Analisa Kinerja Algoritma Detektor Sudut pada Citra Noise Komparasi Operator (Moravec, Susan, Harris, FAST, Eigen dan Forstner)

Umar Al Faruq, Homa. P. Harahap
Sistem Informasi, Fakultas Industri Kreatif dan Telematika, Universitas Trilogi
Jl. Kampus Trilogi, No.1, Kalibata, Jakarta Selatan 12760
faruq@trilogi.ac.id

Abstrak–Penelitian ini bertujuan untuk medeskripsikan komparasi kinerja dari masing-masing algoritma pendeteksi sudut pada citra *noise*. Citra yang digunakan sebagai masukan adalah citra dengan format *grayscale* (citra abu-abu) dan diberikan beberapa jenis *noise*. Algoritma yang dibandingkan adalah algoritma *Moravec*, *Susan*, *Harris*, FAST, Eigen dan *Forstner*. Jenis *noise* yang akan di gunakan adalah *gaussian*, *poisson*, *salt & pepper* dan *speckle*. Hasil pengujian didapatkan sebagai berikut; detektor Moravec mengahasilkan 1.000 titik sudut pada citra *noise* (*gaussian*, *poisson*, *salt and pepper*, *speckle*), rata-rata waktu proses pendeteksian sebesar 2,19 detik. Detektor Susan menghasilkan 100 titik sudut pada citra *noise* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*) dengan rata-rata waktu proses pendeteksian sebesar 23,99 detik. Detektor Harris menghasilkan 3.521 titik sudut pada citra *noise gaussian*, 2.125 titik sudut pada citra *noise poisson*, 1.559 titik sudut pada citra *noise salt and pepper*, dan 9.908 titik sudut pada citra *noise speckle*, rata-rata waktu proses pendeteksian 0,91 detik. Detektor FAST menghasilkan 2.461 titik sudut pada citra *noise gaussian*, 782 titik sudut pada citra *noise poisson*, 2.473 titik sudut pada citra *noise salt and pepper*, 841 titik sudut pada citra *noise speckle*, dengan waktu rata-rata proses pendeteksian sebesar 0,26 detik. Detektor Eigen menghasilkan 7.891 titik sudut pada citra *noise gaussian*, 7.970 titik sudut pada citra *noise poisson*, 2.238 titik sudut pada citra *noise salt and pepper* dan 9.638 titik sudut pada citra *noise speckle*, dengan rata-rata waktu proses pendeteksian sebesar 1,23 detik. Detektor Frostner menghasilkan 821 titik sudut pada citra *noise gaussian*, 868 titik sudut pada citra *noise poisson*, 940 titik sudut pada citra *noise salt and pepper* dan 854 titik pada citra *noise speckle*, degan rata-rata waktu pendeteksian sebesar 6,07 detik. Hasil pengujian akurasi setiap detektor sudut pada citra yang memilki *noise* menyatakan bahwa; a) seluruh detektor sudut tidak mampu menemukan titik-titik sudut dengan tepat, b) seluruh detektor sudut tidak akurat dalam menunjukkan lokasi titik sudut, c) hanya detektor Moravec dan Susan yang stabil terhadap perulangan, d) seluruh detektor sudut tidak stabil atau sangant sensitif terhadap semua tipe *noise*. Hasil pengujian dalam penelitian ini memperlihatkan bahwa seluruh detektor sudut sangat sensitif terhadap *noise*, dengan pengertian lain bahwa tingkat akurasi hasil pendeteksian setiap detektor sudut akan sangat dipengaruhi oleh *noise* dan tipe *noise*.

Kata Kunci – Detektor Sudut, Titik Sudut, *Grayscale*, *Noise*, *Moravec*, *Susan*, *Harris*, *FAST*, *Eigen* dan *Forstner*.

Abstract–This study aimed to describe the comparative performance of each corner detection algorithm to the image noise. Imagery used as fill is an image with grayscale and given some kinds of noise. The algorithm will be compared is the algorithm Moravec, Susan, Harris, FAST, Eigen and Forstner. The kinds of noise that will be used are gaussian, poisson, salt & pepper and speckle. The test results obtained as follows; Moravec detector produces 1.000 corner points, on the image with noise (gaussian, poisson, salt and pepper, speckle), The detection time processing average is 2,19 seconds. Susan detector produces 100 points corner on the image noise (gaussian, poisson, salt and pepper and speckle) with an average time of detection process is 23,99 seconds. Harris detector produce 3.521 points corner on the gaussian noise, 2.125 points corner on the poisson noise, 1.559 points corner on the salt and pepper noise, 9.908 points corner on the speckle noise, the average time of detection process is 0,91 seconds. FAST detector produces 2.461 corner points on the gaussian noise, 782 points corner points on the poisson noise, 2473 corner points corner on the salt and pepper noise, 841 corner points on the speckle noise, the average time detection process is 0.26 seconds. Eigen detector produces 7.891 corner points on the gaussian noise, 7.970 corner points on the poisson noise, 2.238 corner points on the salt and pepper noise and 9.638 corner points on the speckle noise, the average time detection process is 1,23 seconds. Frostner detector produces 821 corner points on the gaussian noise, 868 corner points on the poisson noise, 940 corner points on the salt and pepper noise and 854 corner points on the speckle noise, the average time detection process is 6,07 seconds. Results of testing the accuracy of each corner detector on the image noise mentioned that; a) all corner detector unable to find the corner points appropriately, b) all corner detector does not accurately show the location of the points corner, c) only Moravec and Susan detector stable against repeatation, d) all the corner detector is very sensitive to all types of noise. The test results in this study show that the entire corner detector is very sensitive to noise, in another word that the degree of accuracy of detection results every corner detector will be strongly influenced by the noise and the type of noise.

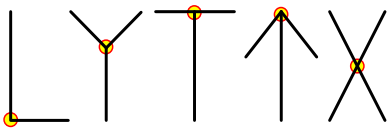
Keyword – Corner Detector, Corner Points, Grayscale, Noise, Moravec, Susan, Harris, FAST, Eigen and Forstner.

I. PENDAHULUAN

Secara fisik membedakan antara sudut, garis, sisi atau titik akan lebih mudah, tetapi secara matematis akan menjadi masalah yang kompleks karena mekanisme untuk membedakan piksel-piksel (sudut, garis, atau titik) yang berada dalam sebuah citra tentu akan menjadi tantangan tersendiri. Untuk mengetahui beberapa informasi pada sebuah citra kita harus menafsirkan sebuah titik fitur lokal yang terkait dari citra tersebut. Pada dasarnya titik kunci (*key point*), titik ketertarikan (*interest point*) dan titik fitur (*fiture point*) merupakan himpunan titik-titik tunggal (*singular point*) yang dimiliki oleh citra dan sulit dibedakan. Proses penentuan titik-titik tunggal (s*ingular points*) dapat dicapai dengan menggunakan detektor dan deskriptor. Deteksi adalah metode ekstransi titik-titik tunggal dari sebuah citra. Sementara detektor menyediakan invarian untuk menemukan titik-titik penting yang sama terkait perubahan-perubahan. Deskriptor melakukan proses identifikasi titik-titik tunggal yang membedakannya dari sisa himpunan titik-titik tunggal. Pada gilirannya, deskriptor harus memastikan invarian menemukan korespondensi antara titik-titik tertentu terkait perubahan-perubahan tersebut.

Deteksi sudut (*corner detection*) merupakan metode dasar yang digunakan untuk menentukan titik sudut dalam sebuah citra dua dimensi (*x*, *y*). Dimana titik sudut tersebut nantinya dapat digunakan dalam penentuan jarak antara titik yang banyak dan berguna dalam proses rekonstruksi citra tiga dimensi (*x*, *y*, *z*), selain itu penggunaan deteksi sudut terus berkembang dan banyak diterapkan dalam bidang komputer visi dan pengolahan citra seperti; 1) *Stereo Matching*, 2) *Image Registration*, 3) *Stitching of Panoramic Photographs*, 4) *Object Detection*, 5) *Object Recognition*, 6) *Motion Tracking*, 7) *Robot Navigation*, dan 8) *feature matching*.

Pada dasarnya sudut sama dengan *interest point*. Banyak detektor *interest point* yang dijadikan referensi dengan berbagai definisi yang berbeda terkait fokus terhadap titik-titik dalam sebuah citra. Beberapa detektor menemukan titik-titik simetri lokal, detektor lain menemukan berbagai variasi tekstur dan ada juga detektor yang menemukan lokasi dari titik-titik sudut. Beberapa jenis sudut (*L-Junction*, *Y-Junction*, *T-Junction*, *Arrow-Junction*, dan *X-Junction*) seperti digambarkan pada gambar 1, jenis-jenis sudut inilah yang akan disimulasikan guna mengetahui kinerja dan respon dari masing-masing detektor sudut. Dalam penelitian ini citra yang digunakan akan diberikan jenis *noise* tertentu sehingga sudut pada citra memiliki nilai keabuan dan intensitas yang berbeda.



Gambar 1. Ilustrasi *L-Junction*, *Y-Junction*, *T-Junction*, *Arrow-Junction* dan *X-Junction*

Penggunaan detektor sudut (*corner detection*) dan titik ketertarikan (*interest point*) untuk menemukan informasi terkait dengan titik-titik yang sesuai di dalam sebuah citra merupakan langkah kunci dalam banyak aplikasi pada bidang pengolahan citra dan komputer visi. Dalam penelitian ini algoritma pendeteksi sudut yang digunakan dan akan dibandingkan adalah algoritma *Moravec*, *Susan*, *Harris*, *FAST*, *Eigen* and *Forstner*. Citra yang digunakan sebagai masukan adalah citra dengan format *grayscale* dan diberikan beberapa jenis *noise*.

Dalam penelitian ini *noise* digunakan untuk mengetahui performa dan sesitifitas dari masing-masing algoritma detektor sudut yang akan di uji. *Noise* akan memberikan pengaruh pada intensitas citra dan jenis *noise* yang digunakan adalah; 1) *noise gaussian*, 2) *noise*

poisson, 3) *noise salt & pepper* dan 4) *noise speckle* dimana kesemua jenis *noise* yang digunakan akan diberikan parameter-parameter numerik tertentu.

Penelitian ini bertujuan untuk menganalisa kinerja dari keenam detektor sudut dan fokus komparasi akan dilihat dari beberapa kriteria (Tuytelaars and Mikolajczyk, 2006) berikut; 1) algoritma mampu menemukan keseluruhan sudut dengan tepat, 2) algoritma mampu menunjukkan lokasi sudut dengan akurat, 3) algoritma memiliki kestabilan yang baik terhadap perulangan, 4) algoritma memiliki kestabilan yang baik terhadap *noise*, 5) algoritma efisien dalam kopleksitas perhitungan dan 6) algoritma memiliki waktu eksekusi yang singkat.

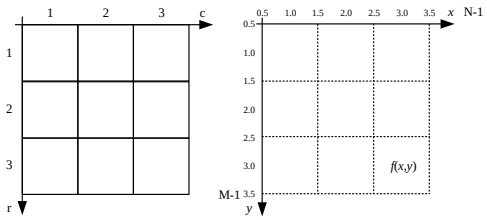
II. KAJIAN PUSTAKA

2.1. Defenisi Citra

Citra dapat didefinisikan sebagai gambar pada bidang dua dimensi (2-D), secara matematis citra merupakan fungsi kontinyu dari intensitas cahaya pada bidang dua dimensi. Citra merupakan representasi, kemiripan, atau tiruan dari suatu objek atau benda. Citra juga dapat kita artikan sebagai suatu bentuk informasi visual, dimana citra memiliki karakteristik atau keunikan yang berbeda bahkan tidak dimiliki oleh data teks, dimana citra memiliki informasi yang lebih kompleks jika dibandingkan dengan data teks. Terbentuknya citra adalah hasil dari proses penangkapan pantulan sebahagian berkas cahaya yang berasal dari sebuah objek yang ditangkap oleh peralatan optik, misalanya mata manusia, kamera, alat pemindahan (*scanner*) dan lain-lain yang kemudian bayangan objek atau yang disebut citra tersebut terekam. Sebuah citra merupakan fungsi dua dimensi fungsi $f(x, y)$, dimana x dan y adalah kordinat spasial, dan amplitudo f pada setiap pasangan kordinat (x, y) yang disebut tingkat intensitas citra.

2.2. Citra Digital

Jika nilai-nilai x, y dan amplitudo f adalah terbatas dan berbentuk diskrit, ini kita sebut dengan citra digital. Sebuah citra digital memiliki elemen dengan jumlah tertentu yang disebut piksel, yang masing-masing memiliki lokasi dan nilai tertentu. Piksel merupakan satuan terkecil dari sebuah citra yang menempati suatu posisi yang menentukan resolusi citra tersebut. Citra digital berbentuk matriks dengan ukuran $M \times N$, dimana M menyatakan tinggi dan N menyatakan lebar dari citra (gambar 2).



Gambar 2.1. Kordinat Piksel dan Matriks Citra (M x N)

2.3. *Graysacale*

Citra *grayscale* adalah citra yang memiliki nilai matriks yang direpresentasikan dengan nilai intesitas dalam rentang tertentu. Format *grayscale* memiliki rentang gradasi warna piksel hitam dan putih. *Grayscale* merupakan format unik terletak pada gradasi warna hitam dengan intensitas terlemah menuju warna putih dengan intensitas yang paling kuat. Walaupun

demikian citra dengan format *grayscale* tentu saja tidak hanya terdiri dari warna hitam dan putih saja, terdapat derajat keabuan bagian tengahnya.

2.4. *Noise* (derau)

Noise pada citra adalah variasi acak (tidak berasal dalam objek foto) yang merupakan informasi kecerahan atau warna dalam sebuah citra, dan biasanya merupakan aspek *noise* elektronis. Hal ini dapat dihasilkan oleh gangguan optik (sensor atau sirkuit) seperti; *scanner* atau kamera digital maupun secara disengaja akibat proses pengolahan yang tidak sesuai atau proses yang kurang sempurna. *Noise* dapat juga berbentuk suara dan *film grain*, derau tidak dapat dihindari olek detektor foton yang ideal. *Noise* citra merupakan hal yang tidak dinginkan karena dapat mengurangi kualitas dan keakuratan informasi sebuah citra.

2.5. Detektor Sudut Moravec

Studi terkait detektor titik pada citra dimulai oleh Moravec pada tahun 1977. Moravec mendefinisikan konsep titik-titik keterterikan (*interest points*) sebagai daerah yang berbeda pada sebuah citra dan menyimpulkan *interest points* ini dapat digunakan untuk menemukan kecocokan area dalam bingkai citra yang berurutan. Detektor Moravec, menggunakan operator sederhana dengan konsep fokus pada perbedaan kecerahan (*brightness*) pada jendela persegi yang dilambangkan *W* dengan ukuran relatif (biasanya 3x3, 5x5, 7x7 piksel) melalui pergeseran kotak *W* sebesar satu piksel di delapan arah (horizontal, vertikal dan diagonal).

2.6. Detektor Sudut Susan

Algoritma Susan (*smallest univalue segment assimilation nucleus*) diusulkan oleh Smith dan Brady pada tahu, 1997) dengan konsep segmentasi area pada citra. Sudut ditentukan oleh segmentasi melingkar di lingkungan yang sama dan area berbeda. Sudut-sudut yang berada di daerah relatif pada tempat serupa (mirip USAN) mencapai lokal minimum di bawah ambang batas (*threshold*) tertentu. Dimana pusat dari piksel disebut dengan inti (*kernel*) yang merupakan nilai intensitas tercatat. Dengan pengertian lain bahwa area akan dibagi menjadi dua kategori; yang memiliki intensitas sama dengan inti atau yang memiliki intensitas berbeda dengan inti.

2.7. Detektor Sudut Harris

Operator ini dikembangkan oleh Chris Harris dan Mike Stephens pada tahun 1988, sebagai langkah pengolahan tingkat rendah untuk membantu membangun penafsiran para peneliti di lingkungan robotika berdasarkan urutan citra. Secara khusus, Harris dan Stephens tertarik menggunakan teknik analisis gerak untuk menafsirkan lingkungan berdasarkan gambar dari sebuah kamera ponsel. Seperti Moravec, mereka membutuhkan satu metode untuk mencocokan korespondensi setiap titik yang sesuai pada urutan *frame* citra, yang pada prosesnya mereka juga tertarik dalam pendeteksian sudut dan tepi antara *frame-frame* citra. Harris dan Stephens mengembangkannya menjadi detektor gabungan sudut dan tepi dengan tujuan untuk mengatasi keterbatasan operator Moravec.

