SVM-Inspired Dynamic Safe Navigation Using Convex Hull Construction

Ondrej Linda
Department of Computer Science
University of Idaho, Idaho Falls
1776 Science Center Dr., Ste. 306
Idaho Falls, ID 83402, USA
lind0812@vandals.uidaho.edu

Todd Vollmer
Idaho National Laboratory
2525 Freemont Avenue
Idaho Falls, ID 83415, USA
Todd.Vollmer@inl.gov

Milos Manic
Department of Computer Science
University of Idaho, Idaho Falls
1776 Science Center Dr., Ste. 306
Idaho Falls, ID 83402, USA
misko@ieee.org

***Abstract*— The navigation of mobile robots or unmanned autonomous vehicles (UAVs) in an environment full of obstacles has a significant impact on its safety. If the robot maneuvers too close to an obstacle, it increases the probability of an accident. Preventing this is crucial in dynamic environments, where the obstacles, such as other UAVs, are moving. This kind of safe navigation is needed in any autonomous movement application but it is of a vital importance in applications such as automated transportation of nuclear or chemical waste. This paper presents the Maximum Margin Search using a Convex Hull construction (MMS-CH), an algorithm for a fast construction of a maximum margin between sets of obstacles and its maintenance as the input data are dynamically altered. This calculation of the safest path is inspired by the Support Vector Machines (SVM). It utilizes the convex hull construction to preprocess the input data and uses the boundaries of the hulls to search for the optimal margin. The MMS-CH algorithm takes advantage of the elementary geometrical properties of the 2-dimensional Euclidean space resulting in 1) significant reduction of the problem complexity by eliminating irrelevant data; 2) computationally less expensive approach to maximum margin calculation than standard SVM-based techniques; and 3) inexpensive recomputation of the solution suitable for real time dynamic applications.**

***Index Terms*— Autonomous Navigation, Convex Hull, Machine Learning, Support Vector Machines**

I. INTRODUCTION

Successful navigation of mobile robots or UAVs in an environment full of static and dynamic obstacles is the key to the automation of transportation and work in many industrial areas [1], [2]. When moving to the desired destination, it is absolutely necessary for the UAV to avoid any possible accident with the obstacles in the environment. This not only prevents the UAV from damaging itself but also from posing risk to its load, other UAV's or the whole environment. The navigation algorithm should minimize the risk of any collision and try to construct the safest path possible. Generally this kind of a safe navigation is needed in any application such as navigating an autonomous transporter through a factory hall or through an airport environment, but it is of a vital importance in tasks such as automated transportation of nuclear or chemical waste [3].

The task of constructing the safest path through a set of obstacles is similar to the problem of constructing the optimal separation margin, which is successfully solved by the Support Vector Machines (SVMs) [4], [5]. The SVM technique constructs a linear separation between clusters of points in arbitrary number of dimensions, attempting to maximize the distance from the separation hyper-plane to the clusters boundaries. However, the SVM construction can be computationally expensive and the solution is hard to maintain as the input data are dynamically modified.

This paper presents the Maximum Margin Search using a Convex Hull construction (MMS-CH) algorithm for finding the optimal path between a set of obstacles. Because the navigation path is calculated on a 2D map of the environment and each recorded obstacle is represented as a pair of x and y coordinates, the problem can be reduced to this simple case and several advantageous properties of the 2D Euclidean space can be utilized. Under such conditions the presented MMS-CH algorithm is computationally less expensive than standard algorithms for solving the SVM. The optimal separation margin determines the optimal safest path for the UAV navigating through a set of obstacles, thus minimizes the risk of a collision during the movement. The input data are preprocessed by constructing convex hulls around the clusters of input data and then using the boundaries of the hulls to adaptively search for the optimal margin. Furthermore the MMS-CH algorithm maintains the solution and updates it only when and where it is necessary as the input data are dynamically changed.

The rest of the paper is organized as follows. Relevant background is discussed in section II. Section III defines the problem to be solved. Section IV presents the analyses of the MMS-CH algorithm and particular steps of the algorithm are explained in section V. Experimental results are demonstrated in section VI and section VII concludes the paper.

