Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

A pattern recognition approach to threat stabilization

M. K. Allouche
DRDC Valcartier

Defence R&D Canada – Valcartier
Technical Report
DRDC Valcartier TR 2002-239
June 2006

Canada

A pattern recognition approach to threat stabilization

M. K. Allouche
DRDC Valcartier

Defence R&D Canada - Valcartier
Technical Report
DRDC Valcartier TR 2002-239
June 2006

Author

Mohamad K. ALLOUCHE

Approved by

Éloi Bossé
Head/A Section

Approved for release by

Gilles Bérubé
Chief Scientist

Abstract

Threat evaluation is of great importance in the command and control decision making process. Due to highly dynamic changes in the environment, it becomes more and more difficult to assign a stable threat value to an entity. This work aims at stabilizing the threat value assigned to a moving entity having a maneuvering behaviour. To this end, a special type of neural networks, the *Self Organizing Maps*, is used in order to extract some important features from the kinematics of the moving entity. The *direction of advance* of such an entity is computed based on the extracted features. The stabilization of such a direction of advance allows the stabilization of the threat value assigned to such an entity, especially when this value is based on the *closest point of approach* concept. The proposed approach is tested over a simulated data set and compared with other stabilization approaches such as the *Least-Squares Regression Line* method.

Résumé

L'*évaluation de la menace* est essentielle à la prise de décision au sein du commandement et contrôle. En raison des changements rapides qui surviennent dans l'environnement, il est de plus en plus difficile d'associer une valeur de menace stable à une entité donnée. Ce travail se propose de stabiliser une valeur de menace associée à une entité en mouvement non uniforme (comportant des manoeuvres). Pour ce faire, un type spécial de réseaux de neurones, les *Cartes de Kohonen*, est employé pour extraire certaines caractéristiques importantes de la trajectoire de l'entité en mouvement. La *direction d'avance* de cette entité est déterminée à partir des caractéristiques extraites par le réseau de neurones. La stabilisation de la direction d'avance permet la stabilisation de la valeur de menace attribuée à cette entité, spécialement quand cette valeur est basée sur le concept du *point le plus près sur la ligne d'approche*. L'approche proposée est testée sur un jeu de données simulé et est comparée à d'autres approches de stabilisation telles que la méthode des *moindres carrés*.

This page intentionally left blank.

Executive summary

In the *situation analysis process, threat evaluation* is one of the most important activity. A situation is constantly evolving in time and changes occur in the environment at any time. Each change performed by a presumably hostile entity in the environment could be considered as a potential threat. In order to characterize such a threat, a threat value is assigned to each hostile entity. Such a value gives an idea about the degree of threat that could be posed by such an entity, and has a direct impact on the decision-making process relating this entity. In a highly dynamic situations, it is too difficult to determine a stable threat value due to events and changes occurring very rapidly in the environment. As elements of the environment, threat values will be constantly changing in time, making hence the decision making an onerous task. Stabilization techniques for these threat values are of great importance. This work tackles the stabilization of threat values assigned to moving entities by smoothing their kinematics.

A new approach of stabilization is developed in order to smooth the reported trajectory of a moving entity with a maneuvering behaviour. Such an approach allows the stabilization of the threat values that are based on the *Closest Point of Approach Concept* and on a particular class of neural networks, the so-called *Self Organizing Maps* (SOM). Training the SOM over the reported positions of the moving entity aims at providing the main features of its trajectory. These features are used in order to compute the *direction of advance* of the moving entity. The stabilization of the direction of advance allows the stabilization of the threat value assigned to this entity.

The use of the SOM network is very novative, since the training is real-time. Usually, SOMs are used with static data and the training is done off-line. The SOM is tested over two sets of simulated data that have two main characteristics:

- *irregular oscillations*. The entity makes sharp turns and its trajectory is composed of irregular oscillations. Mathematical approaches use to model the trajectory with an analytical equation. Hence, irregular oscillations usually complicate finding such an equation. The SOM approach is welladapted to handle irregularities and sharp turns.

- *uncertain data*. The reported positions are subject to uncertainty due to the limited precision of the sensors such as radars. This uncertainty factor has an effect on the reported positions and hence on the reported trajectory of the moving entity. With mathematical approaches, determining an equation of the trajectory becomes hence a harder task. Although the uncertainty factor, the SOM is still able to handle the stabilization of the direction of advance. The impact of uncertainty on the SOM was measured and compared with other approaches.

In some situations entities are able to make three-dimensional maneuvers. It should be interesting to determine its direction of advance based on three-dimensional reported positions. This could be done by using a three-dimensional SOM. Furthermore, the extracted features by the SOM could be used in order to characterize the maneuvers of

the moving entity. These characteristics are useful in some applications such as the *recognition of activities* or the *recognition of intentions* of the enemy. For example, by characterizing its maneuvers, it would be possible to recognize an attacking behaviour of a moving entity.

Mohamad K. ALLOUCHE. 2006. A pattern recognition approach to threat stabilization. DRDC Valcartier TR 2002-239. DRDC Valcartier.