2.8. Detektor Sudut FAST

Algoritma Detektor sudut FAST (*features from accelerated segment test*) pertama kali diusulkan oleh Rosten dan Drummond pada tahun 2005. Algoritma ini meningkatkan performa pengujian segmen menggunakan lingkaran yang terdiri 16 piksel (adopsi dari goritma Bresenham) di sekitar titik calon (*candidate point*) dari *P*. Titik yang merupakan sebuah sudut, jika diasumsikan dengan titik *P* dan memiliki *N* piksel yang berdekatan pada lingkaran,

intensitas yang lebih besar dari *I* adalah (*P* + *t*) atau yang lebih kecil dari intensitas *I* adalah (*P* - *t*), di mana *I* adalah intensitas, dan *t* adalah *threshold*. Selanjutnya, membandingkan intensitas di lokasi vertikal dan horizontal pada lingkaran nomor 1, 5, 9 dan 13 dengan intensitas pada titik P (untuk menjadi kandidat titik sudut palsu). Titik-titik ini akan dieksekusi dengan tiga kondisi yaitu; *I* dari *Pi* > *I* dari *P* + *t* atau *I* dari *Pi* < *I* dari *P* + *t*, *i* = 1,..., 4, pengujian dilakukan pada semua 16 titik-titik yang ada.

2.9. Detektor Sudut Eigen

Detektor sudut sudut Shi-Tomasi dikembangkan oleh Shi-Tomasi dan Kanade-Tomasi, 1993 sebagian besar merujuk dan menggunakan detektor Harris, tetapi berbeda dalam perhitungan respon, dimana operator algoritma menghitung nilai min(λ_1, λ_1) ini diasumsikan bahwa pencaraian sudut lebih stabil. Ini menggunakan persamaan yang sama untuk menganalisis aliran optik dari Kanade Lucas.

Nilai κ harus ditentukan secara empiris, dan dalam nilai-nilai berdasarkan literatur di kisaran 0,04-0,15 dinyatakan baik. Dalam catatan Harris dan Stephens menyatakan bahwa perhitungan yang tepat dari nilai eigen adalah komputasi rumit, karena memerlukan perhitungan akar kuadrat, dan tidak menyarankan fungsi M_c di mana κ adalah parameter sensitivitas (*tunable sensitivity*).

2.10. Detektor Sudut Forstner

Forstner dan Gulch pada tahun 1987, pada dasarnya algoritma ini menggunakan metode angularitas yang sama dengan detektor Harris. Menggunakan implementasi komputasi yang lebih kompleks. Operator ini menggunakan fungsi auto korelasi untuk mengklasifikasikan piksel dalam kategori (*interest points, edges or region*). Tahapan deteksi dan penentuan lokasi dipisahkan, dalam pemilihan jendela, di mana setiap fitur dan lokasi fitur yang diketahui berada dalam jendela yang dipilih. Selanjutnya penggunaan statistik yang memungkinkan perkiraan *threshold* secara otomatis untuk proses klasifikasi.

III. METODE PENELITIAN

Metodologi penelitian yang dilakukan meliputi studi literatur, rancangan aplikasi simulasi, implementasi, pengujian, dan dokumentasi serta penjelasan dari masing-masing detektor sudut yang akan diuji.

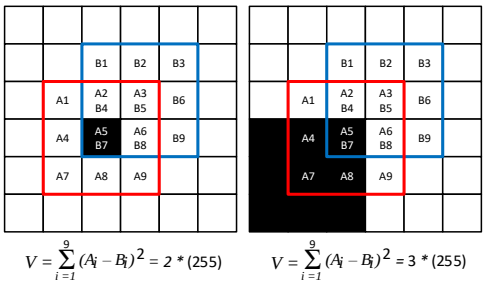
3.1. Studi literatur

Studi literatur merupakan tahap awal penelitian yang dimulai dengan mengumpulkan berbagai referensi yang berhubungan dengan citra, pengolahan citra, cara kerja dan penggunaan algoritma detekto sudut Moravec, Susan, Harris, FAST, Eigen dan Forstner.

3.1.1. Detektor Sudut Moravec

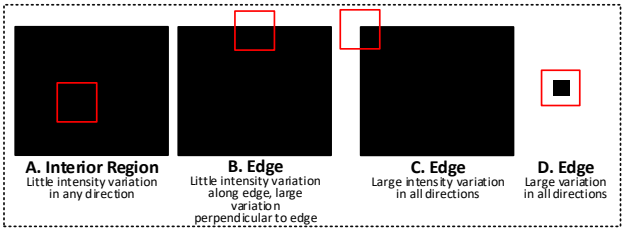
Detektor ini dikembangakan pertama kali oleh Hans. P. Moravec pada tahun 1977, dengan memperkenalkan konsep *interest point*. Operator ini mempertimbangkan jendela lokal (biasanya 3x3, 5x5, 7x7 piksel) dalam citra dengan titik pusat *P* dan menentukan perubahan rata-rata intensitas yang dihasilkan dari pergeseran jendela dengan jumlah yang kecil dalam beberapa arah, dimana variasi intensitas untuk satu pergeseran di hitung dengan mengambil penjumlahan perbedaan intensitas dari setiap piksel dari dua jendela (gambar 3.1). Operasi ini diulang untuk setiap posisi piksel yang tentukan dengan *interest value* yang sama dengan perubahan minimum yang dihasilkan oleh pergeseran tersebut. Titik ketertarikan (*interest point*) adalah nilai lokal maksima dari titik ketertarikan. Moravec menerapkan pendekatan ini

dengan prinsip menghitung autokorelasi lokal tidak normal pada delapan arah (arah horizontal, arah vertikal dan empat arah diagonal) yang menghasilkan respon anisotropik. Respon ini seperti variasi intensitas yang akan hanya dihitung pada pergeseran diskrit.



Gambar 3.1. Perhitungan Variasi Intensitas Jendela 3x3 pada Arah Diagonal Kanan Atas

Variasi intensitas dari satu pergeseran dihitung dengan mengambil jumlah nilai pangkat dua dari perbedaan intensitas piksel-piksel yang berhubungan pada dua jendela. Dalam gambar 3.1, perhitungan untuk satu pergeseran diagonal pada isolasi piksel warna hitam nilai intensitas sama dengan nol (0) dan pada latar belakang (*background*) nilai intensitas 255 serta pada sebuah sudut ideal. Kotak dengan warna merah menandakan jendela awal (*original window*) dan kotak berwarna biru menandakan pergeseran jendela. Variasi intensitas pada *P* merupakan intensitas minimum yang dihitung dengan merujuk pada delapan arah pergeseran (arah horizontal, arah vertikal dan empat arah diagonal).



Gambar 3.2. Ilustrasi Jendela Operator Moravec pada Posisi yang Berbeda

Petimbangan arah pada operator Moravec yang diilustrasikan pada gambar 3.2, terlihat jendela pada empat posisi yang berbeda. Posisi A, merupakan interior atau objek latar belakang yang di asumsikan bahwa intensitas relatif konstan di dalam jendela, maka hasil pergeseran jendela pada beberapa arah terjadi hanya pada variasi intesitas yang kecil. Pada posisi B, pergeseran jendela secara tegak lurus di bagian tepi pada variasi intensitas yang besar, tetapi sepanjang tepi akan menghasilkan variasi intensitas yang kecil. Kedua posisi C dan D masing-masing berhubungan antara sebuah sudut dan piksel yang diisolasi akan memberikan variasi intensitas pada semua arah pergeseran. Ini menandakan operator Moravec merupakan *corner detector*. Sebagai catatan, variasi intensitas yang besar pada setiap arah adalah setara dengan arah gerakan yang diberikan variasi intensitas minimum menjadi besar.

Operator Moravec dapat digunakan untuk memberikan pengukuran dari sudut setiap piksel dalam sebuah citra. Pengukuran ini adalah nilai intensitas minimum yang didapatkan

melalui delapan pergeseran. Penggunaan operator ini untuk setiap piksel dalam sebuah citra akan membentuk peta titik sudut. Gambar 3.3, memperlihatkan peta titik sudut untuk citra sederhana degngan menggukan sebuah jendela 3x3. Peta ini menggambarkan beberapa titik-titik penting dengan nilai sudut yang kesemuannya dibagi dengan $(255)^2$ dan nilai aktual dari 2 menghasilkan nilai $2x(255)^2$, dimana sudut merupakan lokal maksima, nilai dari piksel yang diisolasi dalam peta sudut adalah sama dengan nilai sudut pada peta sudut dan terdapat sebuah area pembatas disekeliling citra pada saat operator ini gagal menentukan secara langsung ditandai dengan x.

x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
x	x	0	0	0	0	0	0	0	0	0	1	1	1	x	x
x	x	0	0	0	0	0	1	1	0	0	1	2	1	x	x
x	x	0	0	0	0	0	2	1	0	0	1	1	1	x	x
x	x	0	0	0	0	0	0	0	0	0	0	0	0	x	x
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

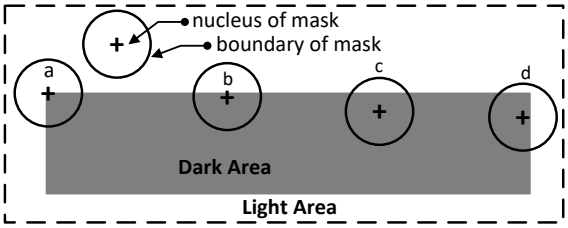
Gambar 3.3. Peta Sudut Operator Moravec

Titik pertama mengindikasikan bahwa sudut tersebut adalah lokal maksima dalam peta sudut, lokal maksima dapat kita tentukan dengan *non maximal supperession*. Adapun piksel-piksel yang diisolasi menandakan bawa titik-titik ini merupakan sudut. Untuk alasan ini detektor sudut Moravec menjadi sesitif terhadap *noise*. Menggunakan ukuran jendela yang besar akan menjadikan algoritma ini lebih fokus menjadikan *noise* sebagai titik sudut, walaupun piksel-piksel yang terisolasi akan tetap menjadi lokal maksima. Dengan pengertian lain semakin tinggi tingkat *noise* pada citra akan menyebabkan semakin banyak pula lokal maksima yang tidak saling bekoresponden terhadap titik-titik sudut. Solusis masalah ini adalah menggunakan pengaturan nilai ambang (*threshold*) dengan nilai *threshold* sama dengan nol (0). Dalam aplikasinya jika operator Moravec gagal dalam satu pergeseran jendela atau lebih mudah akan mengatur nilai menjadi nol (0).

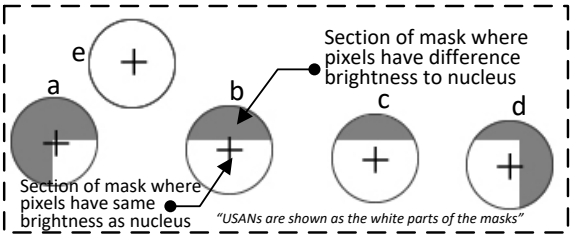
Intensitas citra dari piksel di (x, y) ditandai oleh I (x, y), input citra dengan format *grayscale*, ukuran jendela (*window scale*) dan nilai ambang (*threshold*) T, Semua nilai bukan nol (0) yang ada di dalam peta sudut adalah sudut.

3.1.2. Detektor Sudut Susan

Detektor sudut Susan (*smallest univalue segment assimilating nucleus*), dikembangkan oleh S.M. Smith dan M. Brady. Detektor ini bekerja dengan menggunakan masker melingkar (berbentuk lingkaran). Jika kecerahan dari setiap piksel dalam masker dibandingkan dengan kecerahan pusat masker maka kemudian wilayah masker dapat didefinisikan memiliki kecerahan yang sama (atau serupa) sebagai inti. Dengan pengertian lain algoritma Susan mengasumsikan bahwa di dalam area lingkaran piksel-piksel yang relatif kecil milik objek tertentu akan memiliki kecerahan yang relatif seragam. Algoritma menghitung jumlah piksel dengan tingkat kecerahan mirip dengan piksel pada pusat masker (inti dari masker).



Gambar 3.4. Ilustrasi Empat Masker Lingkaran pada Citra di Lokasi yang Berbeda



Gambar 3.5. Ilustrasi Empat Masker Lingkaran dengan Kesamaan Warna

Daerah masker ini dikenal sebagai USAN, (*univalue Segmen asimilasi Nucleus*). Terlihat pada gambar 3.4, menunjukkan sebuah persegi panjang warna gelap dengan latar belakang putih, lima masker melingkar ditunjukkan pada posisi yang berbeda pada sebuah citra. Sudut dapat dideteksi sesuai dengan bidang USAN. Pusat terletak pada sudut ketika daerah USAN di lokasi sembarang denga nilai terkecil, seperti posisi "a" (gambar 3.5). Dalam rangka untuk mendeteksi sudut, fungsi perbandingan yang sama antara masing-masing piksel dalam masker dan pusat masker ini dihitung dengan menggunakan persamaan di bawah ini.

$$c(r, r_0) = \begin{cases} 1, |I(r) - I(r_0)| \leq t \\ 0, \quad other\ wise \end{cases} \tag{1}$$

Dimana r_0 adalah kordinat *nucleus* dan r adalah kordinat dari titik-titk lain di dalam masker, $c(r, r_0)$ adalah hasil perbandingan. Sementara $I(r)$ adalah titik nilai abu-abu, t adalah ambang batas (*threshold*) perbedaan abu-abu dengan menentukan kemampuan pengurangan *noise* dan memperkecil nilai kontras yang dapat dideteksi oleh detektor Susan. Dalam prakteknya persamaan (1) tidak stabil dan disempurnakan lagi dengan persamaan (2) sebagai berikut.

$$c(r, r_0) = e^{\left\{-\left[\left[\frac{I(r)-I(r_0)}{t}\right]^6\right]\right\}} \tag{2}$$

Ukuran dari area USAN dapat dihitung dengan menggukanan persamaan berikut ini.

$$n(r_0) = \sum_{r \in c(r_0)} c(r, r_0) \tag{3}$$

Mengetahui respon awal terhadap sudut dapat dihitung dengan menggukan persamaan (4) di bawah ini. Dan sesuai dengan konsep Susan, bahwa area USAN terkecil, merupakan respon awal terbesar terhadap sudut.

$$R(r_0) = \begin{cases} g - n(r_0), nr \leq g \\ 0, \qquad\quad \geq g \end{cases} \tag{4}$$

Pada persamaan (4), *g* adalah *threshold* geometrik dengan menemukan level keruncingan sebuah sudut dan keruncingan terkecil. Hal ini meningkatkan informasi sudut pada citra. Akhirnya, sudut dapat ditemukan dengan mempertahankan non maksima.

3.1.3. Detektor Sudut Harris

Detektor ini dikembangakan pertama kali oleh Chris Harris dan Mike Stephens pada tahun 1988, konsep operatornya dikenal dengan operator Plessey, terinspirasi dari operator Moravec, dimana Harris dan Stephens menggabungkan deteksi sudut dan tepi.

Detektor sudut Harris merupakan operator *interest point* yang sering digunakan karena invarian yang kuat terhadap rotasi, skala, variasi iluminasi dan invarian terhadap *noise*. Detektor sudut ini didasarkan pada fungsi autokorelasi sinyal lokal di mana fungsi autokorelasi lokal mengukur perubahan lokal dari sinyal dengan jejak pergeseran dalam jumlah kecil untuk arah yang berbeda, dimana pergeseran (Δ*x,* Δ*y*) dan titik (*x*, *y*) maka fungsi autokorelasi didefinisikan sebagai,

$$C(x,y) = \sum_{x,y} w[I(xi,yi) - I(xi + \Delta x, yi + \Delta y)]^2 \tag{5}$$

dimana (·, ·) menunjukkan fungsi citra dan (*xi*, *yi*) adalah titik di jendela *w* (*gaussian*) yang berpusat pada (*x*, *y*), dimana pergeseran citra didekati dengan ekspansi Taylor dipotong dengan persyaratan urutan pertama,

$$I(xi + \Delta x, yi + \Delta y) = I(xi,yi) + \begin{bmatrix} I_x(xi,yi) & I_y(xi,yi) \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \tag{6}$$

dimana *Ix* (*xi*, *yi*) dan *Iy* (*xi*, *yi*) menunjukkan derivatif parsial masing-masing dalam *x* dan *y*. Dengan *filter* seperti [-1, 0, 1] dan [-1, 0, 1]^T, derivatif parsial citra dapat dihitung dengan mensubsitusi persamaan (6) ke dalam persamaan (5) menghasilkan,

$$C(x,y) = [\Delta x \; \Delta y] \begin{bmatrix} \sum_w (I(xi,yi))^2 & \sum_w I_x(xi,yi)\, I_y(xi,yi) \\ \sum_w I_x(xi,yi)\, I_y(xi,yi) & \sum_w (I(xi,yi))^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

$$= [\Delta x \; \Delta y]\, C(x,y) \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{7}$$

C(*x, y*) adalah matriks autokorelasi yang mengambil struktur intensitas area lokal. Kemudian *λ1* dan *λ2* adalah nilai *eigen* dari *C*(*x*, *y*), maka kita memiliki tiga pertimbangan sebagai berikut; 1) kedua nilai eigen adalah bagian kecil area homogen (intensitas konstan), 2) kedua nilai eigen yang tinggi berarti merupakan titik ketertarikan sudut (*corner*), dan 3) salah satu nilai *eigen* tinggi berarti bagian tepi (*edge*).