II. BACKGROUND

Although the MMS-CH algorithm is only inspired by the SVM techniques, its overview is provided. From the presented

theory the correctness of the presented approach can be derived. Further, the SVM theory supports the claims that by preprocessing and eliminating of some input data, the solution of the problem will not be modified. Additionally, an overview of the properties and algorithms for construction of convex hulls is given.

A. Support Vector Machines (SVM)

SVM is well known machine learning technique performing well in both classification and regression problems [6]-[8]. Its success is due to the solid mathematical foundations and following the structural risk minimization principle [4]. The constructed separation function maximizes the margin between different classes. In machine learning theory this means maximizing the capability of the classifier to generalize over the input data. The SVM problem and solution could be described as follows.

Set S of N training points in m dimensional space belonging to two disjunctive classes l_i can be described as:

$$S = \left\{ (\vec{x}_i, l_i) \,\middle|\, \vec{x}_i \in R^m, l_i \in \{-1, 1\}, i = 1, 2 \ldots N \right\} \quad (1)$$

There exists an optimal linear separation hyper-plane maximizing the separation margin described by m dimensional weight vector w and a scalar bias b:

$$\vec{w}^T \vec{x} + b = 0 \quad (2)$$

The solution could be calculated by constructing the Langrangian and transforming it into a dual form:

$$\max W(\vec{\alpha}) = -\sum_{i=1}^{N} \alpha_i + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j (\vec{x}_i \vec{x}_j) \quad (3)$$

Subject to:

$$\sum_{i=1}^{N} y_i \alpha_i = 0 \text{ and } 0 \le \alpha_i \le C, i = 1, 2, \ldots N \quad (4)$$

Here $\alpha = (\alpha_1, \ldots, \alpha_N)$ is a vector of non-negative Lagrange multipliers.

Support vectors (SVs) are input points x_i with non-zero coefficient α_i . The final separation hyper-plane depends exclusively on the set of SVs and the remaining data points are irrelevant for the final result.

The α_i coefficients can be obtained for instance by Quadratic Programming (QP) or by Sequential Minimal Optimization (SMO) algorithm [10], [11]. The decision surface is found as:

$$\vec{w} = \sum_{\vec{x}_i \in SV_s} \alpha_i y_i \vec{x}_i \quad (5)$$

$$b = \frac{1}{N_{SV}} \sum_{\vec{x}_i \in SV_s} b_i \quad (6)$$

Here SV_S is the set of SVs of data set S and N_{SV} is the number of SVs.

B. Convex Hull

The convex hull of a set S of N points in m dimensional space can be described as the intersection of all convex sets containing S. The convex hull C of points $p_1, \ldots, p_N$ is described as:

$$C \equiv \left\{ \sum_{j=1}^{N} \lambda_j p_j \,\middle|\, \lambda_j \ge 0, \sum_{j=1}^{N} \lambda_j = 1 \right\} \quad (7)$$

In 2-dimensional space the term convex hull usually refers to the boundary of the smallest enclosing polygon around the given set of points S.

Several algorithms for computing the convex hull exist in 2D space. An elegant solution is the Jarvis march algorithm with the time complexity of $O(nh)$, where h is the number of points in the hull [12]. More sophisticated solution is the Graham Scan algorithm that requires only $O(n \log n)$ time [13]. The same time complexity is needed for a divide and conquer and for the incremental approach [14], [15].

The Akl-Toussaint heuristic can effectively preprocess the input dataset and eliminate points that cannot be part of the convex hull in linear time [11]. First, the points with min and max x and y coordinates are found and a quadrangle connecting these points is created. Then all points located inside this quadrangle cannot be part of the convex hull and are irrelevant for the solution.

III. Problem Description

Computing the navigation through a cluttered environment requires input information about the obstacles in it. For instances, this information can be presented as a static 2D map of the environment, as inputs from the UAV's sensors or as dynamically received data from a central control unit monitoring the environment. Commonly, the planning unit of the UAV is supplied with a mixture of static and dynamic data points constituting the input dataset at the given time.

