Ekstraksi Objek pada Citra Radar FM-CW dengan Metode DBSCAN

Object Extraction on FM-CW Radar Image Using DBSCAN Method

Vicky Zilvan
UPT LPSN – Lembaga Ilmu Pengetahuan Indonesia (LIPI), Jl. Ranggamalela No. 11 Bandung, Indonesia
Email:vicky.zilvan@lipi.go.id

Abstract

This paper proposed design and implementation object extraction on FM-CW radar to solve radar image quality problem. Density based spatial clustering of applications with noise (DBSCAN) technique is used to extract objects from input data. The result of this research is a design of object extraction by setting mintPts to 4 and eps to 4 as input parameters for DBSCN. Output of this design is simple data points as a result of object extraction which can resolve radar image quality problem. Furthermore, data points of object extraction result have good quality data because DBSCAN clustering technique has ability to separate noise data of input data.

Keywords: *image processing, FM-CW radar, object extraction, DBSCAN*

Abstrak

Makalah ini membahas rancang bangun dan implementasi ekstraksi objek pada radar FM-CW untuk mengatasi permasalahan kualitas citra yang ditangkap oleh radar. Teknik *clustering density based spatial clustering of applications with noise* (DBSCAN) digunakan untuk mengekstraksi objek dari data input. Hasil dari penelitian ini adalah rancang bangun ekstrasi objek dengan nilai *minPts* sebesar 4 dan nilai *eps* sebesar 4 sebagai parameter input untuk DBSCAN. Hasil dari rancang bangun ekstraksi objek adalah titik-titik data hasil ekstraksi objek yang lebih sederhana yang mampu mengatasi permasalahan kualitas citra yang ditangkap oleh radar. Selain itu, titik-titik data yang dihasilkan juga memiliki kualitas data yang lebih baik karena teknik *clustering* DBSCAN memiliki kemampuan untuk memisahkan *noise* dari data input.

Kata kunci: pengolahan citra, radar FM-CW, ekstraksi objek, DBSCAN

1. Pendahuluan

Image processing atau pengolahan citra merupakan bagian fundamental dalam suatu sistem radar modern disamping teknologi hardware yang digunakan. Bagian *image procesing* ini merupakan modul yang berfungsi untuk mengola sinyal-sinyal analog yang diterima oleh radar menjadi sinyal-sinyal digital yang ditampilkan melalui modul PPI (*Plan position indicator*) Display.

Kendala yang dihadapi oleh pengguna radar dalam menganalisis dengan cara melihat modul PPI Display adalah kesulitan dalam melihat dan membedakan target dari data yang ditampilkan di PPI Display. Hal ini disebabkan pada PPI Display ditampilkan keseluruhan data yang ditangkap oleh radar, baik berupa data objek target, data objek lingkungan sekitar target, maupun data *noise* seperti gelombang air laut. Selain itu, dengan data mentah yang ditampilkan pada PPI Display menyulitkan proses analisis selanjutnya seperti untuk keperluan *object tracking*.

Untuk mengatasi permasalahan tersebut, maka diperlukan suatu metode yang mampu mengekstraksi objek target dari data mentah pada PPI Display sehingga dapat membatu dan mempermudah pengguna radar dalam menganalisis data yang ditangkap oleh radar.

Ektraksi objek dalam suatu sistem radar berfungsi untuk memusatkan sekumpulan titik data dari suatu objek yang sama menjadi satu bagian. Objek-objek yang telah diekstraksi dari sekumpulan data, akan lebih mudah untuk diamati oleh pengguna radar. Selain itu, hasil dari ekstrasi objek juga sangat membantu dalam tahap proses selanjutnya seperti *object tracking*.

Received: 20 Mei 2015; Revised: 19 Juni 2015; Accepted: 26 Juni 2015 ; Published online: 10 Juli 2015

Salah satu teknik untuk melakukan pemusatan data dari sekumpulan data adalah *clustering*. Telah banyak metode *clustering* yang telah dikembangkan dan dipakai dalam beberbagai bidang penelitian. Beberapa metode *clustering* yang ada diantaranya adalah K-means [1], PAM [2], CLARANS [3], DBSCAN [4], CURE [5], and ROCK [6]. Diantara metode-metode tersebut, DBSCAN merupakan salah satu metode yang secara luas telah banyak diaplikasikan dalam berbagai bidang ilmu pengetahuan karena kesederhanaan serta kemampuan untuk mendeteksi *cluster* dengan ukuran dan bentuk yang berbeda. DBSCAN telah diaplikasikan di berbagai bidang ilmu pengetahauan seperti teknik sipil (pengelompokan jaringan infrastruktur sipil spasial [7]), kimia [8], spektroskopi (pengelompokan spektrum massa partikel tunggal [9] dan aerosol waktu-of-flight mass spektrometri [10]), ilmu sosial (pengelompokan serangga berdasarkan feromon kimia Data ([11],[12])), dan diagnosa medis berdasarkan gambar medis (untuk mendeteksi pola atrofi otak [13] dan untuk mendeteksi lesi kulit ([14],[15])). DBSCAN juga dapat diterapkan dalam bidang penginderaan jauh untuk melakukan segmentasi gambar tiga dimensi (gambar multi spektral) [16].

Dengan kesederhanaan pengaplikasian dari DBSCAN dan juga telah banyak digunakan dalam berbagai bidang ilmu pengetahuan, maka pada tulisan ini metode *clustering* DSBCAN dipilih sebagai metode yang digunakan untuk memperoleh ekstraksi objek dari data pantul radar sebagai bagian dari pengolahan citra radar yang diharapkan dapat membantu pengguna radar dalam menganalisis data yang ditangkap oleh radar.

Pada tulisan ini, ektraksi objek diimplementasikan pada radar pengawas pantai jenis FM-CW hasil penelitian Pusat Penelitian Elektronika dan Telekomunikasi (PPET), Lembaga Ilmu Pengetahuan Indonesia (LIPI).