Operator Harris ditujukan sebagai formulasi ulang dari teknik pengukuran peta sudut dengan mempertimbangkan variasi antara pengukuran intensitas pada setiap perbedaan arah. Formulasi dapat di lihat pada persamaan di bawah ini.

$$V_{u,y}(x,y) = Au^2 + Cuv + Bv^2 = [u \quad v]\, M \begin{bmatrix} u \\ v \end{bmatrix} \tag{8}$$

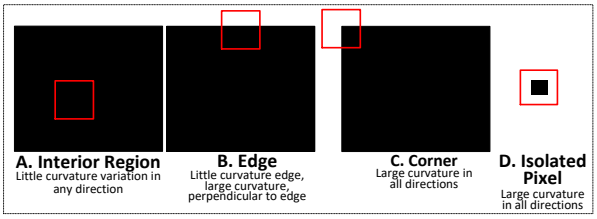
Dimana *M* adalah,

$$M = \begin{bmatrix} A & C \\ C & B \end{bmatrix}, \text{A} = \left(\frac{\partial I}{\partial x}\right)^2 \otimes w, \text{B} = \left(\frac{\partial I}{\partial y}\right)^2 \otimes w, \text{C} = \left(\frac{\partial I}{\partial y}\frac{\partial I}{\partial y}\right)^2 \otimes w \quad (9)$$

dan,

w = Jendela Gaussian
$\otimes$ = Operator Konvulasi

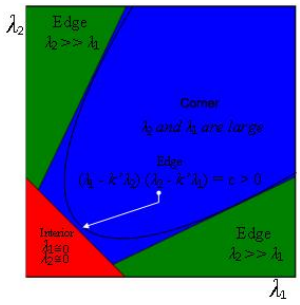
Matriks *M* memiliki semua operator diferensial yang menjelaskan permukaan citra dengan memberikan titik (*x*, *y*). nilai eigen dari *M* akan proporsional untuk konsep kurva dari permukaan dan bentuk sebuah citra.



Gambar 3.6. Ilustrasi Jendela Operator Plessey pada Posisi yang Berbeda

Posisi ini jendela yang berbeda ditunjukkan pada gambar 3.6. Posisi A adalah interior untuk sebuah objek (latar belakang), diasumsikan intensitas citra akan relatif konstan dalam jendela. Karena ada sedikit lengkungan di permukaan dalam jendela ini kedua nilai eigen akan relatif kecil. Untuk jendela lokal diantara tepi, seperti pada posisi B, ada kelengkungan signifikan tegak lurus ke tepi dan sangat sedikit lekukan di sepanjang tepi sehingga salah satu nilai eigen akan besar dan kecil pada lainnya. Kedua posisi C dan D, yang sesuai dengan sudut dan piksel terisolasi, akan memiliki kelengkungan signifikan di kedua arah sehingga kedua nilai eigen akan besar.

Nilai eigen dari *M* dilambangkan dengan λ1 dan λ2. Analisis di atas menunjukkan bidang yang dijelaskan oleh λ1 dan λ2 dapat dibagi menjadi 3 daerah yang berbeda seperti yang ditunjukkan pada Gambar 3.7.



Gambar 3.7. *Division of Eigenvalue Space Into Distinct Feature Regions*

Untuk mengklasifikasikan piksel, tiga garis pemisah bidang pada gambar 3.7, harus dipilih. Hal ini berguna untuk memiliki ukuran peta sudut bukan hanya label diskrit untuk setiap piksel. Ukuran peta sudut memungkinkan bagian tepi untuk menipis dan meningkatkan

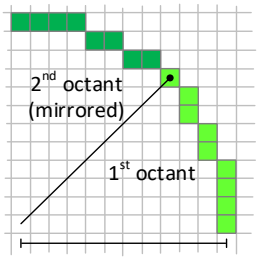
penentuan lokasi sudut dengan mengambil lokal maxima untuk menjadi posisi yang benar dari sudut atau tepi. Harris dan Stephens mengusulkan ukuran peta sudut berikut:

$$C(x,y) = \det(M) - k(\mathrm{trace}(M))^2 = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2 \quad (10)$$

Dimana, $\det(M) = \lambda_1\lambda_2 = AB - C^2$, $\mathrm{trace}(M) = \lambda_1 + \lambda_2 = A + B$, dan $\mathrm{k} = constan$

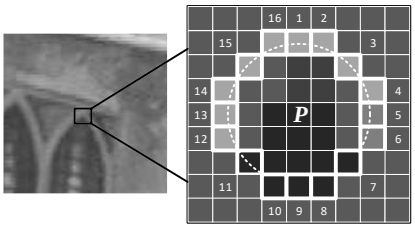
3.1.4. Detektor Sudut FAST

Algoritma Detektor sudut FAST (*Features from Accelerated Segment Test*) pertama kali diusulkan oleh Rosten dan Drummond pada tahun 2005, dan cukup sukses untuk mengidentifikasi *interest point* dalam citra. *Interest point* pada citra adalah piksel yang memiliki posisi yang jelas dan dapat dideteksi. *Interest point* kaya akan konten dan informasi lokal dan secara ideal dapat diulang pada citra yang berbeda. Algoritma ini menggunakan sebuah lingkaran yang terdiri dari 16 piksel menggunakan algoritma Bresenham (gambar 3.8) diterapkan pada pengujian segmentasi.



Gambar 3.8. *Rasterisation of a Circle by the Bresenham Algorithm*

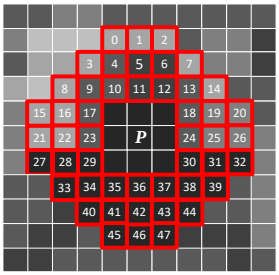
Titik yang merupakan sebuah sudut, jika diasumsikan dengan titik *P* dan memiliki *N* piksel yang berdekatan pada lingkaran, intensitas yang lebih besar dari *I* adalah (*P* + *t*) atau yang lebih kecil dari intensitas *I* adalah (*P* - *t*), di mana *I* adalah intensitas, dan *t* adalah *threshold*. Selanjutnya, membandingkan intensitas di lokasi vertikal dan horizontal pada lingkaran nomor 1, 5, 9 dan 13 dengan intensitas pada titik *P* (untuk menjadi kandidat titik sudut palsu). Jika 3 dari titik-titik ini akan dieksekusi kondisi *I* dari *Pi* > *I* dari *P* + *t* atau *I* dari *Pi* < *I* dari *P* + *t*, *i* = 1, .., 4, pengujian dilakukan pada semua 16 titik-titik yang ada.



Gambar 3.9. Test Segmentasi 12 Titik Pendeteksian Sudut pada Jejak Citra

Percobaan telah menunjukkan bahwa nilai terkecil dari *N*, di mana titik-titik singular mulai muncul secara konsisten pada N = 9 awalnya, algoritma asli adalah FAST-12. Ada modifikasi dari algoritma pohon FAST-9 dan FAST-12 (berdasarkan FAST-9 dan FAST-12), algoritma yang asli memiliki sejumlah kelemahan, misalnya, di dekat lingkungan mungkin

ditemukan beberapa titik singular, efisiensi algoritma tergantung pada urutan pengolahan citra dan distribusi piksel. Pada tahun 2008 Edward Rosten, Reid Porter dan Tom Drummond memperkenalkan perbaikan pada algoritma FAST, dalam hal penentuan titik-titik penting. Algoritma ini disebut FAST-ER (*ER - enhanced repeatability, improved repeatability*). Algoritma stabil untuk properti dari pengulangan; di satu dan dalam citra yang sama dari sudut yang berbeda, ada titik-titik khusus milik objek yang sama. Dalam algoritma ini menggunakan lingkaran cincin dari 1 piksel, bukan di FAST yang menggunakan 48 piksel (gambar 3.10). Proses klasifikasi titik-titik kunci (dugaan titik-titik kandidat) dengan algoritma ID3 menggunakan pohon keputusan.



Gambar 3.10. Posisi *Offset FAST-ER Detector*

Algoritma ID3 mengoptimalkan level proses setiap piksel yang menghasilkan detektor yang lebih efisien, fungsi ini dihitung dengan *R* adalah pengukuran iterasi, *N* adalah jumlah titik-titik khusus yang terdeteksi, dan *S* adalah *node* pada pohon keputusan. Secara detail dapat dilihat dari persaman (11).

$$cost = (k_R + R^{-2})(k_N + N^{-2})(k_S + S^{-2}) \quad (11)$$

3.1.5. Detektor Sudut Eigen

Detektor sudut Shi-Tomasi dikembangkan oleh Shi-Tomasi dan Kanade-Tomasi, 1993 sebagian besar merujuk dan menggunakan detektor Harris, tetapi berbeda dalam perhitungan respon, dimana algoritma menghitung nilai min(λ_1, λ_1) dengan asumsikan bahwa pencarian sudut lebih stabil. Ini menggunakan persamaan yang sama untuk menganalisis aliran optik dan Kanade Lucas. Nilai κ harus ditentukan secara empiris, dan dalam nilai-nilai berdasarkan literatur di kisaran 0,04-0,15 dinyatakan baik. Dalam catatan Harris dan Stephens menyatakan bahwa perhitungan yang tepat dari nilai eigen adalah komputasi rumit, karena memerlukan perhitungan akar kuadrat, dan tidak menyarankan fungsi M_c persamaan (12) di mana κ adalah parameter sensitivitas (*tunable sensitivity*):

$$M_c = \lambda_1\lambda_2 - \kappa(\lambda_1 + \lambda_2)^2 = \det(A) - \kappa \operatorname{trace}^2(A) \quad (12)$$

Untuk dapat menghindari pengaturan parameter κ tersebut, Shi-Tomasi menggunakan pengukuran sudut Noble (M_c') dengan penjumlahan rata-rata harmonik nilai eigen:

$$M_c' = 2\frac{\det(A)}{trace(A) + \epsilon'} \quad (13)$$

Dimana ϵ = konstanta positif bernilai kecil
Matriks kovarian dari posisi sudut adalah A^{-1}, yaitu

$$\frac{1}{\langle I_x^2\rangle\langle I_y^2\rangle-\langle I_x I_y\rangle^2}\begin{bmatrix}\langle I_y^2\rangle & -\langle I_x I_y\rangle\\ -\langle I_x I_y\rangle & \langle I_x^2\rangle\end{bmatrix} \quad (14)$$

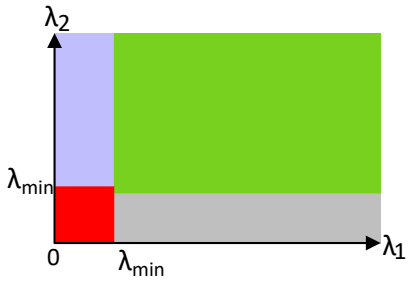
Detektor sudut Shi-Tomasi didasarkan sepenuhnya pada sudut detektor Harris. Namun, salah satu variasi kecil dalam seleksi kriteria membuat detektor ini jauh lebih baik daripada yang asli. Bekerja cukup baik di mana bahkan sudut detektor Harris gagal. Jadi, inilah perubahan minor yang Shi-Tomasi lakukan untuk detektor sudut Harris yang asli. Detektor sudut Harris memiliki kriteria seleksi sudut. Skor *A* dihitung untuk setiap piksel, dan jika skor di atas nilai tertentu, piksel ditandai sebagai sudut. Skor tersebut dihitung dengan menggunakan dua nilai eigen. Artinya, kita akan memberi dua nilai eigen ke dalam fungsi. Fungsi memanipulasi keduanya dan memberi skor kembali. Dengan pengertian lain bahwa fungsi nilai-nilai *eigen* harus digunakan untuk memeriksa apakah piksel tersebut merupakan sudut atau tidak. Skor untuk detektor sudut Harris dihitung seperti ini (*R* adalah skor):

$$R = \det(M) - k(\text{trace}(M))^2 \quad (15)$$
$$\det(M) = \lambda_1\lambda_2 \quad (16)$$
$$\text{trace}(M) = \lambda_1 + \lambda_2 \quad (17)$$

Sementara penentuan peta sudut $R(x_p, y_p)$ dari titik *p* pada operator Shi-Tomasi dapat dihitung dengan persamaan berikut.

$$R = min(|\lambda_1|, |\lambda_2|) \quad (18)$$

Dalam tulisan mereka, Shi-Tomasi menunjukkan eksperimen bahwa kriteria skor ini jauh lebih baik. Jika *R* lebih besar dari nilai standar tertentu, dapat ditandai sebagai sudut. Dengan demikian, wilayah berlaku untuk titik yang menjadi sudut dapat dilustrasikan gambar 3.11, sebagai berikut.



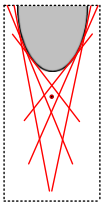
Gambar 3.11. Area λ_1 atau λ_2

- Area hijau, baik λ_1 dan λ_2 lebih besar dari nilai tertentu. Dengan demikian, daerah ini adalah untuk piksel "diterima" sebagai sudut.
- Area biru dan abu-abu, baik λ_1 atau λ_2 kurang dari nilai minimum yang diperlukan.
- Areah merah, baik λ_1 dan λ_2 kurang dari minimum yang diperlukan. Bandingkan di atas dengan grafik yang sama untuk detektor sudut Harris. Maka akan terlihat daerah biru dan abu-abu adalah setara dengan ujung area. Area warna merah adalah untuk bidang datar, area warna hijau adalah untuk sudut. Detektor sudut

Shi-Tomasi lebih lengkap dari detektor sudut Harris, algoritma deteksi sudut Shi-Tomasi banyak digunakan pada OpenCV (*Open Source Computer Vision*).

3.1.6. Detektor Sudut Forstner

Operator Forstner diaplikasikan oleh Forstner tahun 1986, diperbaharui oleh Forstner dan Gulch tahun 1987, operator ini menggunakan fungsi auto-korelasi untuk mengklasifikasikan piksel dalam kategori (*interest points, edges or region*). Tahapan deteksi dan penentuan lokasi dipisahkan, dalam pemilihan jendela, di mana setiap fitur dan lokasi fitur yang diketahui berada dalam jendela yang dipilih. Selanjutnya penggunaan statistik lokal yang memungkinkan perkiraan *threshold* secara otomatis untuk proses klasifikasi. Algoritma ini dapat dioperasikan dengan pembobotan optimal di tengah jendela sehingga akurasi jauh lebih tinggi. Kelemahan dari algoritma ini adalah implementasi yang relatif rumit, sensitif terhadap pencahayaan dan kontras pada citra. Dalam prakteknya algoritma bergantung pada sudut yang ideal dan garis singgung lintas pada satu titik.



Gambar 3.12. *Corner Detection of Forstner Algorithm*

Operator Forstner menggunakan ukuran *cornerness* yang sama dengan operator Harris (meskipun, bagaimana ukuran ini dihitung sangat berbeda) dan menggunakan statistik lokal untuk menghitung pilihan *threshold*. Hasilnya adalah lebih baik pada peningkatan perhitungan penentuan lokasi pada citra. Dalam prakteknya, operator Forstner sering digunakan karena mudah diperluas untuk mendeteksi pusat fitur-fitur melingkar bersama dengan sudut.