A typical situation where the UAV knows its position and

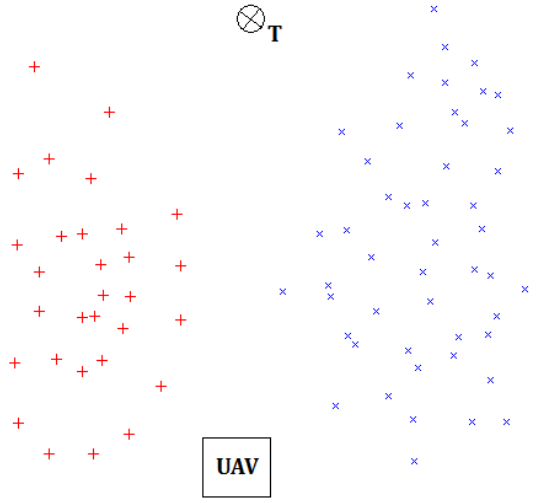

Fig. 1. Map of the environment with marked position of the UAV, its desired target location (T) and obstacles divided into 2 clusters.

the location of the obstacles and tries to navigate to the desired location is shown in Fig. 1. The vector of the desired direction can partition the obstacles into two disjunctive clusters. This partition could be non-trivial and will eventually require decomposition of the problem into multiple sub-problems. The desired trajectory has to be decomposed into multiple segments. This decomposition stage will be necessary for the MMS-CH algorithm to be applicable in any general situation and it is an object of future research.

IV. Analyses of The MMS-CH Algorithm

The MMS-CH algorithm has two main phases. Firstly, the irrelevant input data are eliminated and the complexity of the problem is reduced. Secondly, the sequence of candidate edges is constructed and the optimal margin search is performed by rotating the separation boundary around the convex hull.

A. Irrelevant Data Elimination

Assuming that the obstacles in the environment were partitioned into two linearly separable clusters as denoted in Fig. 1, each data input can be defined as a triplet of coordinates *(x, y)* and label *l* of the cluster it belongs to. The whole data set *S* of *N* inputs can be defined as:

$$S = \left\{ \left(x_i, y_i, l_i\right) \middle| \; x_i, y_i \in R, l_i \in \left\{-1,1\right\}, i = 1,...N \right\} \quad (8)$$

Similarly as in the case of SVMs, the optimal linear separation between the clusters, subject to the width of the margin, has to be calculated. The optimal separation line L_o in 2D is given by:

$$L_o : \; a_o x + b_o y + c_o = 0 \quad (9)$$

L_o must correctly separate all the input data, meaning that for all *i=1,2,...,N*:

$$sign\,(a_o x_i + b_o y_i + c_o) = \begin{cases} 1 & if \quad l_i = 1 \\ -1 & if \quad l_i = -1 \end{cases} \quad (10)$$

For L_o to be optimal it also has to maximize the distance $dist_{SV}$ to the nearest point in the data set *S* from both clusters:

$$dist_{SV} = \min_{i=1,N} \left(\frac{\left| a_o . x_i + b_o . y_i + c_o \right|}{\sqrt{a_o^2 + b_o^2}} \right) \quad (11)$$

Input points at distance $dist_{SV}$ from the separation line are the Support Vectors of the set *S* (SV_S). Based on the convex shape property of the convex hull C_{S_l} of cluster *l*, we can derive the following observation:

$$\frac{\left| a_o . x_i + b_o . y_i + c_o \right|}{\sqrt{a_o^2 + b_o^2}} \begin{cases} > dist_{SV} & if \left(x_i, y_i\right) \notin C_{S_l} \\ \geq dist_{SV} & if \left(x_i, y_i\right) \in C_{S_l} \end{cases} \quad (12)$$

From this observation it is clear that:

