GamRec: A Clustering Method Using Geometrical Background Knowledge for GPR Data Preprocessing

Ruth Janning, Tomáš Horváth, Andre Busche, and Lars Schmidt-Thieme

University of Hildesheim, Information Systems and Machine Learning Lab, Marienburger Platz 22, 31141 Hildesheim, Germany
{janning,horvath,busche,schmidt-thieme}@ismll.uni-hildesheim.de

Abstract. GPR is a nondestructive method to scan the subsurface. On the resulting radargrams, originally interpreted manually in a time consuming process, one can see hyperbolas corresponding to buried objects. For accelerating the interpretation a machine shall be enabled to recognize hyperbolas on radargrams autonomously. One possibility is the combination of clustering with an expectation maximization algorithm. However, there is no suitable clustering algorithm for hyperbola recognition. Hence, we propose a clustering method specialized for this problem. Our approach is a *directed* shape based clustering combined with a sweep line algorithm. In contrast to other approaches our algorithm finds hyperbola shaped clusters and is (1) able to recognize intersecting hyperbolas, (2) noise robust and (3) does not require to know the number of clusters in the beginning but it finds this number. This is an important step towards the goal to fully automatize the buried object detection.

Keywords: Ground penetrating radar (GPR), Object detection, Clustering, Sweep line algorithm, Preprocessing.

1 Introduction

Ground penetrating radar (GPR) is a method to scan the shallow subsurface without destroying the road surface. It can be used to detect buried objects like pipes, cables, ducts and sewers. To get a 2D GPR radargram image (fig. 1 (b), 2 (a)) of a cross-sectional area of the subsurface, electromagnetic waves are transmitted into the ground and the reflected signals are caught by an antenna. The vector of reflections (intensities on the radargram image) measured at one certain point for different answer times is called an A-Scan. The radargram image, the B-Scan, is a sequence of consecutive A-Scans. If the antenna is moved in a line perpendicular to a pipe, then the signals reflected from the pipe have the shape of a hyperbola branch in the radargram image, because (as fig. 1 (a) shows) the more the antenna nears the pipe the shorter is the time in which the corresponding signal is reflected. Originally, GPR radargram images were interpreted manually by human experts, but the pipe localization takes much time in this way (cp. [6]). Hence, an automatization of the radargram interpretation which can accelerate the whole process is required. However, in the beginning it is not known how much hyperbolas there are and where the hyperbola branches are located. Additionally, there might be noise in the radargram image and the different hyperbola branches might intersect (see fig. 2, 3).

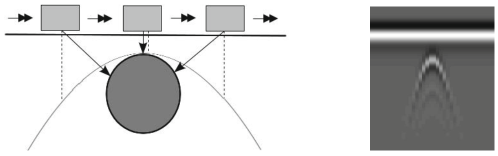

Fig. 1. The antenna (*rectangle*) is moved perpendicular to the buried object (*circle*). The downwards directed arrows represent transmitted signals, the dotted lines indicate the position of the received signal values in the corresponding A-Scans. On the radargram (right) one can see a hyperbola branch corresponding to the buried object.

2 General Problem

For $n \in \mathbb{N}$ let $[n] := \{x \in \mathbb{N} \mid 0 \leq x \leq (n-1)\}$. Given is a dense 2D image

$$\boldsymbol{B} \subseteq [c] \times [r] \times [\iota_{min}, \iota_{max}],\ c, r \in \mathbb{N},\ \iota_{min}, \iota_{max} \in \mathbb{R}\ , \tag{1}$$

a B-Scan composed of c columns (A-scans) with intensities in a range from ι_{min} to ι_{max}. The r rows of $\boldsymbol{B}$ correspond to the signal travel times. In this B-scans upper branches (the origin is the upper left corner of the image) of North-South opening hyperbolas with extreme intensities are searched.

$\boldsymbol{B}^T := \{(j,i) \mid (j,i,\iota) \in \boldsymbol{B} \wedge \iota > \tau\}$ is a sparse representation of $\boldsymbol{B}$ containing only points with extreme intensities larger than a threshold τ.