Sommaire

L'*évaluation de la menace* est cruciale dans le processus de l'*analyse de la situation*. Une situation évolue constamment et des changements apparaissent dans l'environment à tout moment. Chaque changement produit par une entité présumément hostile peut être considéré comme une menace potentielle. Afin de caractériser cette menace, une valeur de menace est attribuée à chaque entité hostile. Cette valeur donne une idée du degré de menace que peut poser une telle entité et conditionne directement sur la prise de décision relative à une telle menace. Dans des situations dynamiques, il est très difficile de déterminer une valeur de menace stable à cause des changements et des événements qui surviennent très fréquemment dans l'environnement. Comme les éléments dans l'environnement, les valeurs de menace vont continuellement changer dans le temps, faisant ainsi de la prise de décision une tâche difficile. Des techniques de stabilisation de ces valeurs de menace deviennent alors nécessaires et d'une importance capitale. Ce travail se propose de stabiliser la valeur de menace associée à une entité en mouvement, en lissant sa trajectoire.

On a mis au point une nouvelle approche de stabilisation pour lisser la trajectoire d'une entité en mouvement pendant qu'elle effectue des maneuvres. Cette approche permet la stabilisation de la valeur de menace qui serait attribuée à cette entité en se basant sur le concept du *point le plus près sur la ligne d'approche*. L'approche proposée se fonde sur un type particulier de réseaux de neurones, appelé *les cartes de Kohonen*. L'apprentissage d'une carte de Kohonen par les différentes positions rapportées de l'entité en mouvement permet d'obtenir certaines caractéristiques importantes de sa trajectoire. Ces caractéristiques sont utilisées afin de calculer la *direction d'avance* de l'entité en mouvement. La stabilisation de cette direction d'avance permet la stabilisation de la valeur de menace attribuée à cette entité.

L'utilisation des cartes de Kohonen est innovatrice car l'apprentissage se fait en temps réel. En général, la carte de Kohonen est plutôt utilisée sur des données statiques et l'apprentissage se fait hors ligne. Le processus de stabilisation par une carte de Kohonen est testé sur deux jeux de données simulés ayant chacun deux caractéristiques principales :

- *Oscillations irrégulières*. L'entité effectue des virages rapides et sa trajectoire est composée d'oscillations irrégulières. Les approches mathématiques essaient de modéliser la trajectoire par une équation analytique. De ce fait, les oscillations irrégulières augmentent la difficulté de trouver une telle expression. L'approche basée sur une carte de Kohonen est bien adaptée à de telles irrégularités dans la trajectoire;

- *Facteur d'incertitude*. Les positions rapportées sont sujettes à un facteur d'incertitude dû à la précision limitée des capteurs utilisés, tels que les radars. Le facteur d'incertitude a un effet sur les positions rapportées de l'entité en mouvement. Avec une approche mathématique, déterminer une équation analytique de la trajectoire rapportée devient une tâche plus difficile. Malgré le facteur

d’incertitude, la carte de Kohonen est capable de stabiliser la direction d’avance. L’impact de l’incertitude a été mesuré et comparé à d’autres approches.

Dans certaines situations, les entités sont capables d’effectuer des manoeuvres en trois dimensions. Il serait alors intéressant de déterminer la direction d’avance en utilisant des données tridimensionnelles. Cela est possible en utilisant une carte de Kohonen tridimensionnelle. De plus, les caractéristiques déterminées par l’apprentissage du réseau peuvent être utilisées dans des applications telles que la *reconnaissance d’activités* ou la *reconnaissance des intentions* de l’ennemi. Ainsi, en caractérisant certaines manœuvres, il serait possible de reconnaître l’intention d’attaquer chez une entité en mouvement.

Mohamad K. ALLOUCHE. 2006. Une approche par reconnaissance de formes à la stabilisation de la menace. DRDC Valcartier TR 2002-239. R et D pour la défense Canada - Valcartier.

Table of contents

List of figures

List of tables

This page intentionally left blank.

1. Introduction

Threat analysis is an important part of the situation analysis process, and has a direct impact on the decision making in the command and control process. One of its major characteristics is the criticality of the real-time aspects. High threat situations require a rapid response from the decision maker in order to eliminate the threat. There is no standard definition of the threat concept since it depends on the situation in which it may occur. For example, one can talk about threats caused by natural disasters, terrorism or any type of warfare. In this work, the naval domain is considered. In war time, warships are constantly subject to various threats coming from torpedoes (anti-submarine warfare), missiles launched by other warships (anti-surface warfare) or/and by an enemy aircraft (anti-air warfare).

In a situation where several threats may occur, it is important to *prioritize* or to *classify* these threats according to different *degrees of criticity*. Such a degree indicates in which order threats should be eliminated. For example, threats could be ordered according to their occurrence time, that is, a threat that is expected to occur first, should be eliminated first. Threats could also be ordered according to their importance, that is, threats with critical impacts should be eliminated first. Hence, the *classification* of threats is strongly related to the real-time aspects of the threat analysis process, since it is always a question of which threat should be eliminated first, and when.

In this work, a specific problem of threat analysis is tackled using a neural approach based on the Kohonen's *Self Organizing Maps* (SOM). It is the *threat stabilization* problem. A launched missile or an attacking aircraft will constantly maneuver while approaching its target. This maneuvering behaviour of the threatening entity makes difficult the prediction of its destination and the identification of the targeted entity. For this reason, it is difficult to prioritize/classify threats caused by maneuvering entities and a process of threat stabilization becomes of great importance. One way to solve this problem is to smooth the reported trajectory of the maneuvering entity. This is the core of this work.

The document is organized as the following. Chapter 2 is an attempt to give a definition of the threat concept. Chapter 2.1 briefly describes the threat analysis process. More specifically, the *Closest Point of Approach* (CPA) concept and the threat stabilization problem are described in Chapters 2.2 and 2.3. Chapter 3 shows the use of neural networks to tackle the threat stabilization problem. It describes a neural network in general, and more specifically, the SOM network. Its application to the threat stabilization problem is addressed in Chapter 5 and illustrated through different sets of simulated data. Finally, coinclusions are given in Chapter 6 pointing to future work.