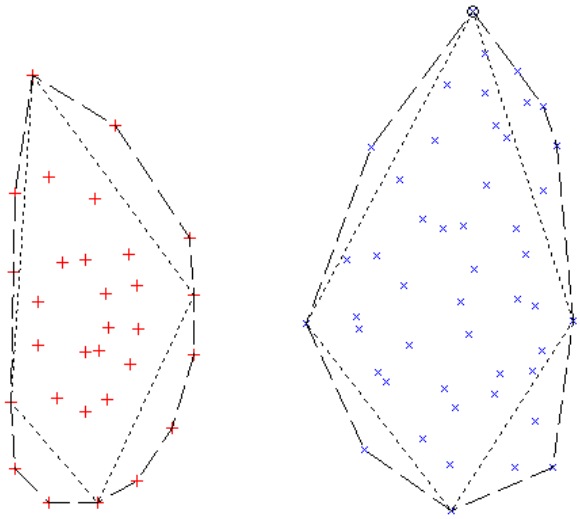

Fig. 2. Clusters of data points with constructed convex hulls (outer dashed line) and the quadrangle of the Akl-Toussaint heuristic (inner dashed line).

$$SV_{S_l} \subset C_{S_l} \quad (13)$$

Hence eliminating the input points located inside the convex hull will not change the final solution. The constructed convex hulls on the input dataset are shown in Fig. 2.

B. Candidate Edges

Knowing the equations of the boundaries of the separation margin is sufficient for calculating the actual separation function L_o:

$$L_o : a_o + b_o + \frac{c_1 + c_2}{2} = 0 \quad (14)$$

Parameters c_1 and c_2 are the c parameters of the linear equations of the boundaries. Therefore, after locating the boundary of the optimal separation margin, the optimal separation function can be constructed.

Having computed one side of the separation boundary, the second one can be easily obtained and consequently the separation function computed. It is possible to reduce the set of edges in the set C_{S_l} to a set of candidate edges E_{S_l}, which could be a potential position of the optimal boundary. The reduction is based on the fact that also the boundaries of the margin are linear separators between the clusters. Therefore each line L_j going through an edge in the set of edges E_{S_l} must fulfill the following condition:

$$\exists\, i, k \;\middle|\; i \neq k,\; sign\left(L_j\left(x_i, y_i\right)\right) \neq sign\left(L_j\left(x_k, y_k\right)\right) \quad (15)$$

In other words a separation line $\underline{L_j}$ going through a particular edge in the set C_{S_l} that locates all the data points into the same half space cannot be a position of the separation line. The input dataset with constructed candidate edges is shown in Fig. 3.

The sets E_{S_l} of both clusters can be ordered in clockwise (CW) or counter-clockwise (CCW) direction forming a

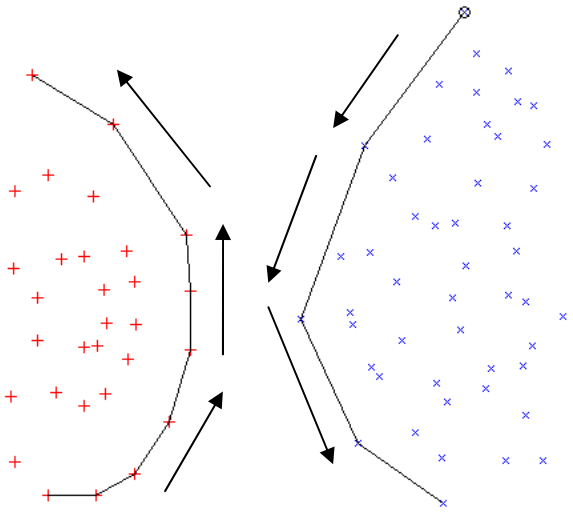

Fig. 3. Sequences of candidate edges.. The arrows show the CCW ordering of the sequences of edges.

sequence of edges. The slopes of edges in these sequences form a monotonically ordered sequence. Based on the slope of one boundary, a pair of neighboring edges in the sequence E_{S_l} of the opposite cluster can be found, with slope interval containing the slope of the first edge. The shared point of these two edges is the position of the second boundary parallel to the first one. The width of the boundary is calculated as the distance between these two parallel lines.

C. *Boundary Rotation*

An operation of rotating line L_j around the set of edges E_{S_l} can be defined as follows:

$$L_{j+1} = Rotate(L_j, \Delta_{angle}, E_{S_l})$$

Initial Step: Set L_0 to the first edge e_0 in the sequence E_{S_l}.