A searched upper hyperbola branch can be defined as $f_{a,b,j_0,i_0} : \mathbb{R} \rightarrow \mathbb{R},\ i \mapsto \sqrt{(1 + \frac{(j-j_0)^2}{b^2}) \cdot a^2} + i_0$ where (j_0, i_0) is the center of the hyperbola, a is the distance between the center and the apexes of the branches and b influences the curvature of the hyperbola. Let the distance between a point and a hyperbola branch be $D((i,j), f) := d((j,i),(j,f(j)))$ for any distance measure between points, e.g. the Euclidean distance $d((j,i),(j',i')) := \sqrt{(j-j')^2 + (i-i')^2}$. The hyperbola recognition problem can then be formalized as follows:

Definition 1 (Hyperbola recognition problem). *Given* $\boldsymbol{B}^T$ *and* D*, find a set of hyperbola branches* $F := \{f_1, ..., f_K\}$ *and a decomposition* $\boldsymbol{B}^T = (\bigcup_{f_k \in F} h_{f_k}) \cup B_{noise}, \forall f_k \in F : B_{noise} \cap h_{f_k} = \emptyset$*, of* $\boldsymbol{B}^T$ *into clusters* h_{f_k} *and a noise cluster* B_{noise} *with* $|B_{noise}| \approx \nu^{|F|} \cdot |\boldsymbol{B}^T|$ *for some* $\nu \in [0,1)$ *such that the following sum of distances of all points* $\in \boldsymbol{B}^T \setminus B_{noise}$ *to their assigned hyperbola branch is minimal:*

$$err := \sum_{k=1}^{K} \sum_{(j,i) \in h_{f_k}} D((j,i), f_k) \tag{2}$$

3 Related Work

In the last few years different approaches were applied to the hyperbola recognition problem on GPR radargrams. The Hough Transform (HT) (e.g. used in [6], [7]) is an often used method. However, the HT is a brute force method and is hence computationally intensive (see e.g. [2]). A more efficient approach is presented in [2]. This method uses clustering combined with a classification expectation maximization algorithm. In an iterative process the algorithm fits hyperbola branches to clusters and then rearranges the clusters by assigning each point to one cluster according to the maximum posterior probability of the given point to be in this cluster. The hyperbola fitting and a rearrangement of the clusters is done alternately until convergence. However, for getting the initial clusters a K-means algorithm is used. The K-means clustering has some important drawbacks in relation to the hyperbola recognition problem:

1. It is not resistant against noise (even if a noise cluster is used which contains the $q\%$ most distant points of the other clusters, see fig. 2 (b), (d)).
2. It can recognize only convex clusters (see fig. 2 (c)).
3. The number K of clusters to detect has to be known beforehand.

In [2] a Bayesian information criterion (BIC) is proposed to estimate K, but also for this procedure a range for the possible number of hyperbolas has to be estimated beforehand and for each of the values in this range the whole approach has to be applied. Afterwards the model with the highest overall likelihood is selected as the final result.

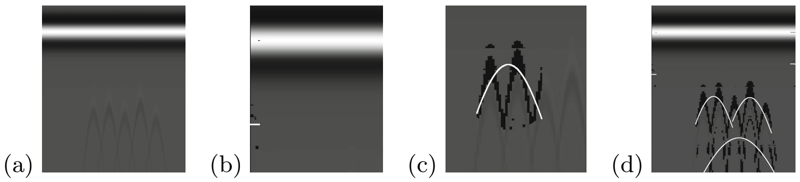

Fig. 2. (a) Radargram image of 5 buried objects. (b) K-means does not recognize the noise points as noise and assigns them to a cluster (*black*, fitted hyperbola: *white*). (c) K-means can only find convex clusters (*black*, fitted hyperbola: *white*). (d) final result of K-means with $K = 5, t = 20$ and $q = 10$ for the noise cluster.

A clustering method which avoids the above mentioned drawbacks of K-means is the shape based clustering method DBSCAN [3]. This method is able to treat noise, to find clusters of arbitrary shapes and the number of occurring clusters has not to be known beforehand. Nevertheless, the original DBSCAN

1. is not able to recognize intersecting hyperbola branches, which are caused by nearby buried objects and often occur in GPR radargrams (it finds one cluster for two or more intersecting hyperbolas (see fig. 3 (b))), and
2. does not differentiate between clusters with hyperbola shape and clusters with other shapes.