2. Threat concept

Any attempt to define the concept of threat is strongly dependent on the context or the situation in which it may occur. For example, in the *American Telecom Glossary 2000*, threat is defined as follows: "1. *Capabilities, intentions, and attack methods of adversaries to exploit, or any circumstance or event with the potential to cause harm to, information or an information system.* 2. *Any circumstance or event with the potential to harm an information system (IS) through unauthorized access, destruction, disclosure, modification of data, and/or denial of service*." (INFOSEC-99: http://www.its.bldrdoc.gov/projects/t1glossary2000/t1g2k.html).

This definition could be adapted to fit any other context than that of the information warfare. In the naval warfare context, the definition of threat could become: "*Capabilities, intentions, and attack methods of adversaries to exploit, or any circumstance or event with the potential to cause harm to, our or friend's forces (including warships, aircrafts, soldiers, ...). Different degrees of threat could be considered. If a significant risk of a substantial harm could not be eliminated or reduced by reasonable accommodations, then the threat is direct, otherwise it is indirect*".

From a purely analytical point of view, the threat concept relates to two major players. The first is active and the second is passive. The active player is performing actions and is considered as the origin of the threat. The second suffers the impact of the performed actions. The term *impact* was introduced in the revised version of the Joint Directors of Laboratories (JDL) Model [1]. Threat can be represented as a relation between two entities, which is characterized by a number of attributes such as the duration (the time left before the threat occurs) and other quantitative measures of the situation such as the distance between the two entities, the motion, speed and direction of each of them.

2.1 Threat analysis

Threat analysis is a process that quantitatively qualifies a situation as much as possible. This assumes that some of the situation elements are already analyzed through the *situation analysis process* [2]. In the JDL Model (both original and revised versions) [1], this process belongs to the third level. According to Waltz and Linas [3, p. 28–33] threat analysis is a process where different elements are quantified and two main functions must be fulfilled:

- *Capability estimation function* (including a force counting function) which tries to estimate the enemy's and friend's capabilities (strength, composition, disposition, location and development).

- *Intent estimation function* (including event/activity templating, prediction/trend estimation function, platform performance models) which tries to identify patterns of movements, operational readiness elements, etc.

It is important to notice that these two functions cannot be considered independently. Even if a presumed hostile entity possesses great capabilities, it is important that this entity has in addition the intention to make harm to our (or friend's) forces. With an unknown intention one can only estimate the potential threat of that entity based on its capabilities. In fact, an entity performing actions may unintentionally cause harm and be a high threat. This threat is only a side effect of the performed actions. For instance, an unidentified aircraft is considered as a threat no sooner it has entered a defended zone, regardless its intention and the purpose of its entrance in this zone.

Conversely, no matter how firm is the intention of a hostile entity to make harm to our (or friend's) forces, the underlied threat remains insignificant until this entity has the required capabilities to undertake hostile operations. For example, a baby snake cannot be considered as a threat even if it bites (has the intention to make harm) as an adult one. Still the fact remains that the lack of poison is an obstacle preventing this baby snake from being dangerous.

In the next section, a special concept is analyzed, the closest-point-of-approach (CPA) concept. It is largely used in the classification of direct threats under strong temporal constraints, that is, threats with significant impact that need to be eliminated as soon as possible. The comprehension of this concept will justify the importance of the threat stabilization process described in Section 2.3. The CPA concept is more based on kinematics information about moving entities than on the analysis of its intention. In our context, we suppose that any moving entity that is heading toward a friend ship has bad intentions. An elaborated analysis of the behaviour of the moving entity would be of interest to recognize its intention but would out of the scope of this document.

2.2 The closest-point-of-approach concept

For each potentially agressive moving entity such as a missile for example, it should be possible to measure its degree of threat within the current mission or objectives [4]. This measure could be used in order to classify and/or prioritize threats, but also to determine a response time. For example, if a missile is launched, it is important to determine the time it will take before hitting its target in order to determine the available time remaining to make an appropriate response.

Many threat evaluation techniques are based on the *closest point of approach* (*CPA*). This presupposes that the missile has been already identified as a threat and has bad intentions. Having a *point of interest* (*POI*) (the ship for instance) and given the velocity vector of an approaching missile, the *CPA* is the orthogonal projection of the *POI* on the extension of the velocity vector, as shown in Figure 1. The distance between the *POI* and *CPA* can be considered as a measure of threat. The smaller this distance is, the higher the degree of threat of the missile is.

By using the *CPA*, it is possible to estimate the time *TCPA* (*time to CPA*) that the missile will take to reach the *CPA* (assuming that its speed S will remain constant). It