Step 1: $slope(L_{j+1}) = slope(L_j) + \Delta_{angle}$

Step 2: If $slope(L_{j+1}) > slope(e_{k+1})$ then set L_{j+1} to the edge e_{k+1}, where e_{k+1} is the next edge in the ordered sequence of edges E_{S_l}.

Due to the convex shape of the clusters boundaries, the width of the margin is a function of the position on the boundary and has only one global maximum. This maximum is located in the optimal position of the margin. Further, it contains no local maximums as it smoothly increases from the beginning to the optimal position and consequently decreases after the optimum is passed. Hence, during the rotation of the boundary around the sequence of edges the gradient of the margin's width can be used to guide the search towards the optimal position.

V. MMS-CH Algorithm

This section describes the MMS-CH. As explained earlier the UAV is given a mixture of static and dynamic data. Therefore, the algorithm also manages additional insertions and deletions of data from the input dataset.

The algorithm can be divided into the following steps:

Step 1: Find the extreme points and construct the quadrangle of the Akl-Toussaint heuristic in both clusters.

Step 2: Apply the heuristic and eliminate data points irrelevant for the construction of the convex hulls.

Step 3: Construct the convex hulls C_{S_l} from the reduced datasets.

Step 4: Construct the ordered sequence of candidate edges E_{S_l} from both convex hulls.

Step 5: Search through the sequence of edges E_{S_l} and find the edge with the widest margin.

Step 6: Rotate the boundary around the located position as long as the width of the margin increases.

Step 7: If the difference in the width of the margin between consecutive rotations falls under the specified threshold terminate the rotation and go to step 8. Otherwise reduce Δ_{angle} and go to step 6.

Step 8: If a new point is inserted, go to step 2 and apply the heuristic for the given cluster. If the point is located inside the quadrangle of the heuristic, nothing has to be recomputed. If it is located outside, recompute steps 1 through 4 for the given cluster. Test if the new point lies inside the previous margin. If it doesn't, the margin does not have to be recomputed. If it passes the test, steps 5 through 7 are repeated.

Step 9: If a point is deleted from the input set S go to step 2 and apply the heuristic for the given cluster. If the point was located inside the quadrangle of the heuristic, nothing has to be recomputed. If it is located outside, recompute steps 1 through 4 for the given cluster. Test if the point being deleted was located on the boundary of the previous margin. If it was not, the margin does not have to be recomputed. If it was, recompute steps 5 through 7.

Step 5 is similar to the operation of rotation around the sequence of candidate edges E_{S_l}. It quickly identifies the position of the boundary and the rotation can be performed only around this position. This is a significant speed-up compared to the rotation of the boundary around the whole sequence E_{S_l} from its start. The steps of the algorithm are shown in Fig. 4. The final result is shown in Fig. 5.

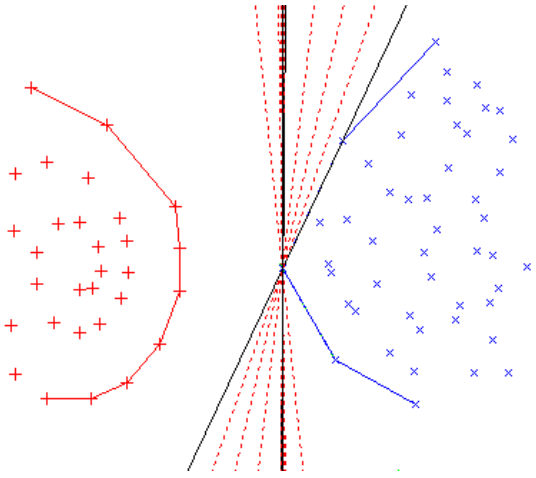

Fig. 4. Particular steps of the adaptive rotation search for the position of the optimal boundary.