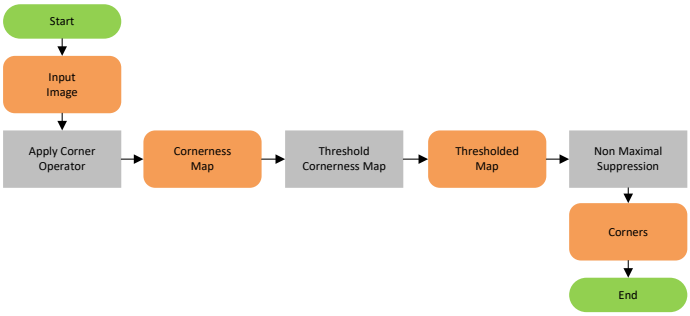
3.2. Rancangan Simulasi

Pada tahap ini dimulai dengan analisa kinerja dari masing-masing algoritma yang akan diuji yaitu; *Moravec*, *Susan*, *Harris*, *FAST*, *Eigen* dan *Forstner* yang akan dijelaskan dalam *flowchart*, kemudian juga dilakukan perancangan terhadap simulasi yang akan dibuat.

3.3. Implementasi

Langkah selanjutnya adalah implementasi dari masing-masing algoritma *Moravec*, *Susan*, *Harris*, FAST, *Eigen* dan *Forstner* untuk melakukan proses pendeteksian sudut pada citra *noise* dengan menggunakan mekanisme dari masing-masing detektor sudut yang akan dibandingkan, terlihat pada gambar 3.1.

.



Gambar 3.13. *Flowchart* Algoritma Pendeteksian Titik Sudut

Pada gambar 3.13 di atas dijelaskan bahwa pertama adalah *start* untuk memulai program selanjutnya melakukan pembacaan citra target (*input image*) sebagai inputan, kemudian melakukan pendeteksian sudut untuk setiap piksel dengan menggunakan masing-masing operator sudut (*apply corner operator*), hasil dari proses ini adalah peta yang berisikan titik sudut dan kandidat sudut pada citra (*cornerness map*), selanjutanya menemukan titik sudut sebagai lokal maksima dalam peta sudut (*threshold cornerness map*), hasil dari proses ini adalah peta sudut sebagai lokal maksima dengan menggunakan pengaturan nilai ambang tertentu (*threshold map*), dilanjutkan dengan menentukan non maksima untuk menentukan lokasi titik yang menjadi titik sudut sebenarnya (*non-maximum suppression*) dalam sebuah citra.

3.4. Pengujian

Selanjutnya adalah melakukan proses pengujian, implementasi algoritma dengan tujuan untuk mendapatkan komparasi performa dan hasil yang diinginkan.

3.5. Dokumentasi

Pedokumentasian hasil penelitan adalah untuk membuat dokumentasi berupa laporan penelitian sebagai bukti tertulis kegiatan penelitian telah dilaksanakan secara baik dan benar serta dapat bermanfaat bagi pengembangan dan penelitian selanjutnya.

IV. HASIL DAN PEMBAHASAN

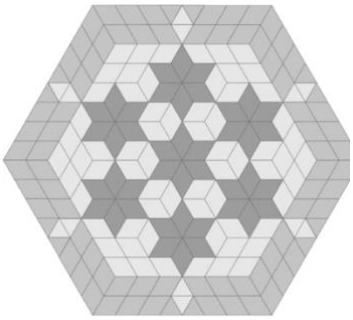
4.1. Hasil Pengujian Setiap Detektor

Citra yang digunakan dalam penelitian ini adalah citra *grayscale* dengan ukuran 605x500 piksel, kemudian tipe *noise* yang digunakan adalah *gaussian*, *poisson*, *salt and pepper* dan *speckle*. Citra yang tidak mengandung *noise* dinamakan citra original. Secara berurutan proses pendekeksian titik sudut dilakukan pertamakali adalah proses deteksi sudut pada citra original untuk masing-masing algoritma pendeteksi sudut yang akan dibandingkan, hasil pendeteksian sudut pada citra yang memiliki *noise* untuk setiap jenis *noise*. Secara lengkap hasil dari pengujian masing-masing detektor akan dijelaskan pada poin 4.1.1 sampai dengan 4.1.6 berikut.

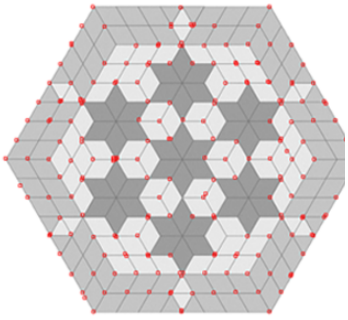
4.1.1. Detektor Sudut Moravec

Hasil pengujian detektor Moravek terlihat secara detail pada gambar 4.16 (citra original), gambar 4.16a (citra tanpa *noise*), gambar 4.16b (citra *noise gaussian*), gambar 4.16c (citra *noise poisson*), gambar 4.16d (citra *noise salt and pepper*) dan gambar 4.16e (citra *noise*

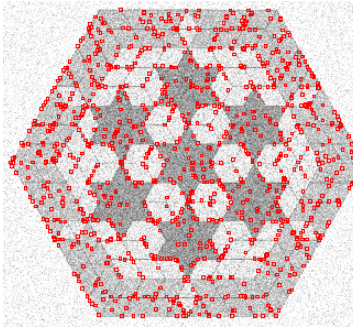
speckle). Terlihat jelas adanya perubahan yaitu pertambahan titik-titik sudut pada citra yang memiliki *noise* (*gaussian*, *poisson*, *salat pepper* dan *speckle*), ini mengidikasikan bahwa algoritma detektor sudut Moravec sangat sensitif terhadap *noise* sehingga akan menurunkan tingkat akurasi dan presisi hasil deteksi titik sudut yang sebenarnya.



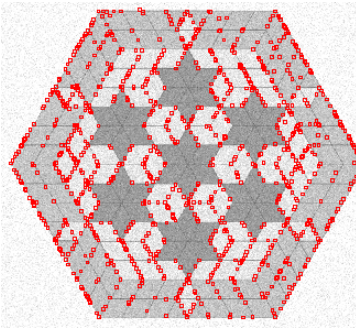
Gambar 4.16. Citra Original



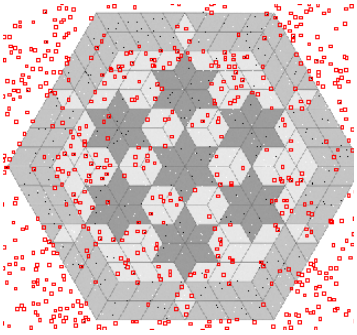
Gambar 4.16a. Citra Tanpa *Noise*



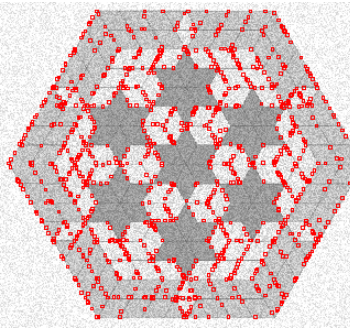
Gambar 4.16b. Citra *Noise Gaussian*



Gambar 4.16c. Citra *Noise Poisson*



Gambar 4.16d. Citra *Noise Salt and Pepper*



Gambar 4.16e. Citra *Noise Speckle*

Titik-titik sudut hasil deteksi oleh detektor Moravec adalah sebagai berikut; 1) 204 titik pada citra tanpa *noise*, 2) 1000 titik pada citra *noise gaussian*, *salt and pepper*, dan *speckle* hasil detail dapat dilihat pada tabel 4.1. Detektor sudut Moravec sangat sensitif terhadap jenis *noise salt and pepper* (gambar 4.16d), semetara untuk jenis *noise gaussian*, *poisson*, dan *speckle* titik-titik yang terdeteksi masih berada dalam area citra sebaliknya pada jenis *noise salt and pepper* titik-titik yang berada di luar area citra juga ikut terdeteksi, dengan pengertian lain tipe *noise* inilah yang paling mempengaruhi tingkat keakuratan dan presisi hasil pendeteksian titik sudut sebenarnya pada detektor sudut Moravec.

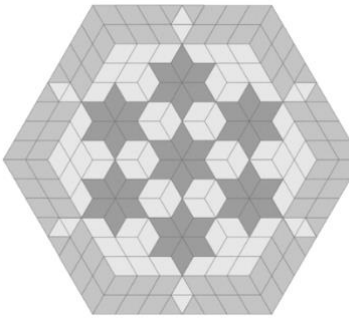
Pertambahan titik-titik sudut yang diakibatkan oleh setiap *noise* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*) adalah sebesar 4.90 kali, hasil detail dapat dilihat pada tabel 4.2. Sedangakan waktu eksekusi algoritma detektor sudut Moravec dalam melakukan proses pendeteksian titik-titik sudut adalah sebagai berikut; 1) citra tanpa *noise* sebesar 0,81 detik, 2) citra *noise gaussian* sebesar 2,37 detik, 3) citra *noise poisson* sebesar 2,04 detik, 4) citra *noise salt and pepper* sebesar 2,22 detik dan 5) citra *noise speckle* sebesar 2,13 detik. Data-data ini menunjukkan bahwa semua jenis *noise* akan mempengaruhi cepat atau lambatnya waktu proses pendeteksian titik-titik sudut pada sebuah citra. Dengan pengertian lain waktu

proses pendeteksian sudut pada citra yang memiliki *noise* akan memakan waktu yang lebih lama jika dibanding proses pendeteksian titik-titik sudut pada citra yang tidak memiliki *noise,* hasil detail dapat dilihat pada tabel 4.3.

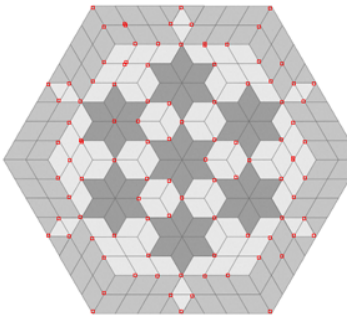
Hasil akurasi dan tingkat presisi detektor Moravec dalam mendeteksi titik sudut yang sebenarnya pada citra tanpa *noise* dan pada citra yang memiliki *noise* adalah sebagai berikut; 1) citra tanpa *noise* seharusnya 217 titik sudut yang sebenarnya menjadi 204 titik sudut kekurangan sebanyak 13 titik sudut, 2) citra *noise gaussian, poisson*, *salt and pepper*, dan *speckle* dari 217 titik sudut yang sebenarnya menjadi 1000 titik sudut atau kelebihan sebanyak 783 titik sudut, hasil detail dapat dilihat pada tabel 4.4. Dapat disimpulkan detektor Moravec sangat sensitif terhadap *noise* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*) karena tidak dapat mendeteksi titik-titik sudut sebenarnya dengan tepat.

4.1.2. Detektor Sudut Susan

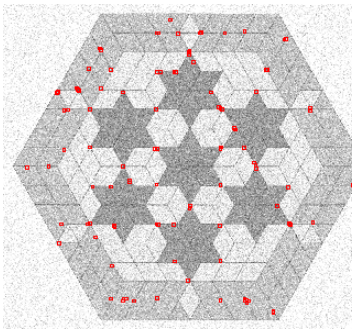
Hasil pengujian detektor Susan terlihat secara detail pada gambar 4.17 (citra toriginal), gambar 4.17a (citra tanpa *noise*), gambar 4.17b (citra *noise gaussian*), gambar 4.17c (citra *noise poisson*), gambar 4.17d (citra *noise salt and pepper*) dan gambar 4.17e (citra *noise speckle*).



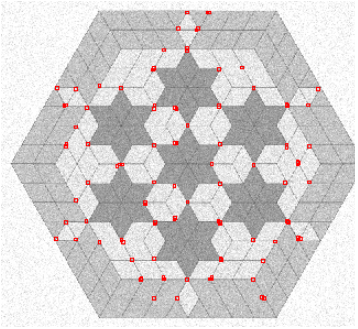
Gambar 4.17. Citra Original



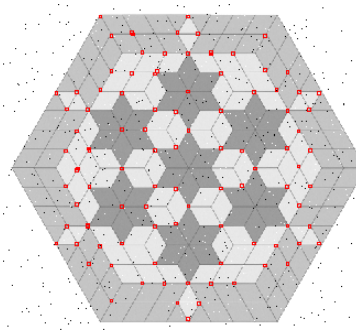
Gambar 4.17a. Citra Tanpa *Noise*



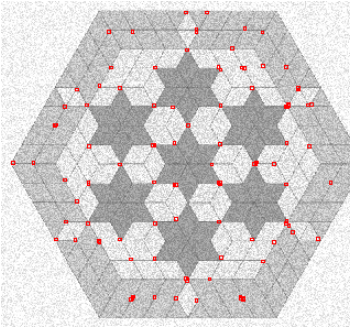
Gambar 4.17b. Citra *Noise Gaussian*



Gambar 4.17c. Citra *Noise Poisson*



Gambar 4.17d.Citra *Noise Salt and Pepper*



Gambar 4.17e. Citra *Noise Speckle*

Titik-titik sudut yang dapat dideteksi oleh detektor Susan adalah sebagai berikut; 1) 100 titik pada citra tanpa *noise* (gambar 4.17a), 2) 100 titik pada citra *noise gaussian*, *salt and*

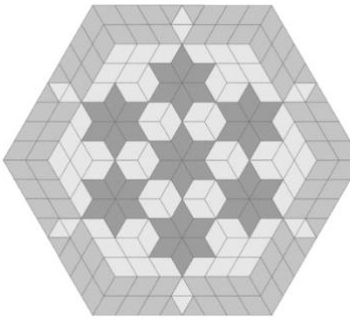
pepper dan *speckle* hasil detail dapat dilihat pada tabel 4.1. Detektor sudut Susan sensitif terhadap jenis *noise gaussian*, *poisson*, *salt and pepper* dan *speckle*, walaupun hasil deteksi titik sudut masih berada di dalam area citra. Sama halnya dengan detektor Moravec *noise* juga akan mengakibatkan menurunya tingkat akurasi dan presisi hasil pendeteksian titik sudut yang sebenarnya. Walaupun *noise* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*) pada citra tidak mengakibatkan pertambahan jumlah titik-titik sudut pada detektor sudut Susan, hasil detail dapat dilihat pada tabel 4.2.

Waktu eksekusi algoritma detektor sudut Susan dalam melakukan proses pendeteksian titik-titik sudut adalah sebagai berikut; 1) citra tanpa *noise* sebesar 16,12 detik, 2) citra *noise gaussian* sebesar 24,54 detik, 3) citra *noise poisson* sebesar 24,57 detik, 4) citra *noise salt and pepper* sebesar 22,27 detik dan 5) citra *noise speckle* sebesar 24,57 detik. Data-data ini menunjukkan bahwa semua jenis *noise* akan mempengaruhi waktu proses pendeteksian titik-titik sudut pada sebuah citra. Hasil detail dapat dilihat pada tabel 4.3.

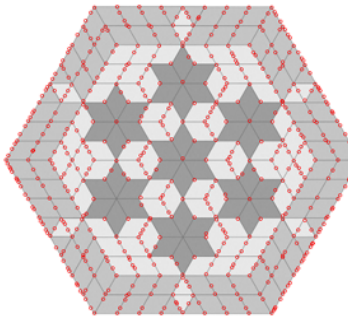
Hasil akurasi dan tingkat presisi detektor Susan dalam mendeteksi titik sudut yang sebenarnya pada citra tanpa *noise* dan pada citra yang memiliki *noise* adalah sebagai berikut; 1) citra tanpa *noise* seharusnya 217 titik sudut yang sebenarnya menjadi 100 titik sudut kekurangan sebanyak 117 titik sudut, 2) citra *noise gaussian, poisson*, *salt and pepper* dan *speckle* dari 217 titik sudut yang sebenarnya menjadi 100 titik sudut atau kekurangan sebanyak 117 titik sudut, hasil detail dapat dilihat pada tabel 4.4. Akurasi hasil deteksi titik sudut detektor Susan dapat dikatakan paling baik dan stabil terhadap *noise* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*) karena masih dapat mendeteksi titik-titik yang merupakan titik sudut yang sebenarnya sebanyak 100 titik sudut, walaupun memiliki waktu yang realif lebih lama jika dibandingkan dengan detektor-detektor sudut lain yang diujikan.

4.1.3. Detektor Sudut Harris

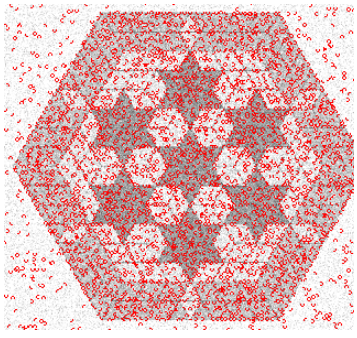
Hasil pengujian detektor Harris terlihat secara detail pada gambar 4.18 (citra original), gambar 4.18a (citra tanpa *noise*), gambar 4.18b (citra *noise gaussian*), gambar 4.18c (citra *noise poisson*), gambar 4.18d (citra *noise salt and pepper*) dan gambar 4.18e (citra *noise speckle*).