2. Tinjauan Pustaka

2.1. DBSCAN: *Density Based Spatial Clustering of Application with Noise*

DBSCAN merupakan algoritma yang didesain oleh ester et.al pada tahun 1996 dapat mengidentifikasi kelompok-kelompok dalam kumpulan data spasial yang besar dengan melihat kepadatan lokal dari elemen-elemen database, dengan hanya menggunakan satu parameter input. DBSCAN juga dapat menentukan apakah informasi diklasifikasikan sebagai *noise* atau outlier. Disamping itu, proses kerja DBSCAN cepat dan sangat baik untuk berbagai macam ukuran database - hampir linear [17].

Dengan menggunakan distribusi kepadatan dari titik-titik dalam database, DBSCAN dapat mengkategorikan titik-titik tersebut ke dalam kelompok-kelompok terpisah yang menandakan kelas-kelas yang berbeda.

Proses komputasi DBSCAN berdasarkan enam definisi dan dua lemma sebagai berikut [17].

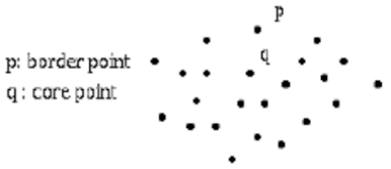
Definisi 1: (*Eps-neighborhood* dari suatu titik)

$$N_{Eps}(p) = \{q \in D | dist(p,q) < Eps\}$$

Untuk suatu titik yang masuk kedalam suatu *cluster*, titik tersebut setidaknya mempunyai satu titik lain yang letaknya lebih dekat ke titik tersebut dibandingkan dengan nilai *Eps*.

Definisi 2: (*directly density-reachable*)

Terdapat dua macam poin yang masuk dalam suatu *cluster*, yaitu titik yang terletak di pinggir *cluster* (*border points*) dan titik-titik yang terletak di pusat *cluster* (*core points*), sebagaimana yang ditunjukkan pada Gambar 1[4].



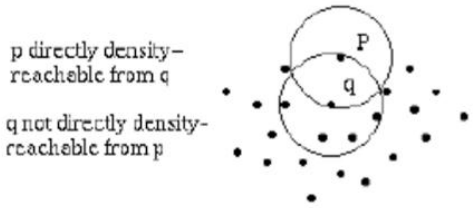
Gambar 1. *Border* dan *Core point*

Eps-neighborhood dari suatu *border point* cenderung memiliki titik yang lebih sedikit dibandingkan dengan *Eps-neighborhood* dari suatu *core point* [4]. *Border point* akan masih menjadi bagian dari suatu *cluster* apabila border point memiliki *Eps-neighborhood* dari suatu *core point* q sebagaimana yang ditunjukkan pada Gambar 2 [4].

$$p \in N_{Eps}(q)$$

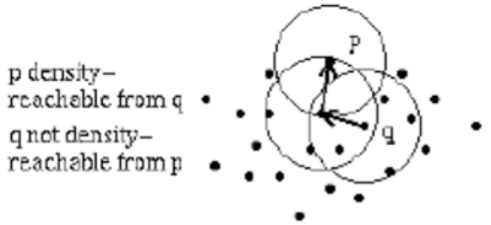
Agar titik *q* menjadi *core point*, titik tersebut perlu memiliki suatu jumlah minimum dari titik dalam *Epsneighborhood*-nya.

$$|N_{Eps}(q)| \geq \text{MinPts (kondisi } \textit{core point})$$



Gambar 2. Titik *p* merupakan *directly density-reachable* dari titik *q* tetapi tidak sebaliknya

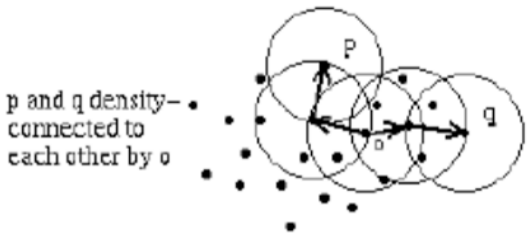
Definisi 3: (*density-reachable*)
Suatu titik *p* merupakan *density-reachable* dari suatu titik *q* berdasarkan *Eps* dan *MinPts* jika terdapat suatu rantai titik-titik $p_1 \ldots, p_n$, $p_1=q$, $p_n=p$ dimana p_{i+1} merupakan *directly density-reachable* dari p_i [4]. Pada Gambar 3 [4] ditunjukkan ilustrasi dari suatu *density-reachable*.



Gambar 3. Titik *p* merupakan *density-reachable* dari titik *q* tetapi tidak sebaliknya

Definisi 4: (*density-connected*)
Terdapat kasus ketika dua *border point* akan masuk ke dalam suatu *cluster* yang sama tetapi di mana dua *border point* tersebut tidak membagi suatu *core point* spesifik. Dalam situasi ini titik-titik tersebut tidak akan menjadi *density-reachable* satu sama lainnya. Namun harus ada *core point q* yang mana menjadi *density-reachable* dari kedua *border point* tersebut. Pada Gambar 4 [4] ditunjukkan bagaimana *density connectivity* bekerja.

Suatu titik *p* merupakan *density-connected* ke titik *p* berdasarkan *Eps* dan *MinPts* jika terdapat sauatu titik *o* dimana keduanya *p* dan *q* merupakan *density-reachable* dari *o* berdasarkan *Eps* dan *MinPts*.



Gambar 4. *Density connectivity*

Definisi 5: (*cluster*)
Jika suatu titik *p* merupakan bagian dari suatu *cluster C* dan titik *q* merupakan *density-reachable* dari titik *p* berdasarkan jarak tertentu dan jumlah minimum titik-titik dalam jarak tersebut, maka *q* juga merupakan bagian dari *cluster C*.

> $\forall$ *p*, *q*: Jika $p \in C$ and *q* is *density-reachable* dari *p* berdasarkan *Eps* and *MinPts*, maka $q \in C$.

Dua titik dimiliki oleh *cluster* yang sama *C*, dimana *p* merupakan *density-connected* ke *q* berdasarkan jarak tertentu dan jumlah minimum titik-titik dalam jarak tersebut.

> $\forall$ *p*, *q* $\in$ *C*: *p* merupakan *density-connected* ke *q* berdasarkan *Eps* dan *MinPts*.