4 Our Contribution

The drawbacks of DBSCAN in relation to the hyperbola recognition problem can be treated by integrating geometrical background knowledge. For this purpose we propose *GamRec* (*Geometric approach for multi-hyperbola Recognition*), a kind of a *directed* DBSCAN combined with a sweep line algorithm. Sweep line algorithms (see e.g. [1]) originate from algorithmic geometry. The idea is to imagine that a line moves (*sweeps*) over the considered plane from one side to the other one and pauses if it touches a special point to apply some action to this point. Such points are visited just once, i.e. every point behind the sweep line is never visited again. The characteristic of sweep line algorithms is that they reduce the computational complexity by translating an n dimensional static problem into an $n-1$ dimensional dynamic problem. A clustering algorithm for spatial data which uses a sweep line algorithm is presented in [5], but it is not able to separate overlapping clusters, i.e. it can also not recognize intersecting hyperbola branches. In GPR radargram images apexes are the highest points of hyperbola branches and if an apex is once found one has to search for the left and right hyperbola branch side below it in left-down and right-down direction. Hence, a sweep line can be moved top down over the image and collect hyperbola shaped clusters composed of an apex, a right and a left side. In doing so intersections are treated by considering that a point of an intersection must belong to the right side of one hyperbola branch and to the left side of another one. The usual sorting phase of a sweep line algorithm is not needed, as the considered points are already sorted by their occurrence in the input image. Our approach treats all the above mentioned problems of K-means and DBSCAN in relation to the hyperbola recognition problem, i.e. it (see also fig. 3 (c), (d))

1. ignores background noise,
2. finds hyperbola branch shaped clusters composed of apex, right and left side,
3. has not to know the number of occurring hyperbolas in the beginning,
4. recognizes intersecting hyperbola branches as different hyperbola branches,
5. distinguishes between hyperbola shaped clusters and non-hyperbola clusters.

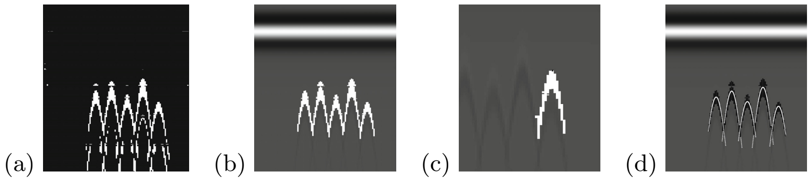

Fig. 3. (a) Thresholded radargram image with $p = 18$. (b) One Single DBSCAN cluster (*white*) found for actually 5 hyperbolas. (c) Single GamRec cluster (*white*). (d) final resulting clusters (*black*) and fitted hyperbola branches (*white*) of GamRec applied to the thresholded radargram image of the radargram in fig. 2 with $e = 8$.

The shapes of the resulting clusters are very close to the shapes of the original hyperbola branches (see fig. 3 (c)). That indicates that a hyperbola fitting has to be applied just once after running GamRec instead of applying it several times in an iterative process to rearranged clusters until the clusters are close enough to the hyperbola branches, as done in the approach of [2]. To strengthen this assumption in our experiments in section 6 we compare the original initialization (K-means with K given) of the approach in [2] with GamRec as an initialization and show that if we use GamRec instead of K-means with K given just one hyperbola fitting step is needed for achieving already very good results. But first we will introduce GamRec in the following section 5.

5 The GamRec Algorithm

Our approach (see also fig. 4) consists of three consecutive steps: the preparation, the sweep step and the noise removing step.

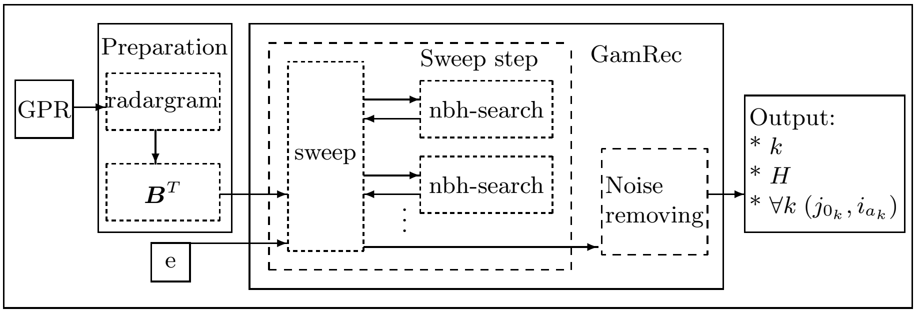

Fig. 4. Architecture of GamRec (nbh-search = *neighborhood search*)