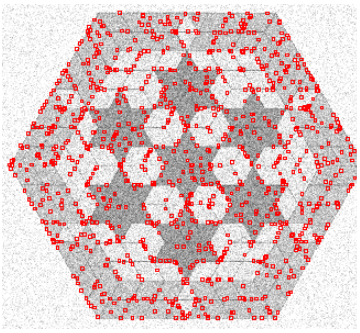
Gambar 4.18. Citra Original



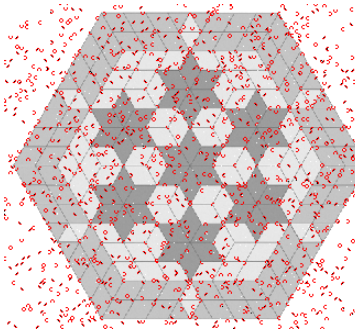
Gambar 4.18a. Citra Tanpa *Noise*



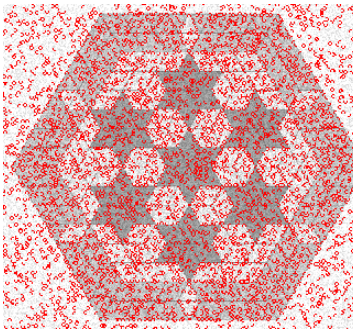
Gambar 4.18b. Citra *Noise Gaussian*



Gambar 4.18c. Citra *Noise Poisson*



Gambar 4.18d. Citra *Noise Salt and Pepper*



Gambar 4.18e. Citra *Noise Speckle*

Titik-titik sudut yang dapat dideteksi oleh detektor Harris adalah sebagai berikut; 1) 571 titik pada citra tanpa *noise* (gambar 4.18a), 2) 3.521 titik pada citra *noise gaussian*, 3) 2.125 titik pada citra *noise poisson,* 4) 1.559 titik pada citra *noise salt and pepper* dan 4.908 titik pada citra *noise speckle*, hasil detail dapat dilihat pada tabel 4.1. Detektor sudut Harris sangat sensitif terhadap jenis *noise gaussian* (gambar 4.18b), *salt and pepper* (gambar 4.18d) dan *speckle* (gambar 4.18e), ketiga tipe *noise* ini mengakibatkan titik-titik yang berada di luar area citra ikut terdeteksi. Sama halnya dengan detektor Moravec dan Susan, pada detektor Harris *noise* juga akan mengakibatkan menurunya tingkat akurasi dan presisi hasil pendeteksian titik sudut yang sebenarnya.

Pertambahan titik-titik sudut yang diakibatkan oleh setiap *noise* pada detektor sudut Harris adalah sebagai berikut; 1) citra *noise gaussian* sebesar 6,17 kali, 2) citra *noise poisson* sebesar 3,72 kali, 3) citra *noise salt and pepper* sebesar 2,73 kali dan citra *noise speckle* sebesar 8,60 kali, hasil detail dapat dilihat pada tabel 4.2.

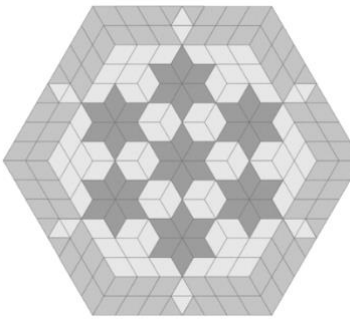
Sedangakan waktu eksekusi algoritma detektor sudut Harris dalam melakukan proses pendeteksian titik-titik sudut adalah sebagai berikut; 1) citra tanpa *noise* sebesar 0,61 detik, 2) citra *noise gaussian* sebesar 0,97 detik, 3) citra *noise poisson* sebesar 0,85 detik, 4) citra *noise salt and pepper* sebesar 0,81 detik dan 5) citra *noise speckle* sebesar 1,00 detik. Data-data menunjukkan bahwa semua jenis *noise* akan mempengaruhi waktu proses pendeteksian titik-titik sudut pada sebuah citra. Hasil detail dapat dilihat pada tabel 4.3.

Hasil akurasi dan tingkat presisi detektor Harris dalam mendeteksi titik sudut yang sebenarnya pada citra tanpa *noise* dan pada citra yang memiliki *noise* adalah sebagai berikut; 1) citra tanpa *noise* seharusnya 217 titik sudut yang sebenarnya menjadi 571 titik sudut kelebihan sebanyak 354 titik sudut, 2) citra *noise gaussian* 3.521 titik sudut atau kelebihan 3.304 titik sudut*,* citra *noise poisson* sebanyak 2.125 titik sudut kelebihan 1.908 titik sudut, citra *noise salt and pepper* sebesar 1.559 titik sudut kelebihan 1.342 titik sudut dan citra *noise speckle* 4.908 titik sudut kelebihan 4.691 titik sudut, hasil detail dapat dilihat pada tabel 4.4. Disimpulkan bahwa detektor sudut Harris juga sangant sentitif terhadap *noise* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*) karena tidak mampu mendeteksi titik-titik sudut yang sebenarnya, walupun waktu dalam proses pendeteksian relatif cepat jika dibandingkan dengan detektor Susan, Forstner, Moravec dan Eigen.

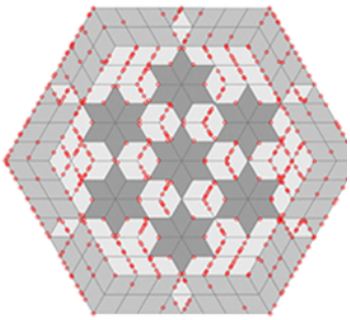
4.1.4. Detektor Sudut FAST

Hasil pengujian detektor FAST terlihat secara detail pada gambar 4.19 (citra original), gambar 4.19a (citra tanpa *noise*), gambar 4.19b (citra *noise gaussian*), gambar 4.19c (citra

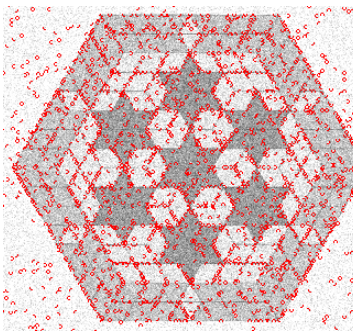
noise poisson), gambar 4.19d (citra *noise salt and pepper*) dan gambar 4.19e (citra *noise speckle*).



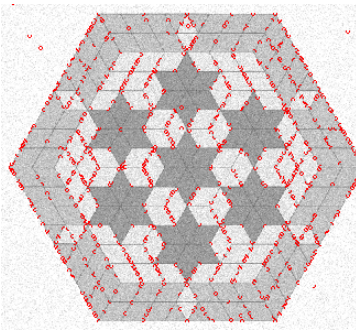
Gambar 4.19. Citra Original



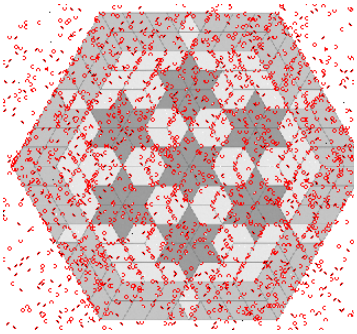
Gambar 4.19a. Citra Tanpa *Noise*



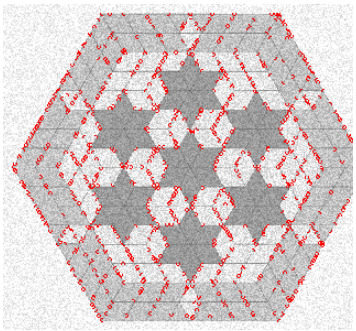
Gambar 4.19b. Citra *Noise Gaussian*



Gambar 4.19c. Citra *Noise Poisson*



Gambar 4.19d. Citra *Noise Salt and Pepper*



Gambar 4.19e. Citra *Noise Speckle*

Titik-titik sudut yang dapat dideteksi oleh detektor FAST adalah sebagai berikut; 1) 538 titik pada citra tanpa *noise* (gambar 4.19a), 2) 2.461 titik pada citra *noise gaussian*, 3) 782 titik pada citra *noise poisson,* 4) 2.473 titik pada citra *noise salt and pepper* dan 841 titik pada citra *noise speckle*, hasil detail dapat dilihat pada tabel 4.1. Detektor sudut FAST sangat sensitif terhadap jenis *noise gaussian* (gambar 4.19b), *salt and pepper* (gambar 4.19d) dan *poisson* (gambar 4.19c), ketiga tipe *noise* ini mengakibatkan titik-titik yang berada di luar area citra juga terdeteksi. Sama halnya dengan detektor Moravec, Susan dan Harris pada detektor FAST *noise* juga akan mengakibatkan menurunya tingkat akurasi dan presisi hasil pendeteksian titik sudut yang sebenarnya.

Pertambahan titik-titik sudut yang diakibatkan oleh setiap *noise* pada detektor sudut FAST adalah sebagai berikut; 1) citra *noise gaussian* sebesar 4,57 kali, 2) citra *noise poisson* sebesar 1,45 kali, 3) citra *noise salt and pepper* sebesar 4,60 kali dan citra *noise speckle* sebesar 1,56 kali, hasil detail dapat dilihat pada tabel 4.2.

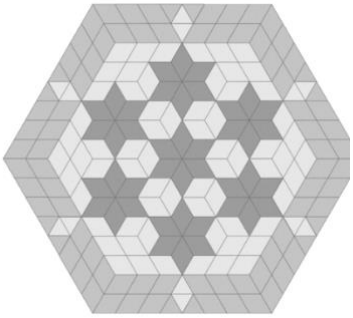
Sedangakan waktu eksekusi algoritma detektor sudut FAST dalam melakukan proses pendeteksian titik-titik sudut adalah sebagai berikut; 1) citra tanpa *noise* sebesar 0,25 detik, 2) citra *noise gaussian* sebesar 0,27 detik, 3) citra *noise poisson* sebesar 0,26 detik, 4) citra *noise salt and pepper* sebesar 0,26 detik dan 5) citra *noise speckle* sebesar 0,25 detik. Ini mengindikasikan relatif tidak terjadi perubahan waktu proses pendeteksian titik sudut baik

pada citra tanpa *noise* dan pada citra yang memiliki *noise*. Dengan pengertian lain waktu proses pendeteksian titik sudut pada detektor FAST relatif tidak dipengaruhi oleh adanya *noise*, hasil detail dapat dilihat pada tabel 4.3.

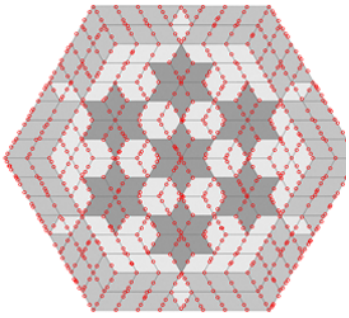
Hasil akurasi dan tingkat presisi detektor FAST dalam mendeteksi titik sudut yang sebenarnya pada citra tanpa *noise* dan pada citra yang memiliki *noise* adalah sebagai berikut; 1) citra tanpa *noise* seharusnya 217 titik sudut yang sebenarnya menjadi 538 titik sudut kelebihan sebanyak 321 titik sudut, 2) citra *noise gaussian* 2.461 titik sudut atau kelebihan 2.244 titik sudut, citra *noise poisson* sebanyak 782 titik sudut kelebihan 565 titik sudut, citra *noise salt and pepper* sebesar 2,473 titik sudut kelebihan 2.256 titik sudut dan citra *noise speckle* 841 titik sudut kelebihan 624 titik sudut, hasil detail dapat dilihat pada tabel 4.4. Akurasi detektor ini juga relatif rendah dan sangat sensitif terhadap *noise* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*) karena tidak dapat mendeteksi titik-titik sudut yang sebenarnya dengan tepat, walaupun memiliki waktu proses pendeteksian tercepat diantara semua detektor sudut yang diujikan.

4.1.5. Detektor Sudut Eigen

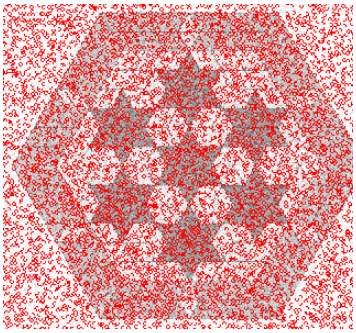
Hasil pengujian detektor Eigen terlihat secara detail pada gambar 4.20 (citra original), gambar 4.20a (citra tanpa *noise*), gambar 4. 20b (citra *noise gaussian*), gambar 4.20c (citra *noise poisson*), gambar 4.20d (citra *noise salt and pepper*) dan gambar 4.20e (citra *noise speckle*).



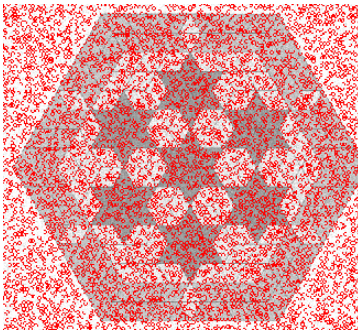
Gambar 4.20. Citra Original



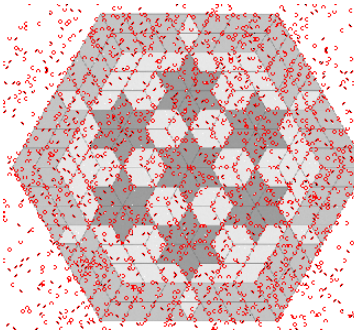
Gambar 4.20a. Citra Tanpa *Noise*



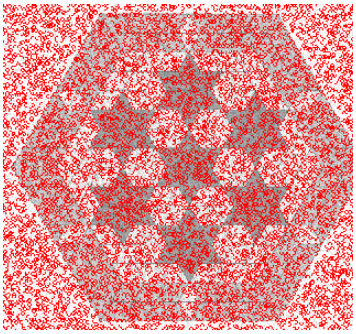
Gambar 4.20b. Citra *Noise Gaussian*



Gambar 4.20c. Citra *Noise Poisson*



Gambar 4.20d. Citra *Noise Salt and Pepper*



Gambar 4.20e. Citra *Noise Speckle*

Titik-titik sudut yang dapat dideteksi oleh detektor Eigen adalah sebagai berikut; 1) 796 titik pada citra tanpa *noise* (gambar 4.20a), 2) 7.891 titik pada citra *noise gaussian*, 3) 7.970 titik pada citra *noise poisson,* 4) 2.238 titik pada citra *noise salt and pepper* dan 9.638 titik pada citra *noise speckle*, hasil detail dapat dilihat pada tabel 4.1. Detektor sudut Eigen lebih sensitif terhadap jenis *noise gaussian*, *poisson* (gambar 4.20c) dan *speckle* (gambar 4.20e), ketiga tipe *noise* ini mendeteksi titik-titik di luar area citra dan hampir menututupi citra, sementara jika dibandingkan dengan *noise salt and pepper* (gambar 4.20d) tidak samapai menutupi citra. Sama halnya dengan detektor Moravec, Susan, Harris dan FAST pada detektor Eigen *noise* juga akan mengakibatkan menurunya tingkat akurasi dan presisi hasil pendeteksian titik sudut yang sebenarnya.

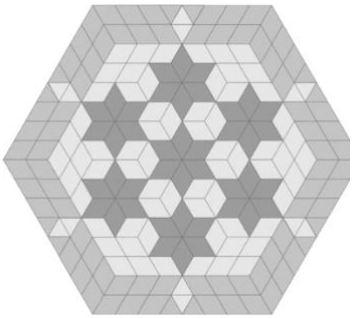
Pertambahan titik-titik sudut yang diakibatkan oleh setiap *noise* pada detektor sudut Eigen adalah sebagai berikut; 1) citra *noise gaussian* sebesar 9,91 kali, 2) citra *noise poisson* sebesar 10,01 kali, 3) citra *noise salt and pepper* sebesar 2,81 kali dan citra *noise speckle* sebesar 12,11 kali, hasil detail dapat dilihat pada tabel 4.2.

Sedangakan waktu eksekusi algoritma detektor sudut Eigen dalam melakukan proses pendeteksian titik-titik sudut adalah sebagai berikut; 1) citra tanpa *noise* sebesar 0,70 detik, 2) citra *noise gaussian* sebesar 1.26 detik, 3) citra *noise poisson* sebesar 1,23 detik, 4) citra *noise salt and pepper* sebesar 0,89 detik dan 5) citra *noise speckle* sebesar 1,43 detik. Ini mengindikasikan adanya perubahan waktu proses pendeteksian titik sudut dari citra tanpa *noise* dibandingkan pada citra yang memiliki *noise*. Hasil detail dapat dilihat pada tabel 4.3.