Definisi 6: (*noise*)
Noise adalah sekumpulan titik-titik, dalam basis data, yang tidak masuk dalam *cluster*.

Lemma 1:
Suatu *cluster* dapat dibentuk dari salah satu *core point* dan akan selalu memiliki bentuk yang sama.

Lemma 2:
Jika *p* menjadi suatu *core point* dalam suatu *cluster C* dengan jarak minimum (*Eps*) dan suatu jumlah minimum titik dalam jarak tersebut (*MinPts*). Jika sekumpulan *O* merupakan *density-reachable* dari *p* berdasarkan *Eps* dan *MinPts* yang sama, maka *C* sama dengan sekumpulan *O*.

Algoritma DBSCAN adalah sebagai berikut [4]:

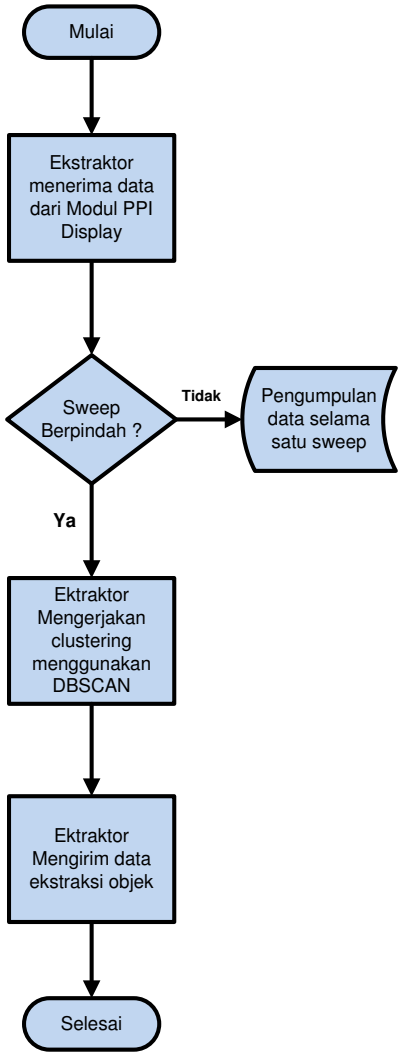
```
DBSCAN (SetOfPoints, Eps, MinPts)
 // SetOfPoints is UNCLASSIFIED
 ClusterId := nextId(NOISE);
 FOR i FROM 1 TO SetOfPoints.size DO
    Point := SetOfPoints.get(i);
    IF Point.ClId = UNCLASSIFIED THEN
      IF ExpandCluster(SetOfPoints,
            Point, ClusterId, Eps,
            MinPts) THEN
          ClusterId :=
              nextId(ClusterId)
      END IF
    END IF
 END FOR
END; // DBSCAN

ExpandCluster(SetOfPoints, Point,
      ClId, Eps, MinPts) : Boolean;
 seeds:=SetOfPoints.regionQuery(Point
       ,Eps);
 IF seeds.size<MinPts THEN // no core
          point
    SetOfPoint.changeClId(Point,
         NOISE);
    RETURN False;
 ELSE // all points in seeds are
      //density-reachable from
      //Point
    SetOfPoints.changeClIds(seeds,ClId
         );
    seeds.delete(Point);
    WHILE seeds <> Empty DO
      currentP := seeds.first();
      result :=
        SetOfPoints.regionQuery(curre
        ntP,Eps);

      IF result.size >= MinPts THEN
        FOR i FROM 1 TO result.size
              DO
          resultP := result.get(i);
```

```
            IF resultP.ClId IN
                {UNCLASSIFIED, NOISE}
                THEN
              IF resultP.ClId =
                   UNCLASSIFIED THEN
                seeds.append(resultP);
              END IF;
              SetOfPoints.changeClId(r
                   esultP,ClId);
            END IF: // UNCLASSIFIED or
               NOISE
         END FOR;
       END IF; // result.size >=
            MinPts
       seeds.delete(currentP) ;
    END WHILE; // seeds <> Empty
    RETURN True;
  END IF
END; // ExpandCluster
```

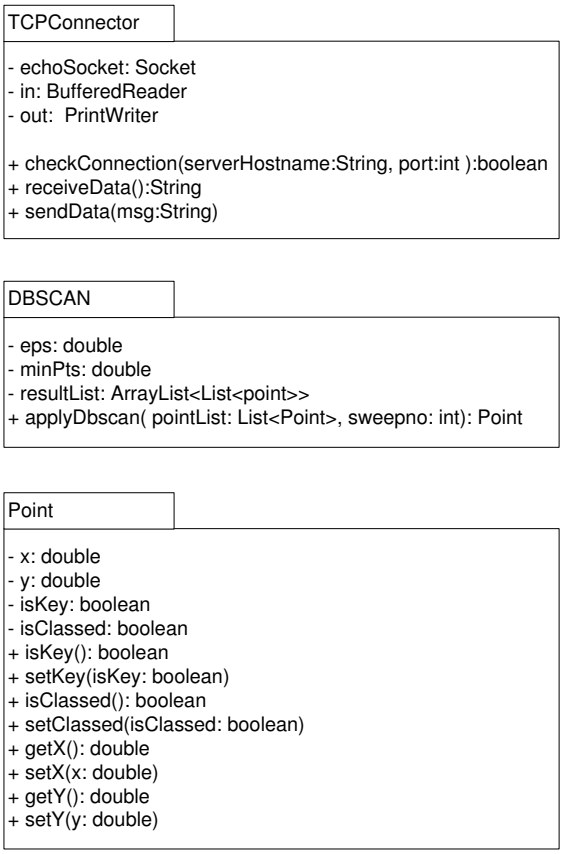
3. Perancangan



Gambar 5. *Flowchart* modul ekstraktor

Perancangan pengolahan citra untuk ekstraksi objek (yang selanjutnya disebut modul ekstrator) secara garis besar digambarkan pada aliran proses digambarkan dengan *flowchart* pada Gambar 5. Modul ekstraktor ini dirancangan dengan teknik berorientasi objek (*object oriented*) dalam bahasa pemrograman Java.