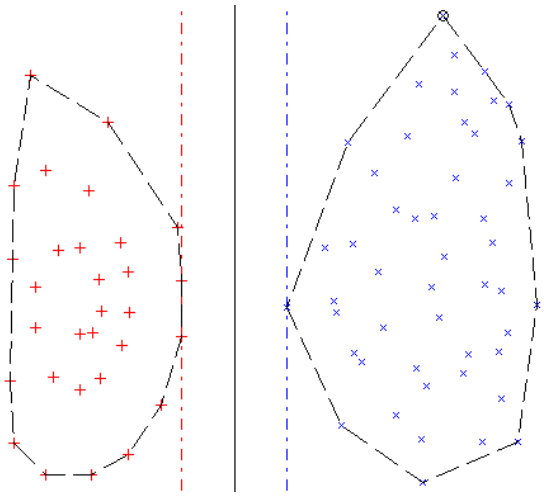

Fig. 5. The margin's boundaries (dashed-and-dotted) and the separation function (solid) computed by the MMS-CH algorithm.

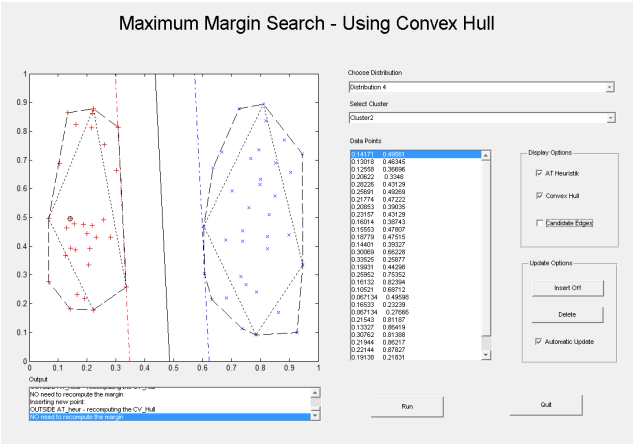

Fig. 6. The GUI of the MMS-CH application implemented in the MATLAB environment.

VI. Experimental Results

The MMS-CH algorithm was implemented in the MATLAB environment. Due to the difficulty of obtaining real world data, manually created distribution of input points was used. Further, the input points were beforehand partitioned into two disjunctive clusters. The graphical user interface (GUI) of the application is shown in Fig. 6.

The GUI displays the input dataset. The MMS-CH algorithm can be applied to the data and the computed result visualized. Along with the results, the convex hull, the sequence of candidate edges and the quadrangle of the heuristic can be displayed to illustrate the preprocessing of the input data.

In order to simulate the dynamic scenario, data points can be manually inserted into or deleted from the input dataset. The MMS-CH algorithm updates the solution online. After each update the new solution is visualized and the GUI outputs description of the performed steps. This output log confirms that the solution is not recomputed when it is not necessary to do so.

To demonstrate the process of dynamic update of the solution, simple case is considered in Fig. 7. There is a moving object inside one of the clusters of obstacles moving in a given direction. The sequence in Fig. 7 demonstrates when and what part of the solution has to be re-calculated. Only when the updated position of the moving object interferes with the boundaries of the previously computed margin, the separation has to be re-computed to reflect the new obstacle in the way.

Testing on various input data distributions and scenarios of insertion and deletion of input points showed the suitability of the MMS-CH algorithm for the real-time dynamic input data. The solution is computed and updated in a fast manner and the elimination of irrelevant input data helps prevent the algorithm from performing unnecessary computations.

VII. Conclusion and Future Work

In this paper the MMS-CH algorithm for calculating the safest path in dynamic environment was presented. The algorithm utilizes the construction of convex hulls over the input data to eliminate data points irrelevant for the solution and to use the boundary of the hulls to search for the optimal separation margin between sets of obstacles. The testing of the algorithm showed that it performs well in dynamic scenarios where the input data might be altered by insertion or deletion of data points. The preprocessing phase of the MMS-CH algorithm can recognize whether the change in the data set does or does not require any recalculation of the previous solution and thus prevents unnecessary computations.