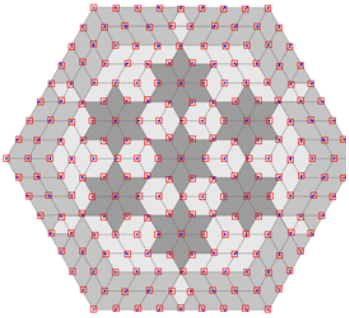
Hasil akurasi dan tingkat presisi detektor Eigen dalam mendeteksi titik sudut yang sebenarnya pada citra tanpa *noise* dan pada citra yang memiliki *noise* adalah sebagai berikut; 1) citra tanpa *noise* seharusnya 217 titik sudut yang sebenarnya menjadi 796 titik sudut kelebihan sebanyak 579 titik sudut, 2) citra *noise gaussian* 7.891 titik sudut atau kelebihan 7.674 titik sudut*,* citra *noise poisson* sebanyak 7.970 titik sudut kelebihan 7.753 titik sudut, citra *noise salt and pepper* sebesar 2.238 titik sudut kelebihan 2.021 titik sudut dan citra *noise speckle* 9.638 titik sudut kelebihan 9.421titik sudut, hasil detail dapat dilihat pada tabel 4.4. Akurasi hasil detektor sudut Eigen juga relatif rendah dan sangat sensitif terhadap *niose* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*), detektor ini tidak dapat mendeteksi dengan tepat titik-titik yang merupakan titik sudut yang sebenarnya. Walaupun waktu proses pendeteksian relatif cukup cepat jika dibandingkan dengan waktu detektor sudut Susan, Forstner dan Moravec.

4.1.6 Detektor Sudut Forstner

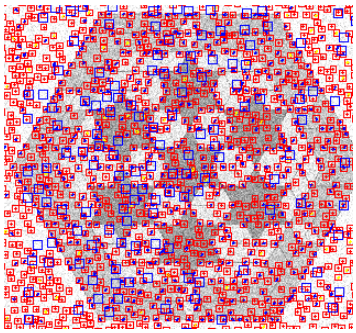
Hasil pengujian detektor Forstner terlihat secara detail pada gambar 4.21 (citra original), gambar 4.21a (citra tanpa *noise*), gambar 4. 21b (citra *noise gaussian*), gambar 4.21c (citra *noise poisson*), gambar 4.21d (citra *noise salt and pepper*) dan gambar 4.21e (citra *noise speckle*).



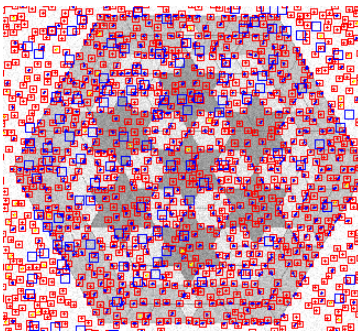
Gambar 4.21. Citra Original



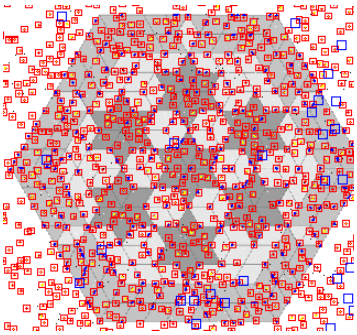
Gambar 4.21a. Citra Tanpa *Noise*



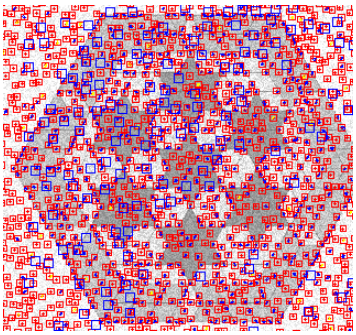
Gambar 4.21b. Citra *Noise Gaussian*



Gambar 4.21c. Citra *Noise Poisson*



Gambar 4.21d. Citra *Noise Salt and Pepper*



Gambar 4.21e.Citra *Noise Speckle*

Titik-titik sudut yang dapat dideteksi oleh detektor Forstner adalah sebagai berikut; 1) 217 titik pada citra tanpa *noise* (gambar 4.20a), 2) 821 titik pada citra *noise gaussian*, 3) 868 titik pada citra *noise poisson,* 4) 940 titik pada citra *noise salt and pepper* dan 854 titik pada citra *noise speckle*, hasil detail dapat dilihat pada tabel 4.1. Detektor sudut Forstner sangat sensitif terhadap kempat jenis *noise*, keempat tipe *noise* mengakibatkan titik-titik yang berada di luar area citra ikut terdeteksi dan hampir menututupi citra. Sama halnya dengan detektor Moravec, Susan, Harris, FAST, Eigen dan pada detektor Forstner *noise* juga akan mengakibatkan menurunya tingkat akurasi dan presisi hasil pendeteksian titik sudut yang sebenarnya.

Pertambahan titik-titik sudut yang diakibatkan oleh setiap *noise* pada detektor sudut Eigen adalah sebagai berikut; 1) citra *noise gaussian* sebesar 3,78 kali, 2) citra *noise poisson* sebesar 4,00 kali, 3) citra *noise salt and pepper* sebesar 4,33 kali dan citra *noise speckle* sebesar 3,94 kali, hasil detail dapat dilihat pada tabel 4.2.

Sedangakan waktu eksekusi algoritma detektor sudut Forstner dalam melakukan proses pendeteksian titik-titik sudut adalah sebagai berikut; 1) citra tanpa *noise* sebesar 2,77 detik, 2) citra *noise gaussian* sebesar 6,37 detik, 3) citra *noise poisson* sebesar 6,04 detik, 4) citra *noise salt and pepper* sebesar 5,59 detik dan 5) citra *noise speckle* sebesar 6,26 detik. Ini mengindikasikan adanya perubahan waktu proses pendeteksian titik sudut dari citra tanpa *noise* dibandingkan pada citra yang memiliki *noise*. Hasil detail dapat dilihat pada tabel 4.3.

Hasil akurasi dan tingkat presisi detektor Forstner dalam mendeteksi titik sudut yang sebenarnya pada citra tanpa *noise* dan pada citra yang memiliki *noise* adalah sebagai berikut;

1) citra tanpa *noise* sebanyak 217 titik sudut yang sebenarnya detektor Forstner berhasil tepat mendetekasi sebanyak 217 titik sudut, 2) citra *noise gaussian* 821 titik sudut atau kelebihan 604 titik sudut, citra *noise poisson* sebanyak 868 titik sudut kelebihan 651 titik sudut, citra *noise salt and pepper* sebesar 940 titik sudut kelebihan 723 titik sudut dan citra *noise speckle* 854 titik sudut kelebihan 637 titik sudut, hasil detail dapat dilihat pada tabel 4.4. Akurasi dari detektor Forstner ini juga ralatif rendah dan sensitif terhadap *noise* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*), tidak dapat mendeteksi titik-titik yang merupakan titik sudut sebenarnya dengan tepat dan waktu proses juga relatif lama jika dibandingkan dengan detektor FAST, Harris, Eigen dan Moravec. Detektor ini memiliki akurasi terbaik hanya pada citra yang tidak memiliki *noise*.

4.1.6. Pendeteksian Titik-titik Sudut

Hasil pengujian dari masing-masing detektor sudut baik terhadap citra tanpa *noise* dan pada citra yang memiliki berbagai tipe *noise* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*) untuk setiap detektor sudut yang telah dijelaskan sebelumnya, secara lengkap disajikan pada tabel-tabel sebagai berikut; a) tabel 4.1, b) tabel 4.2, c) tabel 4.3 dan d) tabel 4.4 di bawah ini.

Tabel 4.1. Hasil Pendeteksian Titik Sudut SetiapDetektor

Detektor	**Original**	**Titik Sudut**			
		Noise			
		Gaussian	***Poisson***	***Salt_Pepper***	***Speckle***
Moravec	204	1.000	1.000	1.000	1.000
Susan	100	100	100	100	100
Harris	571	3.521	2.125	1.559	4.908
FAST	538	2.461	782	2.473	841
Eigen	796	7.891	7.970	2.238	9.638
Forstner	217	821	868	940	854

Data-data pada tabel 4.1, memperlihatkan hasil pendeteksian titik-titik sudut hasil dari setiap detektor yang di ujikan dalam penelitian ini, sebagai pembanding digunakan citra yang tidak memiliki *noise* yang memiliki rentan intensitas antara 124 sampai dengan 234 dan memiliki titik sudut sebenarnya sebanyak 217 titik sudut. Hasil yang didapatkan sebagai berikut; 1) detektor Moravec menghasilkan 204 titi-titik sudut, 2) detektor Susan menghasilkan 100 titik-titik sudut, 3) detektor Harris menghasilkan 571 titik-titik sudut, detektor FAST menghasilkan 538 titik-titik sudut, detektor Eigen menghasilkan 796 titik-titik sudut dan detektor Forstner menghasilkan 217 titik-titik sudut. Pendeteksian pada citra tanpa *noise* hanya detektor sudut Forstner yang dengan tepat menemukan titik sudut sebenarnya.

Tabel 4.2. Pertambahan Jumlah Titik Sudut pada Citra *Noise* Setiap Detektor

Detektor	**Original**	***Noise***			
		Gaussian	***Poisson***	***Salt_Pepper***	***Speckle***
Moravec	204	4,90	4,90	4,90	4,90
Susan	100	0,00	0,00	0,00	0,00
Harris	571	6,17	3,72	2,73	8,60
FAST	538	4,57	1,45	4,60	1,56
Eigen	796	9,91	10,01	2,81	12,11
Forstner	217	3,78	4,00	4,33	3,94

Waktu proses pendeteksian (tabel 4.3) memperlihatkan bahwa waktu tercepat dalam proses pendeteksian titik sudut pada citra tanpa *noise* secara berurutan dari yang paling cepat adalah sebagai berikut; 1) urutan pertama detektor FAST sebesar 0.25 detik, 2) urutan kedua detektor Harris sebesar 0.61 detik, 3) ururtan ketiga detektor Eigen sebesar 0.70 detik, 4) urutan keempat detektor Moravec sebesar 0.81 detik, 5) urutan kelima detektor Forstner sebesar 2.77 detik. Sementara waktu proses pendeteksian titik sudut pada citra noise detektor tercepat secara berurutan adalah sebagai berikut; 1) detektor FAST dengan rata-rata waktu sebesar 0.26 detik, 2) detektor Harris dengan rata-rata waktu sebesar 0.91 deti, 3) detektor Eigen dengan rata-rata waktu sebesar 1.22 detik, 4) detektor Moravec dengan rata-rata waktu sebesar 2.19 detik, 5) detektor Forstner dengan rata-rata waktu sebesar 6.06 detik dan 6) detektor Susan dengan rata-rata waktu sebesar 23.99 detik.

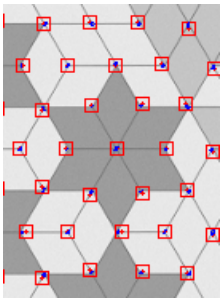
Tabel 4.3. Waktu Eksekusi Setiap Detektor Sudut untuk Masing-masing Tipe *Noise*

Noise	**Waktu (detik)**					
	Detektor Sudut					
	Moravec	***Susan***	***Harris***	***FAST***	***Eigen***	***Forstner***
Original	0,81	16,12	0,61	0,25	0,70	2,77
Gaussian	2,37	24,54	0,97	0,27	1,26	6,37
Poisson	2,04	24,57	0,85	0,26	1,33	6,04
Salt_Papper	2,22	22,27	0,81	0,26	0,89	5,59
Speckle	2,13	24,57	1,00	0,25	1,43	6,26

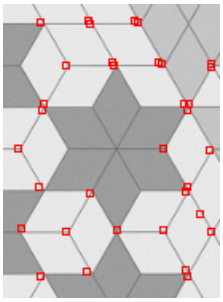
Tabel 4.4. Akurasi Setiap Detektor Sudut untuk Masing-masing Tipe *Noise*

Titik Sudut Sebenarnya	**Detektor**	**Tanpa**	***Noise***			
		Noise	***Gaussian***	***Poisson***	***Salt_Pepper***	***Speckle***
217	***Moravec***	-13	783	783	783	783
	Susan	-117	-117	-117	-117	-117
	Harris	354	3.304	1.908	1.342	4.691
	FAST	321	2.244	565	2.256	624
	Eigen	579	7.674	7753	2.021	9.421
	Forstner	0	604	651	723	637

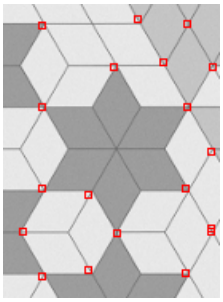
Tingkat akurasi pada tabel 4.4, memperlihatkan tidak satupun detektor sudut yang diuji dalam penelitian ini yang memberikan hasil akurat dalam mendeteksi jumlah titik sudut yang sebenarnya pada citra yang memiliki *noise* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*) ini mengindikasikan bahwa semua detektor yang diuji sensitif terhadap *noise*. Sebagai bandingan keseluruhan detektor juga diujikan untuk mendeteksi titik-titik sudut pada citra yang tidak memiliki *noise* dan hasilnya adalah hanya detektor sudut Forstner yang memiliki tingkat akurasi terbaik dan mampu mendeteksi 217 titik sudut sebenarnya yang ada pada citra (tanpa *noise*). Akurasi dan ketepatan hasil pendeteksian titik sudut terlihat pada potongan gambar 4.22a, 4.22b, 4.22c, 4.22d, 4.22e dan 4.22f. di bawah ini



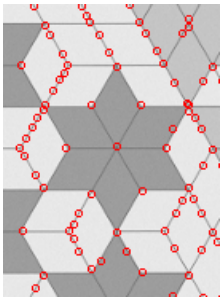
Gambar 4.22a. Forstner



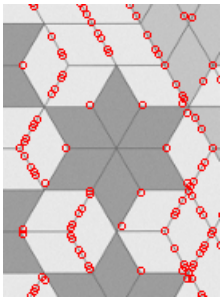
Gambar 4.22b. Moravec



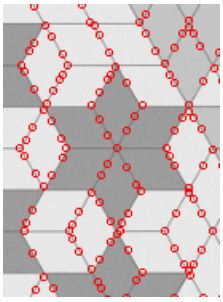
Gambar 4.22c. Susan



Gambar 4.22d. Harris



Gambar 4.22e. FAST



Gambar 4.22f. Eigen

Noise pada citra adalah variasi acak (tidak berasal dalam objek foto) yang merupakan informasi kecerahan atau warna dalam sebuah citra, dan biasanya merupakan aspek *noise* elektronis. *Noise* citra merupakan hal yang tidak dinginkan karena dapat mengurangi kualitas dan keakuratan informasi sebuah citra. Hal ini terbukti dari keseluruhan detektor sudut yang diujikan dalam penelitian ini tidak ada detektor sudut yang mampu dengan tepat melakukan pendeteksian titik-titik sudut yang sebenarnya pada citra yang memiliki *noise*.