Selain perancangan aliran proses yang digambarkan sebelumnya, pada Gambar 3 ditunjukkan relasi-relasi antar class yang terdapat dalam modul ekstraktor. Relasi-relasi antar class ini digambarkan dalam UML Class diagram karena model yang dikembangkan ini diimplementasikan dalam program berorientasi objek. Penggambaran dalam *class diagram* ini memungkinkan kitu untuk menunjukkan isi statis, dan hubungan antara kelas. Selain itu, dalam *class diagram* dapat juga menunjukkan variabel anggota, dan fungsi anggota kelas. Selain itu, dapat menunjukkan apakah suatu kelas mewarisi dari yang lain, atau merupakan referensi dari kelas yang lain. Singkatnya, class diagram dapat menggambarkan semua dependensi *source code* antara kelas [18]. *Class diagram* dalam modul ekstraktor ini digambarkan pada Gambar 6.



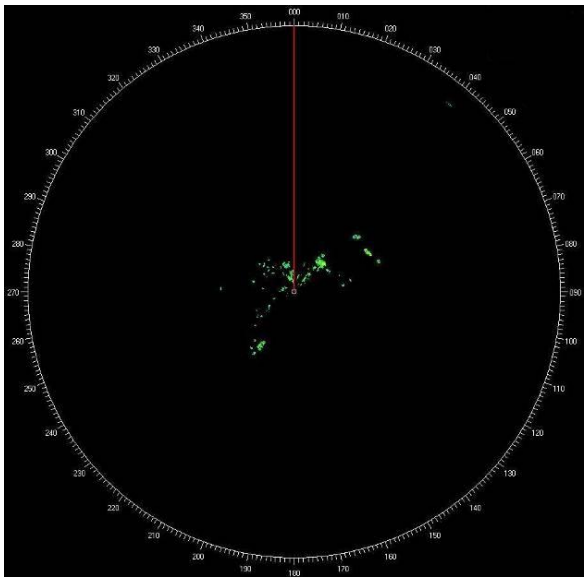
Gambar 6. *Class diagram* modul ekstraktor

4. Hasil dan Pembahasan

Pada penelitian ini, modul ekstraktor diimplementasikan dengan menggunakan bahasa pemrograman Java, dengan JRE versi 1.7.0_80 pada Netbeans IDE 8.0.1. Komputer yang digunakan dalam penelitian ini memiliki spesifikasi Processor Inter(R) Pentium(R) Dual

CPU E2160 @1.8 GHz 1.79 GHz, RAM 2.99 GB, dan menggunakan sistem operasi Windows 7.

Data yang digunakan pada penelitian berasal dari data radar selama satu putaran penuh pada range 1.5 nm. Pada Gambar 4 berikut, ditunjukkan data pantulan yang ditangkap oleh radar yang ditampilkan dalam modul PPI Display.



Gambar 7. Data pantul radar selama satu putaran pada modul PPI Display

Data awal sebagai data input pada penelitian ini didapat dari modul PPI Display (Gambar 7). Data dari modul display dikirim melalui koneksi soket ke modul ekstraktor yang diproses oleh *Class TCPConnector* dengan menggunakan fungsi *reciveData()* sebagaimana yang ditunjukan pada Gambar 3. Format data yang dikirim dari modul PPI Display adalah sebagai berikut:

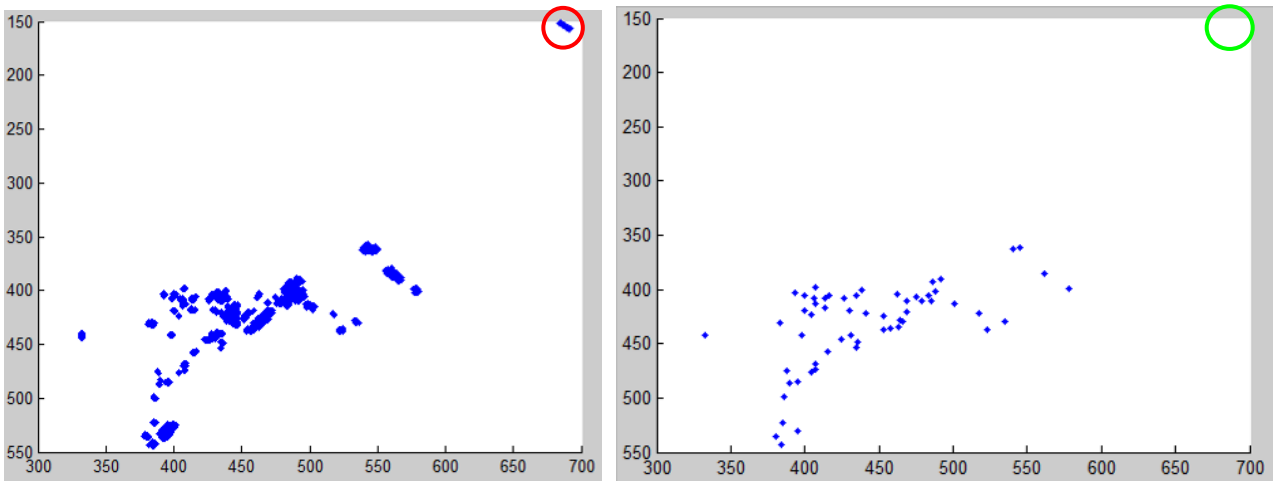
<no sweep>;<koordinat latitude>;<koordinat longitude>

Dimana <*no sweep*> adalah putaran radar ke-n, <*koordinat latitude*> adalah nilai koordinat latitude dari titik pantulan yang ditangkap radar, dan <*koordinat longitude* > adalah nilai koordinat longitude dari titik pantulan yang ditangkap radar.

Dari hasil satu putaran radar yang telah disimpan, sebagaimana yang terlihat pada Gambar 4, didapat data pantulan yang tertangkap radar sebanyak 2627 titik data. Selanjutnya, sebanyak 2627 titik data tersebut diproses oleh *Class point* dan *Class DBSCAN* untuk dilakukan proses *clustering* data dengan menggunakan parameter *minPts* dan parameter *eps*.