5.1 Preparation

Before applying our algorithm (and any other clustering method) the B-scan $\boldsymbol{B}$ has to be translated into $\boldsymbol{B}^T$. Hence, for every row i of the radargram image an intensity threshold τ_i is computed for $\boldsymbol{B}^T := \{(j,i) \mid (j,i,\iota) \in \boldsymbol{B} \wedge \iota > \tau_i\}$. Threshold τ_i corresponds to the lower limit of the upper $p\%$ quantile of the intensities ι of the points $(j,i,\iota) \in \boldsymbol{B}$ in row i. Parameter p depends on the contrast of the image but can be considered as a constant for the same kind of images (for our experiments we used $p = 18$).

5.2 The Sweep Step

GamRec searches for a set of clusters $H := \{h_1, h_2, ..., h_K \mid h_k \subseteq \boldsymbol{B}^T, k = 1, ..., K\}$ with $h_k := \{(j,i) \in \boldsymbol{B}^T \mid \exists (j',i') \in h_k \neq (j,i) : (|j-j'| \leq e) \wedge (|i-i'| \leq e)\}$ where e is the maximal distance between two neighboring points in a cluster. Such clusters h_k shall possess the following properties:

- $h_k = h_{k_L} \cup h_{k_C} \cup h_{k_R}$ is a union of point sets.
  - h_{k_C} contains the apex (j_a, i_a).
  - $h_{k_L} := \{(j,i) \in h_k \mid j < j_a, i \geq i_a\}$ contains all points of the left side.
  - $h_{k_R} := \{(j,i) \in h_k \mid j > j_a, i \geq i_a\}$ contains all points of the right side.
- For every point $(j,i) \in h_k$ must hold $|(\sqrt{(1 + \frac{(j-j_0)^2}{b^2})} \cdot a^2 + i_0) - i| < \delta$ with appropriate a, b, i_0, j_0 and δ. That means the points in the cluster are positioned around and near to a hyperbola branch.

```
Sweep:
$H := \{\}$;
for $i$ from 1 to $r$ do estimate all middle points in row $i$;
    for every found middle point $poi = (j_{mid}, i)$ do
        if $poi$ is not used right and not used left then do
            $h_k :=$ directedNeighborhoodSearch($poi$);
            if $|h_{k_L}| > e^2 \wedge |h_{k_R}| > e^2$ then do $H := H \cup \{h_k\}$;
                for all points $poi_L \in h_{k_L}$ do mark $poi_L$ as used left;
                for all points $poi_R \in h_{k_R}$ do mark $poi_R$ as used right;
```

Fig. 5. GamRec algorithm: sweep step

Our algorithm (fig. 5, 6) applies a sweep line algorithm by moving top down over the thresholded image $T := \{(j,i,\iota) \mid (j,i,\iota') \in \boldsymbol{B} \wedge ((j,i) \in \boldsymbol{B}^T \Rightarrow \iota = 1) \wedge ((j,i) \notin \boldsymbol{B}^T \Rightarrow \iota = 0)\}$. With the sweep line all *middle points* (see def. 2) are visited.

Definition 2 (Middle point). *Let $s = ((j_1,i),...,(j_n,i))$ with $j_1 < j_2 < ... < j_n$, $|j_{w+1} - j_w| \leq e, w = 1,...,n$ be a sequence of points $\in \boldsymbol{B}^T$ in row i. For every such sequence s with length n in row i a middle point (j_{mid}, i) is the point in s with the index $mid = div(\frac{n}{2})$.*