Tabel 4.5. Hasil Komparasi Setiap Detektor Sudut

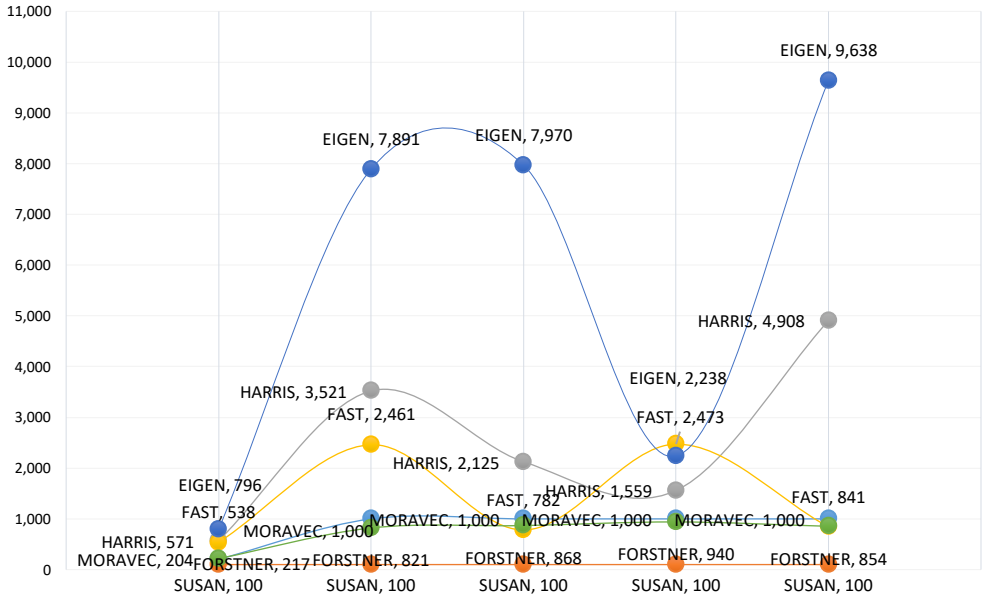
Detektor	Tingkat Akurasi		Tingkat Kestabilan		Kecepatan
	Jumlah	Lokasi	Perulangan	*Noise*	
Moravec	*	**	**	*	**
Susan	*	***	**	**	*
Harris	*	*	*	*	***
FAST	*	*	*	*	****
Eigen	*	*	*	*	***
Forstner	*	*	*	*	**

Keterangan : * = Rendah, ** = Sedang, *** = Tinggi, **** = Sangat Tinggi

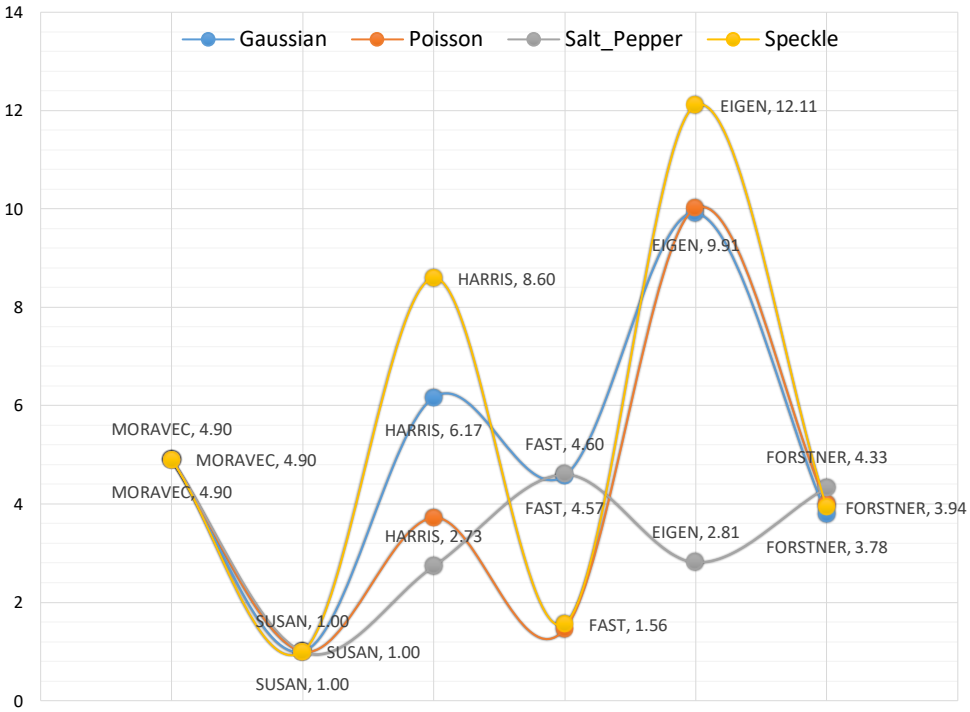
Idealnya setiap algoritma detektor sudut harus memiliki kestabilan respon *noise* yang baik sehingga mampu mengurangi ganguan *noise* yang dapat menurunkan kualitas dan

keakuratan hasil yang diinginkan. Hasil komapasrasi dari masing-masing detektor sudut pada citra yang memiliki noise disajikan selengkapnya pada tabel 4.5. di atas.

Pola pertambahan titik-titik sudut hasil pendeteksian untuk setiap detektor sudut digambarkan oleh gambar 4.24. Gambar ini menjelaskan pertambahan titik-titik sudut akibat *noise* terjadi paling ekstrim pada detektor Eigen, dari 769 titik-titik sudut pada citra tanpa *noise* langsung naik menjadi 7.891 titik sudut pada citra yang memiliki *noise* jenis *gaussian*, 7.970 titik sudut pada citra *noise poisson*, dan turun menjadi 2.238 titik sudut pada citra *noise salt and pepper* dan naik kembali menjadi 9.638 titik sudut pada citra *noise speckle*.



Gambar 4.23. Titik-titik Sudut Setiap Detektor Sudut



Gambar 4.24. Kenaikan Jumlah Titik Sudut Masing-masing *Noise*

4.2. Pembahasan

4.2.1. Detektor Sudut Moravec

A. Algoritma

1. Untuk setiap piksel (*x*, *y*) dalam citra dihitung varian intensitas dari pergeseran (*u*, *v*) sama dengan:

$$V_{u,v}(x,y) = \sum_{\substack{\forall a,b \\ in\ the \\ window}} [I(x+u+a, y+v+b - I(x+a, y+b)]^2 \qquad (4.1)$$

dimana pergeseran (*u*, *v*) yang dipertimbangkan adalah:

(*u*, *v*) ∈{(*1*,*0*), (*1*,*1*), (*0*,*1*), (*-1*,*1*), (*-1*,*0*), (*-1*,*-1*), (*0*,*-1*), (*1*,*-1*)}

2. Konstruk peta seluruh sudut dangan menghitung *C*(*x*,*y*) untuk setiap piksel (*x*,*y*):

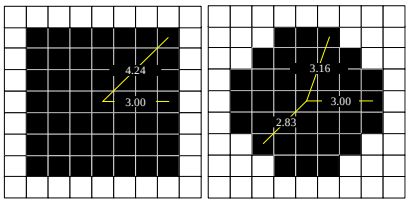
$$C(x, y) = \min(V_{(u,y)}(x, y)) \qquad (4.2)$$

3. *Threshold interest map* dengan mengatur semua *C*(*x*, *y*) di bawah *threshold T* ke nol (0)
4. Persiapkan kondisi *non maximal suppression* untuk menemukan lokal maksima.
5. Menghilangkan duplikasi titik-titik sudut dengan menggunakan prosedur pencarian dari lokal maksima dari fungsi *non maximal suppression*.

6. Titik sudut yang merupakan sudut sebenarnya di dalam peta sudut adalah semua elemen yang saling berkoresponden dan bernilai tidak sama dengan nol (0).

B. Respon *Noise*

Moravec menggunakan jendela persegi dan biner. Untuk mencapai perkiraan yang lebih akurat dari variasi intensitas lokal, maka dibutuhkan jendela melingkar sedemikain rupa sehingga jarak *Euclidean* dari pusat piksel ke tepi jendela adalah sama di semua arah. Bahkan pada grid diskrit, penggunaan jendela melingkar akan menghasilkan peningkatan estimasi variasi intensitas lokal. Gambar 4.4, mengilustrasikan jendela persegi 7x7 dan jendela melingkar dengan diameter 7 dan menunjukkan perbedaan maksimum dalam jarak *Euclidean* untuk kedua jendela tersebut. Estimasi variasi intensitas lokal ditingkatkan dengan menempatkan bobot lebih pada perbedaan intensitas pasangan piksel yang berada dekat dengan pusat jendela. Secara intuitif, perbedaan pasangan piksel yang dekat dengan pusat adalah indikasi yang lebih baik dari varian lokal sehingga akan memiliki efek lebih dekat pada perkiraan. Jendela yang sering digunakan adalah jendela *gaussian*.



Gambar 4.4. Pergurangan Jarak *Euclidien*

4.2.2. Detektor Sudut Susan

A. Algoritma

1. Menempatkan titik pusat dari masker lingkaran ke dalam kernel.
2. Hitung jumlah piksel-piksel yang memiliki intensitas sama atau mirip pada *nucleus* dengan menggunakan sebuah masker lingkaran menggunakan persamaan (3). Titik dari piksel-piksel yang ditentukan oleh USAN adalah; 1) titik di dalam masker, 2) titik pusat kernel, 3) titik intensitas, 4) perbedaan intensitas *threshold* dan 5) hasil perbandingan.

$$c(r, r_0) = e^{\left\{-\frac{I(r)-I(r_0)}{t}\right\}^2} \tag{4.3}$$

3. Pengurangan dari *threshold* ukuran geometrik USAN, untuk mendapatkan sudut pada citra dengan proses sebagai berikut; 1) konsep Susan, nilai awal dari respon tepi adalah area terkecil pada USAN merupakan respon tepi terbesar, 2) *g* adalah *threshold* geometrik, 3) *n* adalah jumlah piksel-piksel dalam USAN yaitu area USAN.

$$R(r_0) = \begin{cases} g - n(r_0), nr \leq g \\ 0, \qquad\qquad \geq g \end{cases} \tag{4.4}$$

$$n(r_0) = \sum_{r \in c(r_0)} c(r, r_0) \tag{4.5}$$

4. Menentukan titik pusat geometrik dari area-area USAN dan kedekatan masing-masing serta idetifikasi titik-titik palsu.
5. Pilih titik-titik spesifik dengan pencarian lokal maksima dari fungsi respon *non maximum suppression*.

B. Respon *Noise*

Penyaringan *noise* operator Susan, prinsipnya hampir sama dengan teknik *noise filtering* yang ada, melindungi struktur citra dengan menghaluskan bagian-bagian yang berdekatan yang terbentuk dari bagian-bagian area yang sama sebagai piksel pusat. Dengan asumsikan nilai konstan, di sini nilai hanya diasumsikan mendekati atau kira-kira konstan. Area miring (*slope*) dihitung dengan mengambil rata-rata semua dari piksel dalam area USAN. Dengan hal ini akan memberikan jumlah maksimal bagian-bagian yang bedekatan dan diambil nilai rata-rata.

Proses penghalusan (*smoothing*) tidak terpengaruh ketika USAN pada posisi miring, karena beberapa variasi di area USAN sudah melakukan proses penghalusan sebelum bobot penghalusan berkurang secara signifikan. Proses ini dapat menentukan batas kekuatan *noise*, pembobotan proses penghalusan dapat dihitung dengan menggunakan persamaan berikut.

$$c(\vec{r},\vec{r}_0) = e^{-\left(\frac{I(\vec{r})-I(\vec{r}_0)}{t}\right)} \quad (4.6)$$

Gaussian dalam domain kecerahan digunakan untuk proses menghaluskan. Ini berarti bahwa *filter* Susan merupakan *filter sigma* (kecerahan dan domain spasial). Ketergantungan *filter* Susan pada parameter *t* tidak penting, biasanya tidak bervariasi, hanya perlu penyesuaian nilai *t* untuk citra dalam kondisi sangat buruk.

4.2.3. Detektor Sudut Harris

A. Algoritma

1. Menghitung derivatif *x* dan *y* dari citra

$$I_x = G_\sigma^x * I \quad (4.7)$$

$$I_y = G_\sigma^y * I \quad (4.8)$$

2. Menghitung produk dari derivatif pada setiap piksel

$$I_{x2} = I_x . I_x \quad (4.9)$$

$$I_{y2} = I_y . I_y \quad (4.10)$$

$$I_{xy} = I_x . I_y \quad (4.11)$$

3. Menghitung jumlah dari produk dari derivatif setiap piksel

$$S_{x2} = G_{\sigma t} * I_{x2} \quad (4.12)$$

$$S_{y2} = G_{\sigma t} * I_{y2} \quad (4.13)$$

$$S_{xy} = G_{\sigma t} * I_{xy} \quad (4.14)$$

4. Menentukan setiap piksel (*x*, *y*) dalam matrik

$$H(x,y) = \begin{bmatrix} S_{x2}(x,y) & S_{xy}(x,y) \\ S_{xy}(x,y) & S_{y2}(x,y) \end{bmatrix} \quad (4.15)$$

5. Menghitung respon dari detektor untuk setiap piksel

$$R = Det(H) - k(Trace(H))^2 \quad (4.16)$$

Dimana, nilai *R* positif untuk sudut, nilai *R* negatif untuk tepi dan nilai *R* kecil untuk bidang datar, serta *k* merupakan konstanta emperik dengan nilai *k* = 0.04-0.06

6. *Threshold* pada nilai *R*, dan hitung *nonmaxima suppression*

B. Respon Anisotropik dan *Noise*

Banyak algoritma menggunakan gradien atau derivatif order tinggi yang akan cenderung sensitif terhadap *noise*. Kita akan melihat efek pada respon Plessey terhadap *salt noise* pada latar belakang objek warna hitam. Setiap piksel terpengaruh oleh *salt noise* akan memiliki gradien yang besar ke segala arah. Artinya, *salt noise* pada dasarnya adalah pixel terisolasi sehingga memiliki variasi intensitas tinggi ke segala arah yang tepat bagaimana sudut didefinisikan di bawah kedua operator antara Moravec dan Harris. Efek *noise* berkurang dengan meningkatkan ukuran jendela *gaussian* karena ini menyebabkan lebih banyak piksel dipertimbangkan ketika menghitung gradien, sehingga mengurangi bobot relatif dari setiap piksel yang terpengaruh oleh *noise*. Namun, meningkatkan ukuran jendela Gaussian meningkatkan kebutuhan komputasi dari algoritma.

4.2.4. Detektor Sudut FAST

A. Algoritma

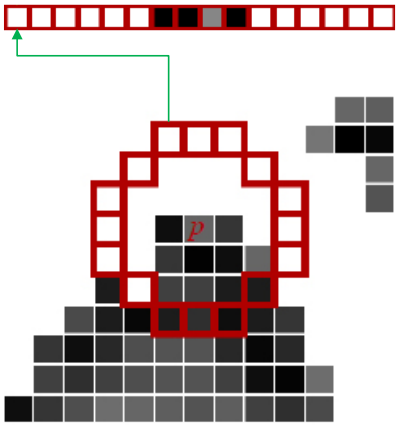
1. Pilih piksel (*p*) dalam citra, dengan asumsi bahwa intensitas dari piksel adalah *Ip*. Piksel ini yang diidentifikasi merupakan titik ketertarikan atau bukan.
2. Lakukan pengaturan *threshold* nilai intensitas (*T*).
3. Gunakan lingkaran dari 16 piksel yang mengelilingi piksel *p* (menggunakan algoritma Bresenham).
4. *N* merupakan piksel-piksel diluar 16 piksel yang menjadi lebih baik atau di bawah *Ip* dengan nilai *T*, jika piksel diperlukan untuk dideteksi menjadi satu *interest point* (digunakan N=12 pada algoritma versi pertama). Lakukan komparasi intensitas dari piksel 1, 5, 9 dan 13 dari lingkaran *Ip*. Seperti terlihat gambar 11, minimal tiga dari empat piksel harus memenuhi kreteria *threshold* maka dengan demikian *interest point* akan didapatkan.
5. Jika tiga dari empat piksel (I_1, I_5, I_9, I_{13}) tidak sama termasuk atau dibawah *Ip* + *T*, kemudian *P* adalah bukan merupakan interest point (sudut). Dalam hal ini maka piksel *p* ditolak sebagai sebuah *interest point*. Jika tiga dari empat piksel

termasuk atau dibawah *Ip* + *T*, maka periksa 16 piksel secara keseluruhan dan periksa jika 12 piksel tidak termasuk dalam kreteria.

6. Ulangangi prosedur ini untuk semua piksel dalam citra.

B. Pendekatan *Machine Learning*

1. Tentukan satu citra untuk percobaan.
2. Gunakan algoritma FAST pada citra untuk mendeteksi *interest point* dengan mengambil satu piksel dan mengevaluasi keseluruahan dari 16 piksel dalam lingkaran.
3. Untuk setiap pixel *p*, menyimpan 16 piksel-piksel yang berdekatan di sekitarnya, sebagai vektor (gambar 4.25).



Gambar 4.25. 16 Nilai Sekitarnya Piksel *p* Disimpan dalam Bentuk Vektor

4. Ulangi prosedur ini untuk semua piksel pada citra. Ini merupakan vektor *P* yang berisikan semua data (perbedaan antara piksel *p* dan vektor *P*).
5. Setiap nilai (salah satu dari 16 piksel disebut degan *x*) dalam vector, dapat berupa salah satu dari kondisi berikut; 1) lebih gelap dari *p*, 2) lebih cerah dari *p* dan 3) sama dengan *p* dan secara matematis dapat dituliskan dengan persamaan berikut.

$$S_{p\to x} = \begin{cases} d, & I_{p\to x} \le I_p - t \ (darker) \\ s, & I_p - t < I_{p\to x} < I_p + t \ (similiar) \\ b, & I_p + t \le I_{p\to x} \ (brighter) \end{cases} \quad (4.17)$$

Dimana $S_{p\to x}$ adalah kondisis, $I_{p\to x}$ adalah intensitas dari piksel *x* dan *t* adalah *threshold*.