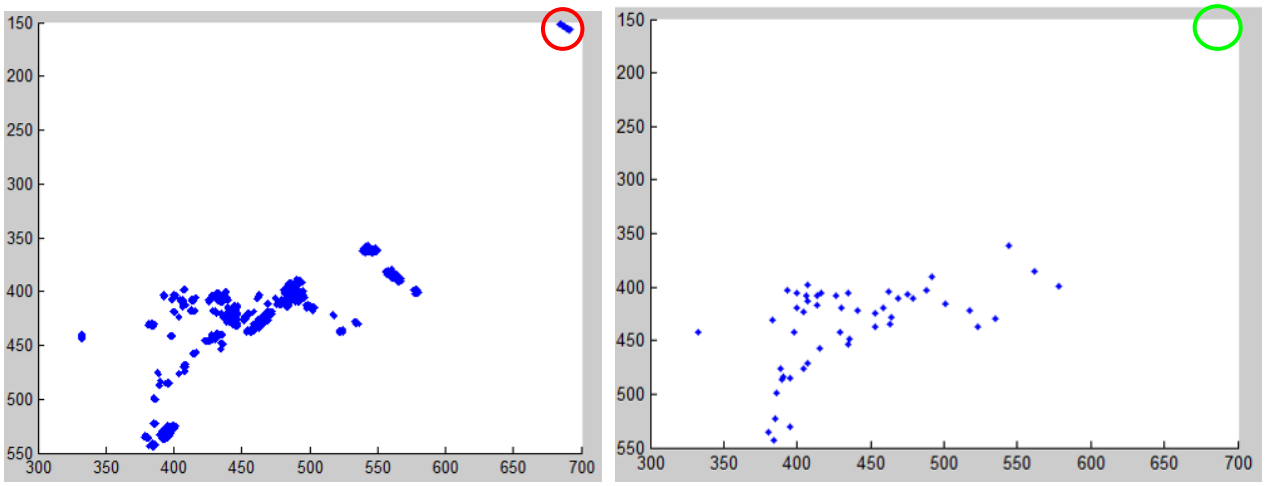
Untuk parameter *mintPts*, dalam penelitian Martin Ester et.al (1996) diketahui bahwa untuk perubahan parameter *mintPts* > 4 tidak bebeda signifikan dengan *mintPts*=4 dan untuk data dua dimensi menggunakan parameter *mintPts* sebesar 4 [4]. Oleh karena itu, karena pada penelitian ini data yang diolah merupakan data dua dimensi, sehingga parameter *mintPts* yang digunakan sebesar 4. Selanjutnya, untuk menentukan nilai parameter *eps* yang sesuai dengan karakteristik radar, pada Gambar 8, 9, 10, 11 dan 12 ditunjukkan hasil ekstraksi objek dari berbagai nilai *eps*.

Pada Gambar 8(b) dengan nilai *eps*=1, hasil ekstraksi objek yang dihasilkan belum optimal apabila dibandingkan dengan data sebelum dilakukan proses ekstraksi objek pada Gambar 8(a). Hal ini disebabkan karena hasil ekstraksi objek dengan nilai *eps*=1 terdapat data objek target (Gambar 8(a) yang ditandai dengan lingkaran merah) yang dianggap sebagai *noise*, sehingga data tersebut tidak diekstraksi sebagai objek target (Gambar 8(b) yang ditandai dengan lingkaran hijau).



(a) (b)

Gambar 8. Perbandingan data sebelum dilakukan ekstraksi objek(a) dengan hasil ekstraksi objek dengan nilai parameter *eps*=1(b)

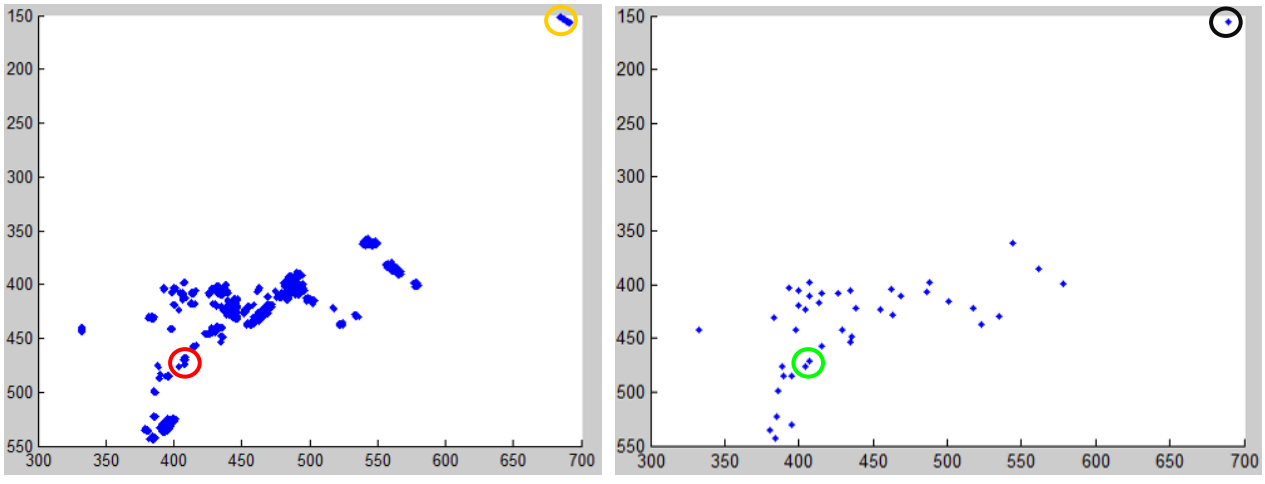


(a) (b)

Gambar 9. Perbandingan data sebelum dilakukan ekstraksi objek(a) dengan hasil ekstraksi objek dengan nilai parameter *eps*=2(b)