In order for the MMS-CH algorithm to be applicable in any general situation, the desired path will have to be decomposed into multiple sub-sections. This is necessary due to the linear property of the calculated solution, which might not always be a global optimum or might not even exist. Furthermore the combination of the MMS-CH algorithm with the kinetic convex hulls should be investigated [16], [17]. In the case of kinetic convex hull all points in the data set have their own direction vector and specified speed. Maintaining such a structure enables the calculation of events that change the shape of the convex hull and consequently lead to the change of the optimal separation margin. This extension to the MMS-CH algorithm would bring the possibility of computing the solution ahead in time, enabling a prediction of the position and the width of the optimal margin and thus improving the ability of the MMS-CH algorithm to minimize the risk of navigating the UAV into a dangerous situation eventually leading to a collision in the near future.

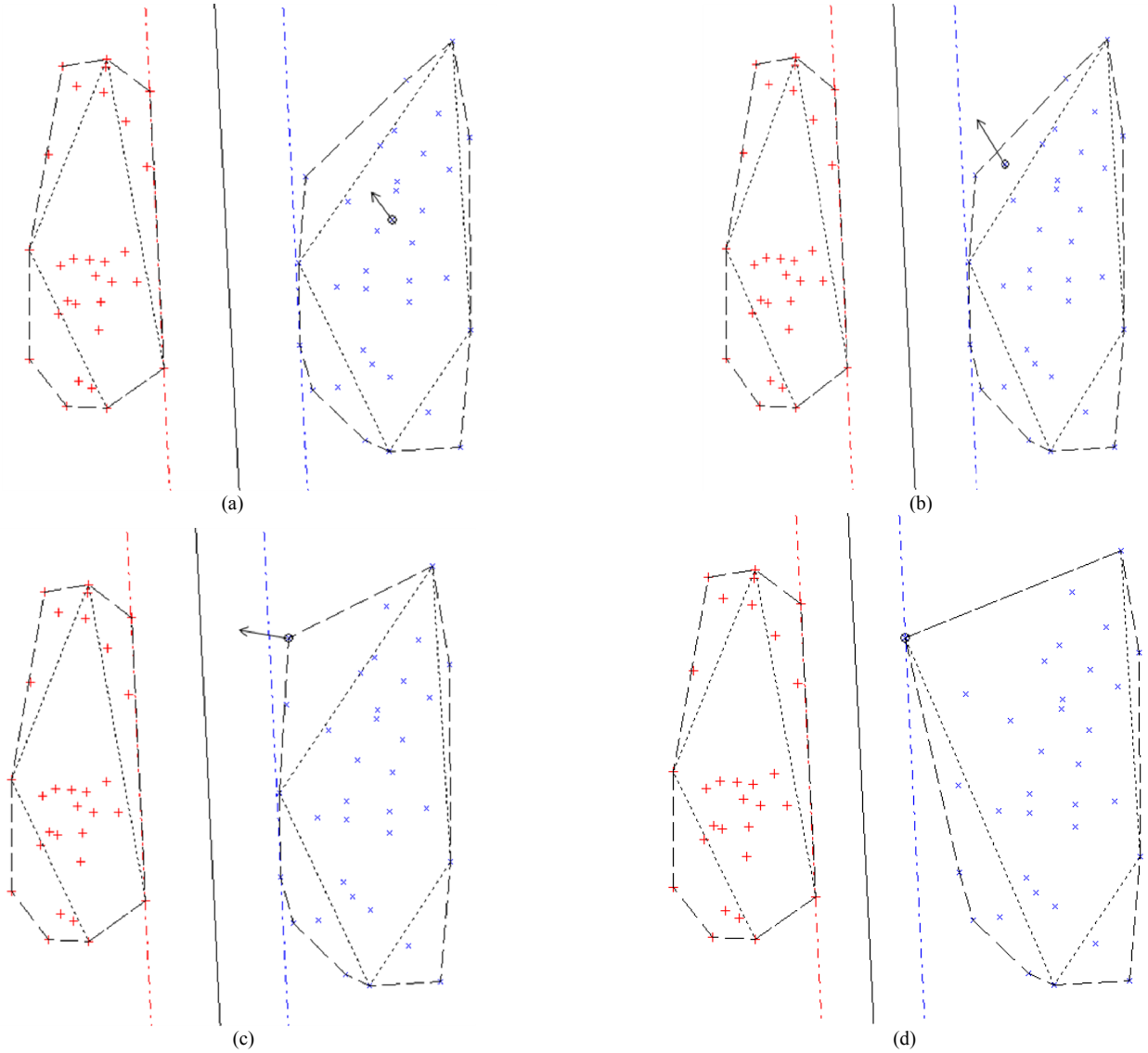

Fig. 7. Example of the dynamic solution calculated by the MMS-CH algorithm. Object located inside the cluster is moving in the given direction (a). When the object moves out from the quadrangle of the heuristic the convex hull has to be recomputed (b). When the object moves out from the convex hull of the cluster, the convex hull is re-computed and the separation margin is checked (c). When the object moves inside the previously computed separation margin, the optimal margin has to be re-computed (d).

ACKNOWLEDGMENT

The authors would like to thank the Idaho National Laboratory and University of Idaho Nuclear Engineering Program for providing support for this project.

REFERENCES

[1] C. Urmson, W. Whittaker, "Self-Driving Cars and the Urban Challenge", IEEE Intelligent Systems, March-April 2008, pp. 66-68.

[2] M. Parent, "Advanced Urban Transport: Automation is on the Way", IEEE Intelligent Systems, March-April 2008, pp. 9-11.

[3] D. Geske, "The Future for Autonomous Haul Trucks", Sept. 2004, http://findarticles.com/p/articles/mi_m0FZX/is_9_70/ai_n7069121.