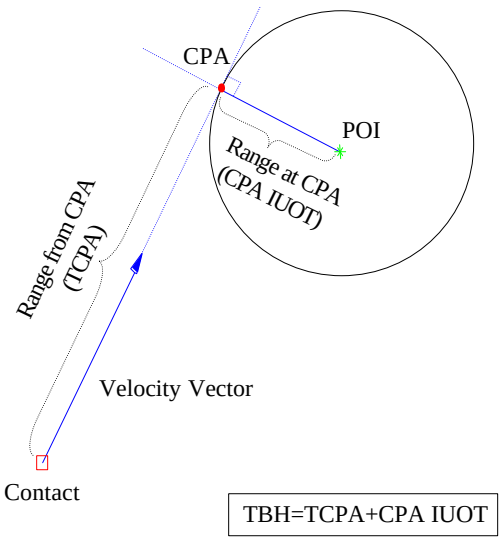


Figure 1: *Closest point of approach (CPA), Time to CPA and Time Before Hit (TBH) concepts*

corresponds to the distance between the current position of the missile and the *CPA* divided by the speed of the missile:

$$TCPA = \frac{distance(Missile, CPA)}{S} \tag{1}$$

From the *CPA*, it is also possible to compute the time for the missile to hit the *POI* (assuming that at the *CPA*, the missile will turn $90°$ toward the *POI* at a constant speed S). This is referred to as the *CPA in units of time* (*CPA IUOT*). It is the distance between the *CPA* and the *POI* divided by the current speed of the missile:

$$CPA\ IUOT = \frac{distance(POI, CPA)}{S} \tag{2}$$

From the *TCPA* and the *CPA IUOT*, it is possible to estimate the total time for the missile to hit the *POI*. This is referred to as the *Time Before Hit* (*TBH*):

$$TBH = TCPA + CPA\ IUOT$$

The *TCPA* and *CPA IUOT* can be used to generate a countermeasure firing solution. For example, a counter-missile missile could be launched in order to destroy the missile (see Figure 2). The *TCPA* and *CPA IUOT* can be used to predict an interception point. Knowing the speeds of the missile and that of the counter-missile missile, the interception point should be reached as soon as possible (in Δ seconds for instance). The interception point is the intersection point between the extension of the velocity

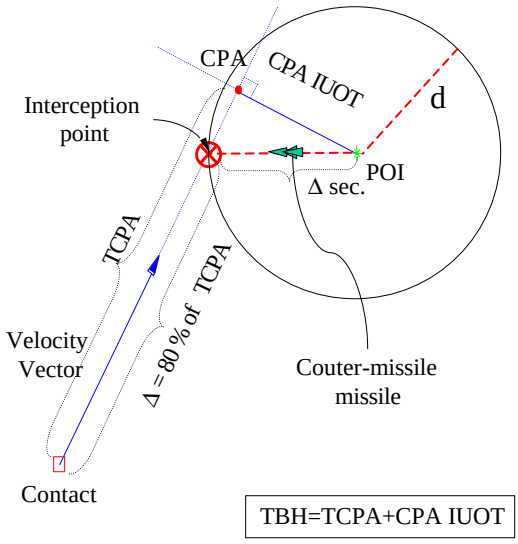


Figure 2: *Interception point between the missile and the point of interest (POI)*

vector of the missile and the circle having the *POI* as center and d as radius, where d is the distance crossed by the counter-missile missile in Δ seconds (see Figure 2). In general Δ must be smaller then the *TCPA*, or else it would be too late to counter the missile. In Figure 2, Δ is 80% of the *TCPA*.

2.3 The threat stabilization problem

The closest point of approach (CPA) concept described in Section 2.2 is useful if we make the assumption that the missile will maintain the same direction both during its trajectory to the *CPA*, and during its trajectory from the *CPA* to the *POI*. Unfortunately, this remains unrealistic in real-world situations. Generally, a launched missile never goes in a straight line to its target. Instead, it will constantly maneuver while approaching the target. For example, the *Sea Skimming* missiles such as the *Exocet*, the *Harpoon* (Sub-Sonic) and the *SS-N-22 "Sunburn"* (Super-Sonic) maneuver above the water surface. They are so close to the water surface so that they are too difficult to be detected. A *High Diving* missile such as the *AS-4 "Kitchen"* is another type of attacking missiles. It maneuvers at high altitude and once it is above its target, it picks down on it at a very high speed.

As the missile will maneuver during its approach, the values of *CPA*, *TCPA*, *CPA IUOT* and *TBH* will be constantly changing in such a way that it will be impossible to classify a set of threats related to different missiles. In addition, in the case of multiple *POIs*, it is difficult to determine which among the *POIs* is really threatened by a specific missile.

Figure 3 shows two points of interest, POI_1 and POI_2. The missile moves along an

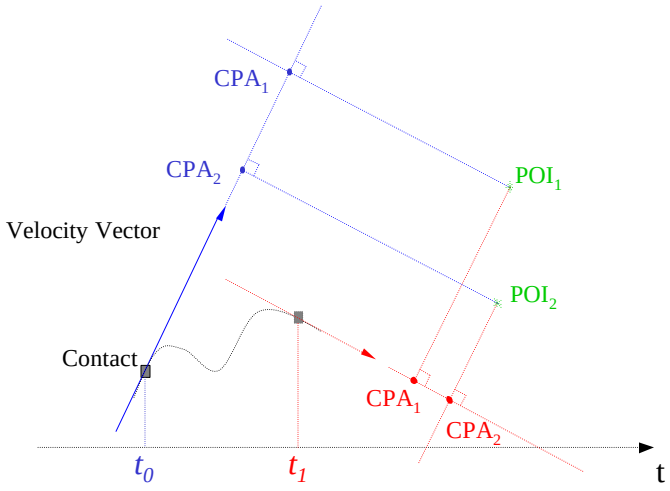


Figure 3: *A maneuvering behaviour of the missile makes the CPA concept useless*

undulating trajectory. At time t_0 the missile should reach the closest point of approach to POI_2 (CPA_2) before that of POI_1 (CPA_1). At time t_1 that missile has changed its direction. This resulted in changing the order of the CPAs. Now the CPA_1 should be reached before the CPA_2. At time t_0 the POI_2 was more threatened than POI_1. At time t_1, POI_2 became more threatened than POI_1.