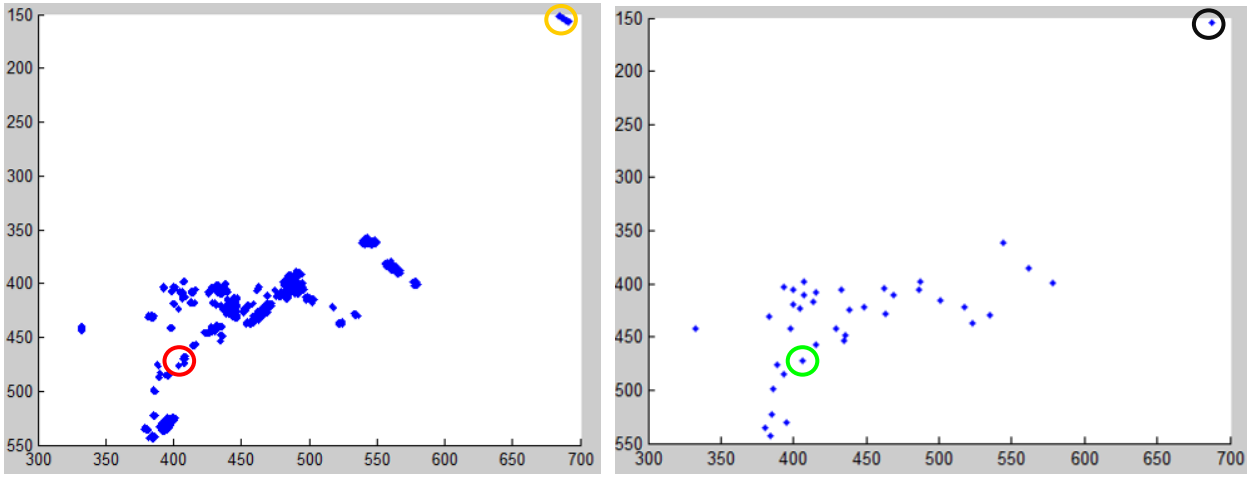
Pada Gambar 9(b) dengan nilai *eps*=2, hasil ekstraksi objek yang dihasilkan juga belum optimal. Hal ini terlihat pada hasil ekstraksi objek dengan nilai *eps*=2 masih terdapat data objek target yang dianggap sebagai *noise*, sehingga data tersebut hilang. Salah satu data hasil ekstrasi objek yang dianggap sebagai *noise* ditunjukkan pada Gambar 9(a) yang ditandai dengan lingkaran merah.

Berbeda dengan nilai *eps*=1 maupun *eps*=2, dengan nilai *eps*=3 data objek target yang kategorikan sebagai *noise* untuk nilai *eps*=1 maupun *eps*=2 (Gambar 10(a) yang ditandai dengan lingkaran kuning), dapat diekstraksi menjadi objek target untuk nilai *eps*=3 (Gambar 10(b) yang ditandai dengan lingkaran hitam). Akan tetapi untuk nilai *eps*=3 terdapat hasil ekstraksi objek yang menghasilkan 2 titik data objek target (Gambar 10(b) yang ditandai dengan lingkaran hijau) yang diekstraksi dari 1 data objek target (Gambar 10(a) yang ditandai dengan lingkaran merah), sehingga nilai *eps*=3 juga belum optimal.



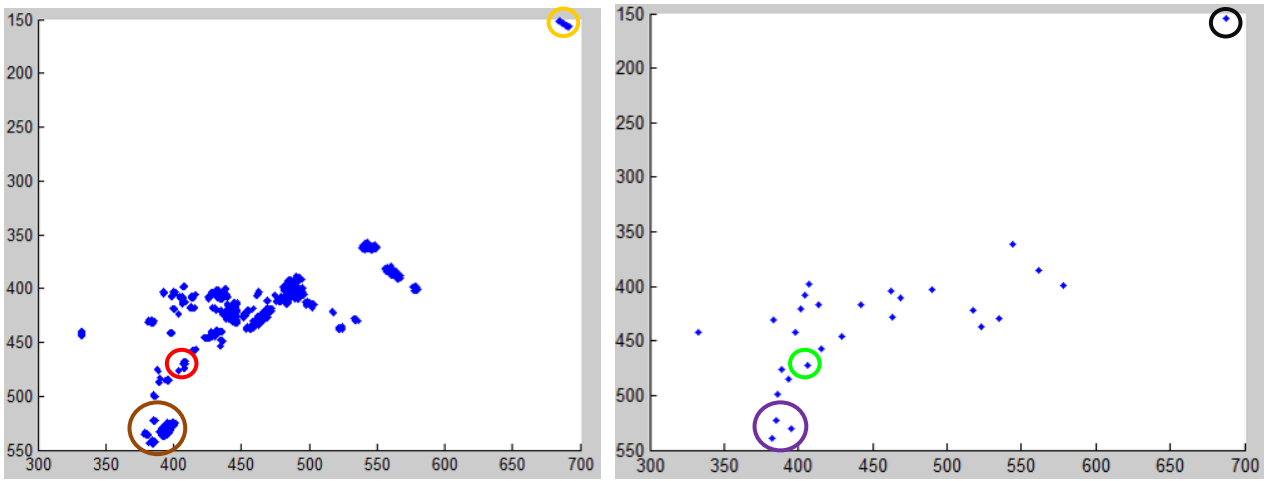
(a) (b)

Gambar 10. Perbandingan data sebelum dilakukan ekstraksi objek(a) dengan hasil ekstraksi objek dengan nilai parameter *eps*=3(b)



(a) (b)

Gambar 11. Perbandingan data sebelum dilakukan ekstraksi objek(a) dengan hasil ekstraksi objek dengan nilai parameter *eps*=4(b)



(a) (b)

Gambar 12. Perbandingan data sebelum dilakukan ekstraksi objek(a) dengan hasil ekstraksi objek dengan nilai parameter *eps*=5(b)

Selanjutnya dengan nilai *eps*=4, data objek target yang kategorikan sebagai *noise* untuk nilai *eps*=1 maupun *eps*=2 (Gambar 11(a) yang ditandai dengan lingkaran kuning), dapat diekstraksi menjadi objek target untuk nilai *eps*=4 (Gambar 11(b) yang ditandai dengan lingkaran hitam). Begitu juga untuk data yang diekstrasi dengan jumlah yang bebeda sebagaimana yang terjadi untuk nilai *eps*=3 (Gambar 11(a) yang ditandai dengan lingkaran merah), untuk nilai *eps*=4 mampu diekstrasi dengan jumlah yang sesuai (Gambar 11(a) yang ditandai dengan lingkaran hijau).