For every *middle point* a directed neighborhood search is started (fig. 6), if the point was not *used left* or *used right* before (a point is assigned as *used left* (*used right*), if it belongs to the left (right) side of a hyperbola cluster already found). For the points $\in \boldsymbol{B}^T$ of the left side h_{k_L} of the appendant cluster is searched in the neighborhood in left and down direction and for the points $\in \boldsymbol{B}^T$ of the right side h_{k_R} in right and down direction. A neighboring point $\in \boldsymbol{B}^T$ is inserted into h_{k_L}, (1) if the point was not *used left* and not *used right* before or (2) if it was yet *used right* for another cluster and is now located left from the current apex. h_{k_R} is treated in the same way but with *left* and *right* transposed. In this way GamRec is able to treat intersections of different hyperbola branches. It allows that points may belong to more than one cluster if they are located in the intersection of two hyperbola shaped clusters. GamRec uses one hyper parameter e for the maximal distance between neighboring cluster points (equivalent to ϵ in DBSCAN). Parameter e depends on the closeness of the points belonging to a hyperbola cluster. For our experiments we used a constant value of $e = 8$, estimated by a grid search. The second hyper parameter of DBSCAN, the minimal number of neighbors min_N, is not needed in GamRec, as noise is identified mainly by considering the hyperbola shape as described in the following section 5.3.

```
directedNeighborhoodSearch($(j_{mid}, i)$)

1. $h_{k_C} := \{(j_{mid}, i)\}$;
2. $h_{k_L}^{(0)} := \{poi_L = (j_{mid} - e_1, i + e_2) \in \boldsymbol{B}^T \mid e_1 = 1, ..., e, e_2 = 0, ..., e,$
       $poi_L$ is not $used\ left \vee poi_L$ is $used\ right\}$;
   until $h_{k_L}^{(t)} = h_{k_L}^{(t-1)}$ do
       $h_{k_L}^{(t)} := h_{k_L}^{(t-1)} \cup \{poi_L = (j, i) \in \boldsymbol{B}^T \mid \exists (j_v, i_v) \in h_{k_L}^{(t-1)} : j = j_v - e_3,$
       $i = i_v + e_4,\ e_3 = 0, ..., e, e_4 = 0, ..., e,\ poi_L$ is not $used\ left \vee poi_L$ is $used\ right\}$;
   $h_{k_L} := h_{k_L}^{(t)}$;
3. $h_{k_R}^{(0)} := \{poi_R = (j_{mid} + e_1, i + e_2) \in \boldsymbol{B}^T \mid e_1 = 1, ..., e, e_2 = 0, ..., e,$
       $poi_R$ is not $used\ right \vee poi_R$ is $used\ left\}$;
   until $h_{k_R}^{(t)} = h_{k_R}^{(t-1)}$ do
       $h_{k_R}^{(t)} := h_{k_R}^{(t-1)} \cup \{poi_R = (j, i) \in \boldsymbol{B}^T \mid \exists (j_v, i_v) \in h_{k_R}^{(t-1)} : j = j_v + e_3,$
       $i = i_v + e_4,\ e_3 = 0, ..., e, e_4 = 0, ..., e,\ poi_R$ is not $used\ right \vee poi_R$ is $used\ left\}$;
   $h_{k_R} := h_{k_R}^{(t)}$;
4. return $h_{k_L} \cup h_{k_C} \cup h_{k_R}$;
```

Fig. 6. GamRec: directed neighborhood search

5.3 The Noise Removing Step

Finally, clusters which do not correspond to hyperbola branches are removed from H (see fig. 7). GamRec identifies such a *non-hyperbola* cluster by considering geometrical properties of hyperbolas. It investigates if

- the side length of one side of the rectangle around all of its points is less than $2e$ (to eliminate larger background noise),
- in this rectangle the points in the upper left corner, in the upper right corner or in the lower center belong to the cluster and
- the height of one side is less than half of the height of the other side.

In this cases the cluster cannot have the shape of a hyperbola branch.

```
Noise removing:
for all $h_k \in H$ do
  if $(j_{max} - j_{min}) \leq 2e \vee (i_{max} - i_{min}) \leq 2e$ then do $H := H \setminus h_k$;
  else do
    if   $(\{(j_{min} + e_3, i_{min} + e_4) \mid e_3 = 0, ..., e, e_4 = 0, ..., e\} \subset h_{k_L})$
       $\vee$ $(\{(j_{max} - e_3, i_{min} + e_4) \mid e_3 = 0, ..., e, e_4 = 0, ..., e\} \subset h_{k_R})$
       $\vee$ $(\{(j_{mid} \pm e_3, i_{max} - e_4) \mid e_3 = 0, ..., e, e_4 = 0, ..., e\} \subset h_k)$
    then do $H := H \setminus h_k$;[a]
    else do
      if   $((i_{max_L} - i_{min_L}) < 0.5(i_{max_R} - i_{min_R}))$
         $\vee$ $((i_{max_R} - i_{min_R}) < 0.5(i_{max_L} - i_{min_L}))$ then do $H := H \setminus h_k$;
```