[4] V. N. Vapnik, *Statistical Learning Theory*. John Wiley and Sons, 1998.

[5] C. J. Burges, "A tutorial on support vector machines for pattern recognition", *Data Mining and Knowledge Discovery*, vol. 2, no. 2, 1998.

[6] T. Joachims, "Text categorization with support vector machines", *Proc. 10th European Conference on Machine Learning*, Chemnitz, Germany, 1998.

[7] H. Yu, J. Han, and K. C. Chang, "PEBL: Positive-example based learning for Web page classification using SVM", *Proc. 8th Int. Conf. Knowledge Discovery and Data Mining*, Edmonton, Canada, 2002.

[8] A. Smola, B. Sch, "A tutorial on support vector regression", Technical report, 1998.

[9] L. Kaufman, "Solving the quadratic programming problem arising in support vector classification", In B. Scholkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods-Support Vector Learning,* pp. 147-168. MIT Press, 1998.

[10] J. C. Platt, "Fast training of SVMs using sequential minimal optimization" In B. Scholkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods-Support Vector Learning,* pp. 185-208. MIT Press, 1998.

[11] S. G. Akl, G. Toussaint, "Efficient Convex Hull Algorithms for Pattern Recognition Applications" *Proc. 4th Intl. Joint Conference on Pattern Recognition*, Kyoto, Japan, pp. 483-487, 1978.

[12] R. A. Jarvis, "On the Identification of the Convex Hull of a Finite Set of Points in the Plane", *Info. Proc. Letters 2,* pp. 18-21, 1973.

[13] R. Graham, "An Efficient Algorithm for Determining the Convex Hull of a Finite Point Set", *Info. Proc. Letters 1*, pp. 132-133, 1972.

[14] F. Preparata, S. J. Hong, "Convex Hulls of Finite Sets of Points in Two and Three Dimensions", *Comm. ACM 20*, pp. 87-93, 1977.

[15] M. Kallay, " The Complexity of Incremental Convex Hull Algorithms in $R^{d>}$", *Info. Proc. Letters 19*, pp. 197, 1984.

[16] J. Basch, L. J. Guibas, J. Hershberger, "Data structure for mobile data", *J. Algorithms 31*, pp. 1-28, 1999.

[17] P. K. Agarwal, L. Arge, J. Erickson, H. Yu, "Efficient tradeoff schemes in data structures for querying moving objects", *Proc 12th European Symposium on Algorithms*, pp. 4-15, 2004.