Untuk nilai *eps*=5, hasil ekstraksi objek yang dihasilkan kembali tidak optimal karena terdapat hasil ekstraksi objek yang menghasilkan perbedaaan titik data objek target (Gambar 12(b) yang ditandai dengan lingkaran coklat) apabila dibandingkan dengan data objek target sebelum diekstraksi (Gambar 12(a) yang ditandai dengan lingkaran ungu).

Dari hasil yang didapat dari nilai-nilai *eps* yang telah diujicobakan, diketahui bahwa untuk nilai *eps*=4 tidak menghasilkan objek target yang dikategorikan sebagai *noise*. Selain itu, untuk nilai *eps*=4 juga tidak menghasilkan perbedaan jumlah objek target hasil ekstrasi objek dibandingkan dengan jumlah target sebelum ekstraksi. Oleh karena itu, nilai *eps*=4 merupakan nilai yang optimal untuk karakteristik radar.

Dengan menggunakan parameter *eps*=4 dan *minPts*=4, pada Tabel 1 ditampilkan waktu proses yang dibutuhkan untuk memproses *clustering* dan pada Tabel 2 ditampilkan jumlah data dari setiap *cluster* yang terbentuk dari hasil *clustering*.

Tabel 1. Waktu proses komputasi modul ekstraktor

Ulangan ke	Jumlah data	Jumlah *cluster* yang terbentuk	Waktu Komputasi
1	2627	39	9 s
2	2627	39	9 s
3	2627	39	9 s

Tabel 2. Hasil *Clustering* data

No. *Cluster*	Jumlah data
1	600
2	18
3	10
4	313
5	356
6	264
7	12
8	63
9	47
10	78
11	5
12	25
13	12
14	15
15	123
16	24
17	15
18	12
19	8
20	23
21	39
22	8
23	5
24	16
25	29
26	227
27	10
28	12
29	25
30	4
31	15
32	30
33	30
34	12
35	19
36	17
37	8
38	416
39	199
Jumlah	3144

Dari hasil proses *clustering* yang ditunjukan pada Tabel 2, terlihat jumlah titik data masing-masing *cluster* berbeda. Hal ini dipengaruhi oleh pemusatan titik-titik dari pantulan objek. Semakin banyak titik pantulan yang berdekatan, maka akan semakin juga jumlah banyak juga jumlah anggota *cluster* dari *cluster* yang terbentuk yang mencakup titik tersebut.

Dari hasil proses dengan parameter *minPts* = 4 dan *eps* = 4, setiap titik data pantul tidak ada yang tidak memenuhi kedua parameter tersebut. Sehingga dari proses tersebut tidak didapat titik data pantul yang diidentifikasikan sebagai *noise*.

Selain itu, dari hasil proses juga terlihat terdapat perbedaan jumlah data sebelum proses *clustering* dengan jumlah data setelah *clustering* sebagai mana yang ditunjukan pada Tabel 3.

Tabel 3. Perbandingan jumlah data sebelum proses dan setelah proses ekstraksi objek

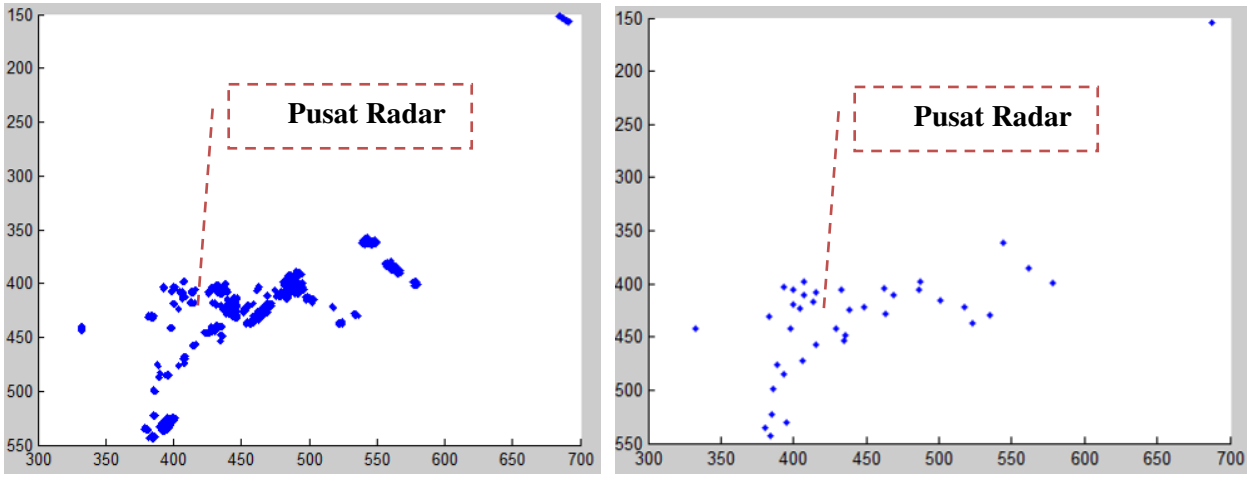
Jumlah data sebelum *clustering*	Jumlah data setelah *clustering*	Selisih
2627	3144	517

Pada Tabel 3 terlihat terdapat penambahan jumlah data setelah dilakukan proses *clustering* dengan selisih sebanyak 517 titik data. Hal ini disebabkan beberapa titik data yang sama masuk dalam beberapa *cluster*, sehingga keseluruhan jumlah data dari seluruh *cluster* mengalami peningkatan dibandingkan dengan jumlah titik data sebelum dilakukan proses *clustering*. Beberapa titik data yang sama masuk dalam beberapa *cluster* disebabkan karena titik-titik data tersebtu terletak pada beberapa *cluster* yang berjarak sangat deket, sehingga titik-titik tersebut menjadi *border point* dari di masing-masing *cluster* yang memilikinya [4]. Meskipun terdapat titik-titik data yang sama yang terdapat dalam lebih dari satu *cluster*, hal ini tidak mempengaruhi hasil akhir dari proses *clustering* karena hasil akhir diambil dari titik tengah dari masing-masing *cluster*.