[a] j_{min}, i_{min} are the minimal j and i values of all points $\in h_k$, j_{max}, i_{max} the maximal j and i values (correspondingly i_{min_R}, i_{max_R} and i_{min_L}, i_{max_L} for the right and left side) and j_{mid} is the j value of the apex of h_k.

Fig. 7. GamRec algorithm: noise removing step

5.4 Complexity

Let n be the number of considered points for a clustering. In our experiments we compare the original initialization (K-means with K given) used in [2] with GamRec as an initialization. Hence, we consider the complexity of GamRec as well as of the K-means clustering. The complexity $O(n \cdot K \cdot t)$ of K-means clustering results from the t iterations multiplied by the n investigations for every point of its distances to all K cluster centers. The complexity of GamRec corresponds to the sum of the complexities of the sweep and the noise removing step. As GamRec is applied to points of an image, the sorting of points for the sweep can be skipped and an efficient neighborhood search is enabled. The sweep consists of two main steps. The first step is the search for the m *middle points*, which takes $O(n)$ time, as every point in every row is visited. The second step includes a directed neighborhood search with a complexity equal to the size of the corresponding cluster for every found *middle point* times $2e^2$, as for every point $2e^2$ points in the neighborhood are investigated. The complexity of the second step overall is $O((\sum_{i=1}^{m} size_i) \cdot e^2)$, where $size_i$ is the size of the cluster belonging to the ith *middle point.* Hence, the complexity of the sweep is $O(n + (\sum_{i=1}^{m} size_i) \cdot e^2)$. To specify $O((\sum_{i=1}^{m} size_i) \cdot e^2)$ we investigate three cases:

1. If all points would be *middle points* ($m = n$), then $O((\sum_{i=1}^{m} size_i) \cdot e^2) = O(n \cdot e^2)$, because then every point has either no neighbors (two *middle points* of the same row cannot have a distance less or equal than e, see def. 2) and $size_i = 1$ for every i, or a *middle point* possesses neighbors below it which are *middle points* and belong to its cluster but these points are then marked as *used left* or *used right.* That means that from these points no neighborhood search will be started in the following. Additionally, these points will either be visited never again or only once again if it belongs to an intersection.
2. If no point would be a *middle point* ($m = 0$), then $O((\sum_{i=1}^{m} size_i) \cdot e^2) = O(n \cdot e^2)$ because then holds $n = 0$.
3. If $0 < m < n$, then $O((\sum_{i=1}^{m} size_i) \cdot e^2) = O(n \cdot e^2)$, as every point may belong to at most two clusters (just intersection points belong to two clusters) so that the sum of all cluster sizes $size_i$ has to be less than $2n$.

In every case the complexity of the sweep is $O(n + n \cdot e^2) = O(n \cdot e^2)$. The noise removing step needs $O(m + m \cdot e^2 + m)$ time, as for every cluster h_i two constant time operations are applied and e^2 points at four places in the rectangle around h_i are investigated. Because of $m \leq n$ we assign a complexity of $O(n + n \cdot e^2 + n) = O(n \cdot e^2)$ time to the noise removing. Altogether GamRec has a complexity of $O(n \cdot e^2 + n \cdot e^2) = O(n \cdot e^2)$ time. As one can see, the complexities $O(n \cdot K \cdot t)$ and $O(n \cdot e^2)$ of K-means clustering and GamRec are similar efficient. In our experiments e.g. we have chosen $K = 5, t = 20, e = 8$, which results in $O(n \cdot 100)$ and $O(n \cdot 64)$.