The change of direction of the missile resulted in changing its threat values regarding two different points of interest. In war situations, it is critical to know which of the two points of interest is really the intended target. In addition, a maneuvering behaviour makes it impossible to determine a precise interception point to engage the missile. For all these reasons, it is important to *stabilize* the threat value associated with the missile all over its trajectory. One way to do this, is to smooth this trajectory in order to obtain a straight line. This is the estimated direction of advance of the missile. Such a straight line is necessary to compute a *threat value* based on the *CPA* and *TCPA* values.

3. Neural approach

Our approach to threat stabilization aims at smoothing the trajectory of a maneuvering missile. It is based on a special class of neural networks, the so-called *Self Organizing Maps* (SOM) [5]. The following gives a general introduction to neural networks, followed by a description of a SOM, and its application to threat stabilization. Readers who are already familiar with neural networks and SOM can skip Sections 3.1 and 3.2 and go directly to read Chapter 4.

3.1 Neural networks

Neural networks are used in numerous research programs [6, 7] for different applications such as:

- Investment analysis in order to predict shares prices;
- Handwriting recognition used in banks to recognize clients' signatures or in post offices in order to recognize zip codes written on letters [8];
- Pattern recognition and automatic classification [9, 10];
- Image and signal processing [11];
- Process control and monitoring (nuclear plants, aircraft systems, networks);
- Marketing.

There is no universal definition of neural networks. The most popular one is given in [12] where it is admitted that, a neural network is composed of a number of processing units (nodes of the network) that simulate the activity of an animal neuron. The units are connected via communication channels (connections) in order to transmit numerical information (weights of the connections). A processing unit is able to handle data stored in a local memory and the weights of the connections. Neural networks were created with the aim of simulating some behaviours of the human brain, in order to better understand its functions. Most neural networks require a training period in which the weights of the connections between the processing units are adjusted. Each sample of the training set is presented as an entry to the network and has an effect on the adjustment of the weights. It is important to note that processing the units can be done in parallel because they are totally independent. Hence, a neural network can be thought of as a kind of multi-processor machine. The processing units are organized by layers. A neural network has at least two layers, the *input* and *output layers*, and possibly several intermediate layers called *hidden layers*. A layer contains any number of processing units that are connected to those of the preceding and the following layers. Among the most well-known architectures, we can mention the simple Perceptron, the multi-layer Perceptron and the RBF (Radial Basis Function) network. For illustration purposes, we present a simple Perceptron in order to learn the Boolean function "$\geq$" over $\{0,1\}$ (Figure 4).

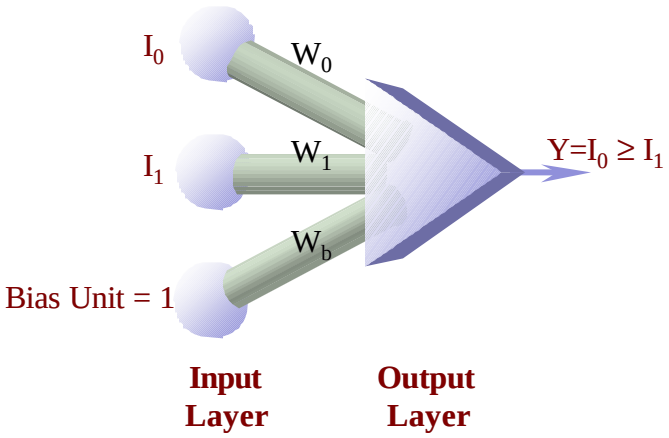


Figure 4: *A simple Perceptron to learn the Boolean function "$\geq$" over $\{0, 1\}$*

This network has two entries I_0 and I_1 and a third entry called *Bias Unit*[1] which is constantly equal to 1. Very often, this entry is used in order to guarantee optimal results after a training period (see Subsection 3.1.2). The three entries form the input layer. The output layer has only one output Y. The connections weights are W_0, W_1 et W_b. For each pair of entries the output is binary and has the following values:

$$Y = \begin{cases} 1 & \text{if } I_0W_0+I_1W_1+W_b > 0, \\ 0 & \text{if } I_0W_0+I_1W_1+W_b \leq 0 \end{cases} \tag{3}$$

3.1.1 Initialization of a neural network

The initialization of a neural network is to assign the first values to the weights of the connections. This is the initial state of the weights before the training phase. In general, the assigned values are randomly chosen in the space of values. The initial values of W_0,W_1 et W_b are randomly chosen in [0,1] (W_0 = 0.8560, W_1 = 0.4902 and W_b = 0.8159).

3.1.2 Training a neural network

In general, a neural network needs a training algorithm in order to adjust the weights of connections in such a way that, for each entry, the network generates the desired output. The set of entries used in the training period is

[1]The output of the network corresponds to a hyperplane that divides the entry set into two parts. To avoid this hyperplane being obliged to cross the origin (equation: $I_0W_0+I_1W_1 = 0$), the bias unit (whose weight is W_b), which is constantly equal to 1 is used, and the equation becomes: $I_0W_0+I_1W_1+W_b = 0$.