P merupakan himpunan dari semua piksel-piksel dalam sebuah citra. Tentukan *x* bagian *P*. Misalkan *P* adalah himpunan semua piksel dalam semua citra. Memilih x sebagai partisi *P* menjadi tiga himpunan bagian; *Pd*, *Ps* dan *Pb*, di mana; *Pb* = {*p* ∈ *P*: *Sp* → *x* = *b*}, dan *Pd* dan *Ps* didefinisikan sama. Dengan kata lain, pilihan yang diberikan *x* digunakan untuk membagi data menjadi tiga himpunan. Himpunan *Pd* berisi semua titik-titik piksel *x* yang lebih gelap dari piksel pusat dengan *thresholad t*, *Pb* berisi titik-titik piksel yang

lebih terang dari pixel pusat oleh dengan thresholad *t*, dan *Ps* berisi titik-titik piksel yang tersisa yang mirip dengan piksel pusat.

6. Tergantung pada kondisi-kondisi seluruh vektor *P* akan dibagi menjadi tiga himpunan bagian, *Pd*, *Ps*, *Pb*.
7. Tentukan variabel *Kp true* jika *p* merupakan *interest point* dan *false* jika *p* tidak *interest point*.
8. Gunakan algoritma ID3 (*decision tree classifier*) untuk *query* setiap bagian menggunakan variabel *Kp* untuk mengetahui kelas yang tepat.
9. Algoritma ID3 bekerja pada prinsip minimisasi entropi. *Query* 16 piksel sedemikian rupa menyatakan bahwa kelas yang benar ditemukan (*interest point* atau tidak) dengan jumlah minimum dari *query*. Atau dengan kata lain, pilih piksel *x*, yang memiliki sebagian besar informasi tentang piksel *p*. Entropi untuk kumpulan *P* dapat dituliskan secara matematis sebagai berikut.

$$H(P) = (c + \bar{c})\log_2(c + \bar{c}) - c\,\log_2 c - \bar{c}\,\log_2 \bar{c} \qquad (4.18)$$

Dimana, $c = |\{p|K_p \text{ is true}\}|$ $(number\ of\ corners)$
dan, $\bar{c} = |\{p|K_p \text{ is false}\}|$ $(number\ of\ non\ corners)$

Pilihan *x* kemudian menghasilkan pertambahan informasi:

$$H_g = H(P) - H(P_d) - H(P_s) - H(P_b) \qquad (4.19)$$

C. *Non Maximal Suppression*

Non maximal suppression untuk menghapus sudut yang berdekatan, bertujuan untuk mengatasi proses deteksi *interest point* yang lokasinya berdekatan antara satu titik dengan titik lain. algoritma ini dijelaskan sebagai berikut.

1. Hitunglah skor dari fungsi *V* untuk masing-masing *interest point* yang terdeteksi. Fungsi skor didefinisikan sebagai jumlah perbedaan mutlak antara piksel-piksel dalam busur yang berdekatan dan pusat piksel.
2. Pertimbangkan dua *interest point* yang berdekatan, dengan membandingkan nilai *V* masing-masing *interest point*.
3. Buang salah satu yang nilai *V* nya lebih rendah.

Proses ini dapat ditulis secara matematis sebagai berikut.

$$V = \max\begin{cases}\Sigma\ (\text{pixel values} - p) \text{ if } (\text{value} - p) > t \\ \Sigma\ (p - \text{pixel values}) \text{ if } (p - \text{value}) > t\end{cases}$$

$$V = \max\left\{\sum_{x\in S_{bright}} \left|I_{p\to x} - I_p\right| - t,\ \sum_{x\in S_{dark}} \left|I_p - I_{p\to x}\right| - t\right\} \qquad (4.20)$$

dimana,

$$S_{bright} = \{x|I_{p\to x} \geq I_p + t\}$$
$$S_{dark} = \{x|I_{p\to x} \leq I_p + t\}$$

dimana, *p* adalah piksel pusat, *t* adalah *threshold* untuk nilai deteksi dan piksel yang sesuai dengan *N* piksel yang berdekatan dalam lingkaran.

4.2.5. Detektor Sudut Eigen

A. Algoritma

1. Menghitung gradien seluruh citra
2. Untuk setiap titik citra p;
   a) Bentuk matriks *C* melalui $(2N + 1)$ x $(2N + 1)$ titik tetangga Q,
   b) Hitung $R(x_p, y_p)$,
   c) Jika $R(x_p, y_p) > \tau_{KLT}$, simpan kordinat dari *p* ke dalam list *L*.
3. Melalui daftar *L* turunkan level dari *R* (*xp*, *yp*) jika tidak termasuk dalam ruang pemisahan minimal setiap sudut yang dipilih sebelumnya, kemudian pilih; jika tidak maka membuangnya.

4.2.6. Detektor Sudut Forstner

Dalam prakteknya algoritma detektor ini bergantung pada sudut yang ideal dan garis singgung lintas pada satu titik. Secara matematis dapat dilihat pada persamaan-persamaan dibawah ini.

Persamaan garis singgung $T_{x'}(x)$ pada piksel x' sebagai berikut:

$$T_{x'}(x) = \nabla I(x')^T (x - x') = 0 \tag{4.21}$$

Dimana $\nabla I(x')^T = \left[I_x, I_y\right]^T$ adalah vektor gradien dari citra *I* pada x'

Titik x_0 merupakan titik paling dekat dengan semua garis singgung di jendela *N* adalah:

$$x_0 = \underset{x \in \mathbb{R}^{2x2}}{\operatorname{argmin}} \int_{x' \in N} T x'^{(x)^2} dx' \tag{4.22}$$

Jarak dari x_0 dengan garis singgung $T_{x'}$ dibobot dengan besarnya gradien, sehingga garis singgung melewati piksel dengan nilai gradien yang paling tinggi. x_0 dapat dihitung dengan persamaan berikut.

$$x_0 = \underset{x \in \mathbb{R}^{2x2}}{\operatorname{argmin}} (x^T A x - 2x^T b + c) \tag{4.23}$$

$A \in \mathbb{R}^{2x2}, B \in \mathbb{R}^{2x1}, c \in \mathbb{R}$ didefinisikan sebagai:

$$A = \int \nabla I(x') \nabla I(x')^T dx' \tag{4.24}$$
$$b = \int \nabla I(x') \nabla I(x')^T x' dx' \tag{4.25}$$
$$c = \int x'^T \nabla I(x')^T x' dx' \tag{4.26}$$

Meminimalkan persamaan ini dapat dilakukan dengan membedakan terhadap *x* dengan menentukan nilai *x* sama dengan nol (0), dapat dituliskan dengan persamaan berikut.

$$2Ax - 2b = 0 \Rightarrow Ax = b \tag{4.27}$$

Dimana $A \in \mathbb{R}^{2x2}$ adalah struktur tensor. Sebagai solusi untuk persamaan ini adalah *A* harus dibalik, yang menyiratkan bahwa *A* harus peringkat penuh (rank 2). persamaannya adalah sebagai berikut.

$$x_0 = A^{-1} b \tag{4.28}$$

Hanya ada di mana sudut yang sebenarnya ada di jendela *N*.

Pemilihan skala otomatis untuk metode penentuan lokasi sudut ini telah disampaikan oleh Lindeberg dengan meminimalkan sisa proses normalisasi dari skala, dihitung dengan persamaan sebagai berikut.

$$\tilde{d}_{min} = \frac{c - b^T A^{-1} b}{\text{trace A}} \quad (4.29)$$

Dengan demikian, metode ini memiliki kemampuan secara otomatis untuk menyesuaikan tingkat skala untuk menghitung gradien citra pada tingkat *noise* dalam data citra, dengan memilih tingkat skala kasar untuk data *noise* pada citra dan tingkat skala yang lebih halus dan dekat dengan struktur sudut yang ideal.

Catatan:

- *C* dapat dilihat sebagai sisa dalam *least square* dan dihitung, jika *C* = 0 maka tidak ada kesalahan.
- algoritma ini dapat dimodifikasi untuk menghitung pusat fitur melingkar dengan mengubah garis-garis singgung ke garis yang normal.

V. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan data-data hasil pengujian dari enam algoritma detektor sudut pada sebuah citra; 1) citra tanpa *noise* yang memiliki 217 titik sudut sebenarnya, dapat disimpulkan sebagai berikut; detektor Moravec mendeteksi 204 titik-titik sudut dengan waktu proses 0.81 detik, detektor Susan mendeteksi 100 titik-titik sudut dengan waktu proses 16.20 detik, detektor Harris mendeteksi 571 titik-titik sudut dengan waktu proses 0.61 detik, detektor FAST mendeteksi 538 titik-titik sudut dengan waktu proses 0.25 detik, detektor Eigen mendeteksi 796 titik-titik sudut dengan waktu proses 0.70 detik dan detektor Forstner mendeteksi 217 titik-titik sudut dengan waktu proses 2.77 detik. Secara berurutan detektor Forstner memiliki tingkat akurasi tertinggi dalam mendeteksi titik-titik sudut sebenarnya pada sebuah citra tanpa *noise*, kemudian detektor Moravec, Susan dan tingkat akurasi terendah adalah detektor Harris, Eigen, FAST. Sementara dari waktu proses pendeteksian titik sudut detektor FAST memiliki waktu paling cepat kemudian detektor Harris, Eigen dan Moravec serta yang paling lama adalah detektor Susan dan Forstnet. 2) citra yang memiliki *noise* dapat disimpulkan sebagai berikut; detektor Moravec mengahasilkan 1,000 titik sudut pada citra *noise* (*gaussian*, *poisson*, *salt and pepper*, *speckle*), rata-rata waktu proses pendeteksian sebesar 2,19 detik. Detektor Susan menghasilkan 100 titik sudut pada citra *noise* (*gaussian*, *poisson*, *salt and pepper* dan *speckle*) dengan rata-rata waktu proses pendeteksian sebesar 23,99 detik. Detektor Harris menghasilkan 3.521 titik sudut pada citra *noise gaussian*, 2.125 titik sudut pada citra *noise poisson*, 1.559 titik sudut pada citra *noise salt and pepper*, dan 9.908 titik sudut pada citra *noise speckle*, rata-rata waktu proses pendeteksian 0,91 detik. Detektor FAST menghasilkan 2.461 titik sudut pada citra *noise gaussian*, 782 titik sudut pada citra *noise poisson*, 2.473 titik sudut pada citra *noise salt and pepper*, 841 titik sudut pada citra *noise speckle*, dengan waktu rata-rata proses pendeteksian sebesar 0,26 detik. Detektor Eigen menghasilkan 7.891 titik sudut pada citra *noise gaussian*, 7.970 titik sudut pada citra *noise poisson*, 2.238 titik sudut pada citra *noise salt and pepper* dan 9.638 titik sudut pada citra *noise speckle*, dengan rata-rata waktu

proses pendeteksian sebesar 1,23 detik. Detektor Frostner menghasilkan 821 titik sudut pada citra *noise gaussian*, 868 titik sudut pada citra *noise poisson*, 940 titik sudut pada citra *noise salt and pepper* dan 854 titik pada citra *noise speckle*, degan rata-rata waktu pendeteksian sebesar 6,07 detik. Akurasi hasil deteksi titik-titik sudut sebenarnya pada citra tanpa *noise* secara berurutan sebagai berikut; detektor sudut Frostner memiliki tingkat akurasi tertinggi, kemudian detektor sudut Susan dan selanjutnya detektor sudut Moravec. Untuk tiga detektor sudut lainnya (Harris, FAST, Eigen) merupakan detektor sudut dengan akurasi terendah. Sementara hasil uji detektor sudut pada citra yang memilki *noise* menunjukkan bahwa; a) seluruh detektor sudut tidak mampu menemukan titik-titik sudut dengan tepat, b) seluruh detektor sudut tidak akurat dalam menunjukkan lokasi titik sudut, c) hanya detektor Moravec dan Susan yang stabil terhadap perulangan, d) seluruh detektor sudut tidak stabil atau sangant sensitif terhadap semua tipe *noise*. Hasil pengujian dalam penelitian ini memperlihatkan bahwa seluruh detektor sudut sangat sensitif terhadap *noise*, dengan pengertian lain bahwa tingkat akurasi hasil pendeteksian setiap detektor sudut akan sangat dipengaruhi oleh adanya *noise*.

5.2. Saran

Bagi para peneliti yang sangat tertarik pada bidang komputer visi dan pengolahan citra khususnya *stereo matching*, *image registration*, *stitching of panoramic photograps*, *object detection*/*recognition*, *motion tracking* dan *robot navigation* mungkin dapat melakukan pengujian beberapa detektor sudut lain seperti; Beaudet, Kitchen & Rosenfeld, Deriche, Wang & Brady, CSS, Trajkovic & Hedley (4 atau 8-neighbours), Zheng & Wang. Ataupun mengkombinasikan serta menambahkan metoda dalam algoritma detektor sudut guna pengembangan selanjutnya guna mengetahui dan menemukan detektor sudut dengan performa terbaik dalam melakukan pendeteksian titik-titik sudut dalam sebuah citra yang memiliki *noise*.

DAFTAR PUSTAKA

[1] Pattar, S.Y, Study of Corner Detection Algorithms and Evaluation Methods, *International Journal of Innovative Research in Science*, *Engineering and Technology*, vol. 4, pp. 2780–2787, May 2015.

[2] Wei Jianhua, Kong Xiangnan, Corner Detection Using Multi-directional Gabor Filters , *Journal of Fiber Bioengineering and Informatics*, vol.8, pp 615–624, 2015.

[3] Suran Kong, QR Code Image Correction based on Corner Detection and Convex Hull Algorithm, *Journal Of Multimedia*, vol. 8, pp. 662–667, December 2013.

[4] Dey Nilanjan, Pradipti Nandi, Nilanjan Barman, Debolina Das and Subhabrata Chakraborty, A Comparative Study between Moravec and Harris Corner Detection of Noisy Images Using Adaptive Wavelet Thresholding Technique, *International Journal of Engineering Research and Applications*, vol.2, pp. 601–606, 2012.

[5] Mahesh, Subramanyam.M.V, Corner Detection using Curvelet and Harris Algorithm, *International Journal of Computer Science And Technology*, vol.3, pp.612–616, 2012.

[6] Bhatia Nitin, and Megha Chhabra, Accurate Corner Detection Methods using Two Step Approach, *Global Journal of Computer Science & Technology*, vol.11, pp. 25–29, April 2011.

[7] Zhao Fuqing, and Chunmiao Wei, An automated x-corner detection algorithm (AXDA), *Journal of Software*, vol.28, pp.791–797, 2011.
[8] E. Rosten, R. Porter and T. Drummond, Faster and better: A machine learning approach to corner detection [J], IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.32, pp. 105-119, 2010.
[9] Chen Jie, Li-hui Zou, Juan Zhang and Li-hua Dou, The Comparison And Application Of Corner Detection Algorithms, *Journal Of Multimedia*, vol. 4, pp.435–441, December 2009.
[10] Parks, D. and J.P. Gravel, Center for Intelligent Machines, *International Journal of Computer Vision*, McGill University, 2004
[11] F. Shen and H. Wang, Real Time Gray Level Corner Detector, 6th International Conference on Control, Automation, Robotics and Vision (ICARCV2000), 2000.
[12] Harris, C and Stephens, M.J. A combined corner and edge detector. In *Proceedings of the Fourth Alvey Vision Conference*, pp. 147–151, UK, 1988.
[13] E. Rosten and R. Porter, T. Drummond, Faster and better: A machine learning approach to corner detection [J], IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.32, pp. 105-119, 2010.
[14] Yang Yang, Zhang Tianwen, Assessing criterion of corner finders, *Journal of Harbin Institute of Technology*, vol. 30, pp.7-10, 1998.
[15] S. Smith, J. Brady, SUSAN-A new approach to lowlevel image processing, *International Journal of Computer Vision*, vol. 23, pp.45-48, 1997.
[16] Moravec, H, Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover, Tech Report CMU-RI-TR-3, Carnegie-Mellon University, Robotics Institute, September 1980.
[17] H. P. Moravec. Towards Automatic Visual Obstacle Avoidance. Proc. 5th International Joint Conference on Artificial Intelligence, pp. 584, 1977.