6 Experiments

We simulated a cross-sectional area of 2.50 meter length and 0.60 meter depth. The radargrams (see 2 (a)) of this cross-sectional area were produced by GprMax [4], an electromagnetic wave simulator for modeling GPR data based on the Finite-Difference Time-Domain numerical method. GprMax is an often used tool in this field, as real data are rare and expensive to achieve. Additionally, with this data we can easily compare the results of GamRec and of the original initialization of the approach in [2], as we know exactly the ground truth, i.e. where the pipes are located. This means we can compare the positions found with the real positions of the buried objects. In each of 45 scenarios, 5 perfect conductors (e.g. wires) with a radius of 3 centimeters are buried in a 2.50×0.60 meter area of wet sand at different locations with a distance of 25 centimeters between neighboring pipes. We used a 600 MHz antenna frequency for the measurements. In this way we investigated 225 pipes at different horizontal positions and different depths. For the experiments we applied GamRec (with $e = 8$) as well as the original initialization (K-means with K given, $K = 5, t = 20$) of the approach in [2] to the thresholded radargram images (with $p = 18$, see fig. 3 (a)). To the clusters found a hyperbola fitting algorithm was applied (from the K-means clusters the 10% most distant points were removed into the noise cluster). Subsequently, the positions found (fitted apex for K-means, depth of apex of hyperbola cluster and horizontal position of fitted apex for GamRec) were compared to the upper positions of the corresponding buried objects by computing the Euclidean distance between them (see table 1). Table 1 shows that after one run of GamRec the real positions of the buried objects can be estimated already with an error of just a few millimeters and at most about one centimeter. K-means with K given in contrast delivers position estimations with an error from a few centimeters up to more than one meter. These results indicate that GamRec delivers not just the better initialization but also that there is no need for a rearrangement of the clusters found by GamRec. One hyperbola fitting step after the application of GamRec suffices to reach already very good position estimation results.

Table 1. Average distance errors in meter of the 5 pipes in each of the 45 scenarios ordered from minimal to maximal Euclidean distance. The overall average Euclidean distance error of all 225 pipes is presented in the last column.

	1.	**2.**	**3.**	**4.**	**5.**	**average**
GamRec (standard deviation)	0.0016 m (0.0008 m)	0.0018 m (0.0008 m)	0.0027 m (0.0015 m)	0.0066 m (0.0031 m)	0.0092 m (0.0033 m)	0.0044 m (0.0037 m)
K-means with K given (standard deviation)	0.0678 m (0.0512 m)	0.1292 m (0.0497 m)	0.2413 m (0.1185 m)	0.5078 m (0.3052 m)	1.0707 m (0.5791 m)	0.2365 m (0.4718 m)

7 Conclusions

We presented GamRec, a clustering method specialized for an improved GPR data preprocessing. Just one run of GamRec and one hyperbola fitting step applied to the hyperbola shaped clusters found is needed to reach already very good pipe position estimation results. Furthermore, GamRec finds the number of occurring hyperbolas by itself, it is able to recognize intersecting hyperbola branches and it is resistant against background noise. This work is an initial study in the field of pipe detection by GPR. Our final goal is a full automatization of the whole pipe detection process, including pipe position estimation and pipe course detection.

Acknowledgments. This work is co-funded by the European Regional Development Fund project AcoGPR under the grant agreement no. WA3 80122470.

References

1. Boissonnat, J.-D., Yvinec, M.: Algorithmic Geometry. Cambridge University Press (1998)
2. Chen, H., Cohn, A.G.: Probabilistic Robust Hyperbola Mixture Model for Interpreting Ground Penetrating Radar Data. In: Proceedings of the 2010 IEEE World Congress on Computational intelligence (WCCI 2010), Barcelona (2010)
3. Ester, M., Kriegel, H., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD 1996 (1996)
4. Giannopoulos, A.: Modelling ground penetrating radar by GprMax. Construction and Building Materials 19(10), 755–762 (2005)
5. Zalik, K.R., Zalik, B.: A sweep-line algorithm for spatial clustering. In: Advances in Engineering Software, vol. 40, pp. 445–451 (2009)
6. Simi, A., Bracciali, S., Manacorda, G.: Hough transform based automatic pipe detection for array GPR: algorithm development and on-site tests. In: Radar Conference RADAR 2008, Rome, pp. 1–6 (2008)
7. Windsor, C.G., Capineri, L., Falorni, P.: The Estimation of Buried Pipe Diameters by Generalized Hough Transform of Radar Data. In: Progress in Electromagnetics Research Symposium, Hangzhou, China, pp. 345–349 (2005)