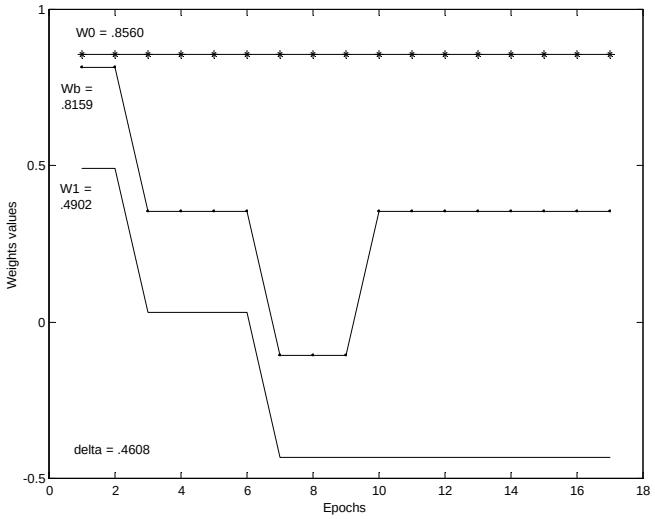


Figure 5: *Weights adjustment during training*

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
I_0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
I_1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Y	1	0	1	1	1	0	1	1	1	0	1	1	**1**	**0**	**1**	**1**
D	1	1	1	1	1	1	1	1	0	0	1	1	**1**	**0**	**1**	**1**

Table 1: *Values of desired and real outputs during the training.*

called the training set. In our example, we train the weights W_0, W_1 et W_b in order to obtain Y = 1 if $I_0 \geq I_1$, and Y = 0 otherwise. To this end, we use the training rule of the Perceptron that adjusts the weights proportionally to the difference between the output Y and the desired output D. The weights adjustment is defined by the following equation: $\Delta W_i = \delta$(D-Y)I_i, where δ is the learning rate. In this example, the value of δ is chosen randomly and is equal to 0.4608.

Figure 5 illustrates the weights adjustment during the training period. Their final values are W_0 = 0.8560, W_1 = -0.4313 and W_b = 0.3552. The training stopped after 16 iteration steps. In Table 1 are shown the real and desired output values during the training. We can notice that, during the iteration steps 13 through 16, the real and desired outputs are equal, the matter why the training stopped.

The next section gives a general description of a special type of neural networks commonly called the *Self Organizing Maps*. Such a network is used

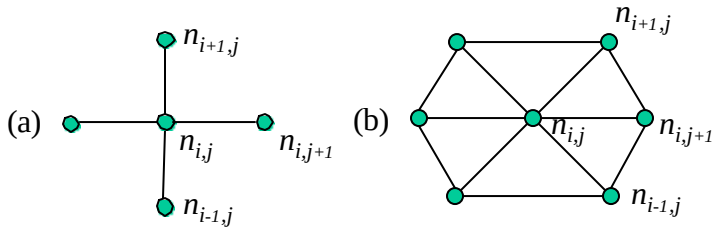


Figure 6: *The topological neighborhood can be rectangular (a) or hexagonal (b)*

in the threat stabilization process that will be described in Chapter 4.

3.2 The Self-Organizing Maps (SOM)

The *Kohonen's Self Organizing Maps* (SOM) [5] are essentially used in order to build simple representations of complex data such as image or sound data. Learning data with a Kohonen-type network comes down to approximate the initial data set by building another set of data with lesser dimensions. The objective of using a SOM is to have a simpler model of the initial data and hence to decrease the computational requirements on these data. A SOM network is fundamentally different from the one given in Section 3.1, in the sense that the desired outputs are unknown. The main characteirstics of a SOM is the topolological relation that exists between the outputs of the network. The role of this relation in the threat stabilization process will be detailed in the following sections.

3.2.1 Initialization of a SOM

Let $\Re^d$ be the initial data space where each vector in this space can be written as v_i=$\{x_1,\ldots,x_d\}$. The initial data set is composed of m vectors. The new data set, which represents some percentage of the initial data set (10% for example) is randomly chosen in $\Re^d$. The vectors c_i of this set are called *codebooks*. A topological neighborhood is defined between all the codebooks in order to obtain a two-dimension map (h×w), so that each codebook c_j will correspond to a node $n_{k,l}$ on the map (k=1..h, l=1..w). The topological neighborhood can be either rectangular or hexagonal, as shown in Figure 6.

Of course, at the beginning, the topological neighborhood has no sense since the codebooks are random vectors in $\Re^d$. The objective is then to learn the initial data set (training set) so that the codebooks form a meaningful map.

3.2.2 Training a SOM

The training period consists in scanning the training set and for each training sample v_i to do the following processing.

Let c_m be the codebook that corresponds to the best-matching node $n_{r,s}$ of the map, that is, the nearest codebook to v_i:

$$\|v_i - c_m\| = \min_j \{\|v_i - c_j\|\} \tag{4}$$

The training rule is applied to all codebooks c_j whose corresponding nodes belong to a neighborhood set N_m around the node $n_{r,s}$:

$$c_j(t+1) = c_j(t) + h_{mj}(t)[v_i - c_j(t)] \tag{5}$$

where t is the epoch number and $h_{mj}(t)$ the so-called *neighborhood kernel*:

