parameter maka semakin lambat waktu komputasi. Ini disebabkan oleh jumlah fitur yang diproses.
3) Hasil dari algoritma ini berupa *polygon*, bukan *bounding box*. Namun, hasil *polygon* ini dapat dikonversi menjadi *bounding box* sehingga dapat diuji pada parameter *success plot* dan *precision plot*.
4) Dari hasil perbandingan antara 2 metode, U-SURF dengan nilai parameter *Metric Threshold* sebesar 100 dipilih karena selisih jumlah *frame* yang tidak terdeteksi hanya sebesar 3, namun waktu komputasi yang diperoleh U-SURF 7 detik lebih cepat disbanding SURF.

Ucapan Terima Kasih

Kami sampaikan terimakasih kepada yang telah memberikan dana penelitian, sehingga dapat tereksekusi dengan baik. Berkat pembimbing, peneliti mampu melakukan pengujian berdasarkan ilmu yang telah diajarkan. Yang kedua, kami sampaikan terimakasih kepada jajaran Fakultas Teknik Elektro dari Universitas Telkom yang telah memperkenankan kami untuk melakukan publikasi ini.

Daftar Pustaka

[1] B. Han and L. Davis, *"Object tracking by adaptive feature extraction,*" *Proc. - Int. Conf. Image Process. ICIP*, vol. 3, pp. 1501–1504, 2004.

[2] Y. Wu, J. Lim, and M.-H. Yang, "1- *Object Tracking -Online Object Tracking: A Benchmark*," *2013 IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 2411–2418, 2013.

[3] D. Castro, U. Nunes, and A. Ruano, *"FEATURE EXTRACTION FOR MOVING OBJECTS TRACKING SYSTEM IN INDOOR ENVIRONMENTS* Daniel Castro*, Urbano Nunes † , António Ruano," *Transform*, 2004.

[4] C. Harris and M. Stephens, *"A Combined Corner and Edge Detector," Procedings Alvey Vis. Conf. 1988*, p. 23.1-23.6, 1988.

[5] P. Majer, *"The Influence of the γ-Parameter on Feature Detection with Automatic Scale Selection BT - Scale-Space and Morphology in Computer Vision," Scale-sp. Morphol. Comput. Vis.*, no. Chapter 21, pp. 245–254, 2001.

[6] K. Yan and R. Sukthankar, "PCA-SIFT: *A More Distinctive Representation for Local Image Descriptors*," *CVPR*, pp. 506 – 513, 2004.

[7] H. Bay, T. Tuytelaars, and L. Van Gool, *"SURF: Speeded up robust features.,*" *Eur. Conf. Comput. Vis.*, 2006.

[8] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "*Speeded-Up Robust Features (SURF)*," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, 2008.

[9] Y. Sakai, T. Oda, M. Ikeda, and L. Barolli, *"An object tracking system based on SIFT and SURF feature extraction methods*," *Proc. - 2015 18th Int. Conf. Network-Based Inf. Syst. NBiS 2015*, pp. 561–565, 2015.

[10] A. P. Nugraha, S. A. Wibowo, and N. Andini, *"Performance Analysis of Metric Threshold in SURF for Object Tracking*," *2nd Symp. Futur. Telecommun. Technol.*, pp. 48–49, 2018.

[11] S. Kole, C. Agarwal, T. Gupta, and S. Singh, "SURF and RANSAC: *A Conglomerative Approach to Object Recognition*," *Int. J. Comput. Appl.*, vol. 109, no. 4, pp. 975–8887, 2015.

[12] P. Dreuw, P. Steingrube, H. Hanselmann, and H. Ney, "SURF-Face : *Face Recognition Under Viewpoint Consistency Constraints Human Language Technology and Pattern Recognition*," *Pattern Recognit.*, vol. 60, no. October, pp. 2008–2008, 2008.