Setelah dilakukan perhitungan untuk mencari titik tengah pada masing-masing *cluster*, pada Gambar 5(b) ditunjukan titik tengah dari *cluster* yang didapat. Titik tengah dari setiap *cluster* ini lah yang merupakan hasil ekstraksi objek dari titik-titik pantul yang ditangkap oleh radar.

Dari perbandingan data sebelum dilakukan proses ekstraksi objek pada Gambar 5(a) dan data hasil ekstraksi objek pada Gambar 5(b) terlihat bahwa data hasil ekstraksi obejek lebih sederhana, dengan hanya diwakili oleh satu titik dari setiap objek (sejumlah 39 titik objek) dari data input (sejumlah 2627 titik) yang diproses.

Bagian terakhir dari perancangan pengolahan citra ini adalah proses pengiriman hasil akhir ke bagian-bagian yang lain yang memerlukan data hasil ekstraksi objek sebagai data input untuk proses selanjutnya.



(a) (b)

Gambar 5. Data pantul radar sebelum dilakukan proses ekstraksi objek(a) dan Data hasil ekstrasi objek oleh modul ekstraktor(b)

5. Kesimpulan

Pada penelitian ini telah berhasil dirancangan dan diimplementasikan pengolahan citra radar menggunakan DBSCAN untuk ekstraksi objek-objek dari data hasil refleksi objek yang ditangkap radar FM-CW.

Dari hasil ekstraksi objek didapat data yang lebih sederhana, serta data yang memiliki kualitas yang lebih baik karena kemampuan DBSCAN untuk memisahkan *noise* data dari data input secara otomatis.

Ucapan Terima Kasih

Saya ucapkan terima kasih kepada Nova Hadi Lestriandoko dan Octa Heriana yang telah membantu dalam penelitian ini, Bapak Mashury Wahab selaku ketua Tim Radar PPET-LIPI, dan seluruh rekan-rekan Tim Radar LIPI. Tak lupa juga saya ucapkan terima kasih kepada satuan kerja PPET-LIPI & satuan kerja UPT LPSN-LIPI yang telah membantu dalam memfasilitasi terlaksananya tulisan ini.

Daftar Pustaka

[1] A.K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, 1988.

[2] L. Kaufman and P.J. Rousseeuw, *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley & Sons, New Jersey, 1990.

[3] R. Ng and J. Han., "Efficient and Effective Clustering Method for Spatial Data Mining". In *Proceedings of the 20th VLDB Conference*, Chile, pp. 144–155, 1994.

[4] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu., "A Density-Based Algorithm for Discovering Clusters In Large Spatial Databases with Noise". In *Proceedings of the Second Int'l Conference on Knowledge Discovery and Data Mining*, Portland, OR, pp. 226–231, 1996.

[5] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim., "CURE: An Efficient Clustering Algorithm for Large Databases". In *Proc. of 1998 ACM-IGMOD Int. Conf. on Management of Data*, Seattle, 1998, pp. 73–84.

[6] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim., "ROCK: A Robust Clustering Algorithm for Categorical Attributes". In *Proc. of the 15th Int'l Conf. on Data Eng*., Sydney, pp. 512-521, 1999.

[7] D.P. de Oliveira, J.H. Garrett Jr., L. Soibelman, *A Density-Based Spatial Clustering Approach for Defining Local Indicators of Drinking Water Distribution Pipe Breakage*, "Advanced Engineering Informatics", Elsevier, pp. 380–389, 2011.

[8] M. Daszykowski, B. Walczak, D.L. Massart, *Looking for Natural Patterns in Data: Part 1. Density-Based Approach*, "Chemometrics and Intelligent Laboratory Systems", Elsevier, pp. 83–92 , 2001.

[9] L. Zhou, P.K. Hopke, P. Venkatachari, *Cluster Analysis of Single Particle Mass Spectra Measured at Flushing, NY*, "Analytica Chimica Acta", Elsevier, pp. 47–56, 2006.

[10] W. Zhao, P.K. Hopke, K.A. Prather, *Comparison of Two Cluster Analysis Methods Using Single Particle Mass Spectra*, "Atmospheric Environment", Elsevier, pp. 881–892, 2008.

[11] S. Liu, Z.T. Dou, F. Li, Y.L. Huang, "A New Ant Colony Clustering Algorithm Based on DBSCAN". In *Proceedings of the 3rd International Conference on Machine Learning and Cybernetics*, Shanghai, pp. 1491–1496, 2004.

[12] A. Ghosh, A. Halder, M. Kothari, S. Ghosh, *Aggregation Pheromone Density Based Data Clustering*, "Information Sciences", Elsevier, pp. 2816–2831.

[13] C. Plant, S.J. Teipel, A. Oswald, C. Böhm, T. Meindl, J. Mourao-Miranda, A.W. Bokde, H.

Hampel, M. Ewers, *Automated Detection of Brain Atrophy Patterns Based on MRI for The Prediction of Alzheimer's Disease*, "NeuroImage", pp. 162–174, 2010.

[14] M.E. Celebi, Y.A. Aslandogan, P.R. Bergstresser, "Mining Biomedical Images with Density-Based Clustering". In *International Conference on Information Technology: Coding and Computing*, Proceedings, IEEE, pp. 163–168, 2005.

[15] M. Mete, S. Kockara, K. Aydin, *Fast Density-Based Lesion Detection in Dermoscopy Images,* "Computerized Medical Imaging and Graphics", Elsevier, pp. 128–136, 2011.

[16] J. Gong, C.H. Caldas, *Data Processing for Real-Time Construction Site Spatial Modeling*, "Automation in Construction", Elsevier, pp. 526–535, 2008.

[17] Backlund, H., Hedblom, A., Neijman, N.. (2011). *DBSCAN – A Density-Based SpatialClustering of Application with Noise*. Linköpings Universitet – ITN, 2011, http://staffwww.itn.liu.se/~aidvi/courses/06/dm/Seminars2011/DBSCAN(4).pdf (diakses tanggal 20 Mei 2015)

[18] Martin, Robert Cecil, *UML for Java Programmmers*. Prentice Hall, Englewood Cliffs, New Jersey, 2003.