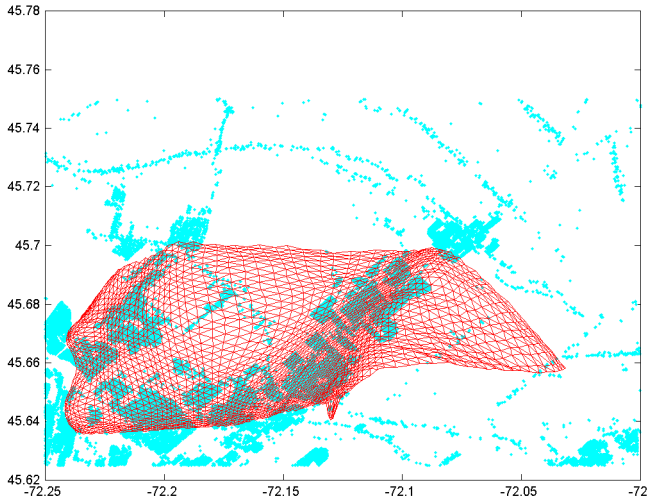
$$h_{mj}(t) = \begin{cases} 0 & \text{if } n_{u,v}\text{, node of } c_j \notin N_m, \\ h(\|r_m - r_j\|) & \text{otherwise.} \end{cases} \tag{6}$$

and finally, r_m = (r,s) and r_j = (u,v).

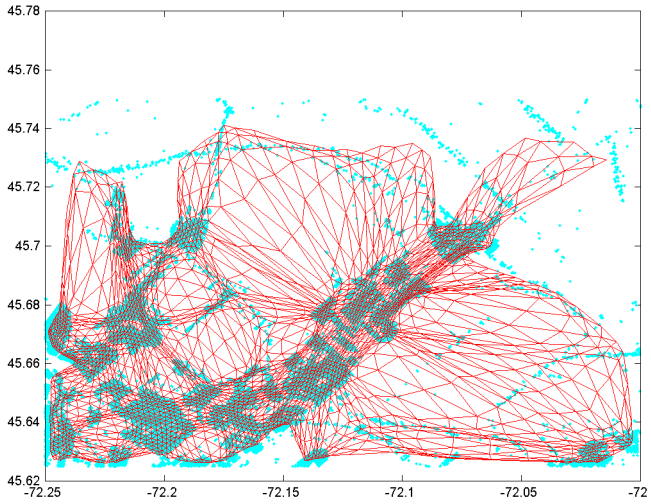
This will result in moving all codebooks c_j toward the training sample v_i. The function h is a monotonically decreasing function ($0 < h < 1$). The radius of N_m is also a monotonically decreasing function. In general, a SOM is trained with two different sets of parameters. The first set is used in a short training phase in order to *unfold* the net. In this set, the radius of N_m is rather high ($\approx h$ or w). In the second set, the parameters are user-defined. Generally, the radius of N_m is smaller than w and h. This set is used in a second training phase that leads the codebooks to their final state. In both training phases, the radius of N_m will gradually decrease to reach 1. Hence, during the training, the number of updated codebooks and their displacement amounts will gradually decrease with time.

In Figure 7, we show an example of a SOM over a data set in $\Re^2$. The data set corresponds to the distribution of houses in an urban area near Montreal. They are represented by points.

Figure 7-a shows how the SOM *unfolds* after the first training phase.
Figure 7-b shows the final state of the SOM. The initial data set is approximated by a set of codebooks organized as a net. The number of codebooks is about 10% of 24487, the total number of points. The resulted SOM is a 70×35 net.



(a) First training phase



(b) Second training phase

Figure 7: *Example of a self organizing map (SOM) in* $\Re^2$

4. Application to threat stabilization

The SOM network was used in order to tackle the threat stabilization problem described in Section 2.3. The whole process is described in details in the following sections.

4.1 Entries

The SOM network takes a couple (x,y) as an entry. It corresponds to a reported position of the missile. It can be obtained from a *Target Tracking Radar*. Such a radar consists of a transmitter, a receiver, an antenna system, a display device and a computer system. The transmitter sends a high-energy signal, which is reflected by solid objects, and sent back to the receiver. The time taken by the signal to travel, hit the object and come back allows the computer system to determine the position of the missile by giving its range and angle (Polar coordinates). The couple (x,y) can be computed by making a simple transformation from Polar to Cartesian coordinates. The computer system is also capable of tracking the missile, that is, predicting its future positions on the basis of its flight parameters such as speed and direction. This enables the radar to keep tracking the missile by moving itself to point at it.

It is important to notice that the reported positions are always subject to uncertainty. For example, if the missile is moving along a straight line, the positions reported by the radar will never form such a line. The SOM network can manage this uncertainty factor, in the sense it is able to compute the direction of advance whatever the degree of uncertainty of the entries. However, this uncertainty factor should affect the results. In Section 5.2, results are compared for two sets of simulated data. The first one, the reference data set, simulates an oscillating trajectory of a missile. The second is obtained by adding some noise to the reference data set in order to increase the degree of uncertainty related to the reported positions of the missile.

4.2 Initialization

In addition to classical parameters needed to train a SOM (see Subsection 3.2.2), the following parameters are necessary in this application:

- The *frequency f* defines the number of reported positions per second. For example, if the frequency is 0.5, then a position is reported every two seconds. This parameter must be taken into account because it is dependent on the system that reports the positions. For example, some radars are able to report positions with a frequency of 1 (one position reported per second), 0.5 (one position is reported every 2 seconds) or 0.25 (one position is reported every four seconds).

- A *temporal window T* specifies the number of seconds taken into account by the stabilization process at once. This parameter is defined in order to limit the number of positions taken into account by the stabilization process to train the SOM and hence to limit the computational cost. For example, the temporal window can be set

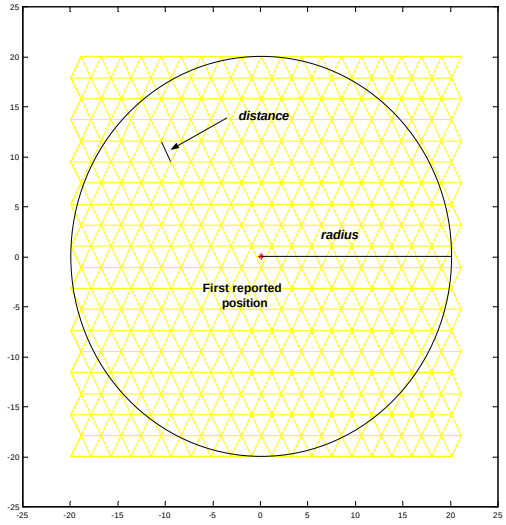


Figure 8: *The initial version of the SOM in threat stabilization*

to 20 seconds, that is, if the frequency is 0.5, then only 10 reported positions will be taken into account at once by the stabilization process.

The initial codebook vectors are set to form a hexagonal grid as shown in Figure 8. The region covered by the net is defined as the following:

- First the speed s of the missile is estimated. It corresponds to the distance between two reported positions divided by the time taken by the missile to travel from the first to the second reported position.
- The $radius$ of the covered region by the net corresponds to the speed s multiplied by the temporal window T ($radius$=$s \times T$). This dimension is sufficient since it corresponds to the maximum distance that could be crossed by the missile by maintaining the same speed s and by going in a straight line during T seconds.
- The dimensions of the net are $w \times h$, where w=T/$(4f)$ and h=w. This means that the distance between two codebooks is four times the distance between the first two reported positions in a temporal window. The ratio 1/4 was chosen on the basis of experimental results.

4.3 Real-time training

In general, a SOM is trained by using a single training set (see Subsection 3.2.2). This set is unchanged all over the training period. In this application, the SOM training is different in that it is real-time, that is, every second, the current version of the SOM is

trained by all available reported positions during the last T seconds. At the beginning of the training, the SOM is trained by only the first reported position. Next, it is trained by using the first two reported positions, and so on. It is important to notice that the initial version of the SOM is already a hexagonal grid (see Figure 8). Hence, only one training phase is necessary since there is no need to unfold the net. In the training process, the function h was chosen to be Gaussian, as described in [13]:

$$h_{mj}(t) = h(\|r_m - r_j\|) = \alpha(t) e^{-(\|r_m - r_j\|)^2 / 2\sigma^2(t)} \tag{7}$$

where $\sigma(t)$ is the radius of N_m at epoch t. The function α is monotonically decreasing: $\alpha(t) = \alpha(0)(1 - \frac{t}{itlen})$, where $itlen$ is the total number of epochs and $0 < \alpha(0) < 1$.

4.4 Feature extraction

During the training, an algorithm estimates the direction of advance of the moving entity each time a new position is reported. To this end, some relevant features are extracted from the SOM. These features identify regions of *high activity* of the SOM. When the trajectory is a curved line, a high-activity region is formed around the center of this curvature (Figure 9). In such a region, codebooks are subjected to *centripetal* attraction by the points of the trajectory (reported positions). This results in a concentration of codebooks in some regions. Hence, the edges of the SOM are the smallest in these regions. In other words, in order to identify such regions, it is sufficient to select the *small* edges of the SOM. It is possible to choose a threshold to this end, but it must be defined by the user. This is inconvenient in a real-time context. The process of selection needs to be fully automatic during the training of the SOM, as shown by the following:

- The smallest edge of the SOM is added to an empty list ξ at the beginning of the process.
- For each new reported position of the moving entity:
  - if at least one of the edges in ξ changes, that is, the coordinates of at least one of its codebooks (two tips of the edge) are updated by the training of the SOM, then ξ remains unchanged;
  - if all edges in ξ are not altered by the training, then the smallest edge in the current SOM is added to the list. In Figure 9, the selected edges in the list ξ are illustrated by thick lines.

It is important to mention that it is only possible to add elements to the list. Deletion is not permitted. Even if an edge in the list is *stretched* during the training, and hence becomes longer than other edges in the SOM, it is kept on the list during the training. This is due to the fact that initially such an edge was selected because it has identified a high-activity region on the net. So, even if it is stretched during the training, it still identifies the same region. In Figure 10, one can easily distinguish some edges in the list that are longer than other edges in the SOM. The SOM is shown 9 seconds

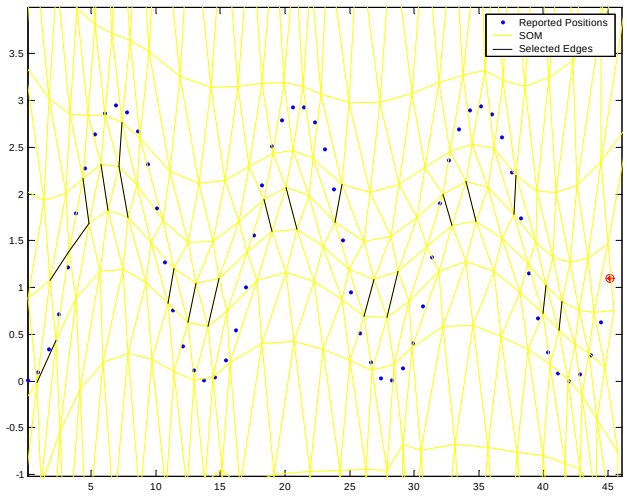


Figure 9: *The high-activity regions are formed around the centers of curvatures of the trajectory*

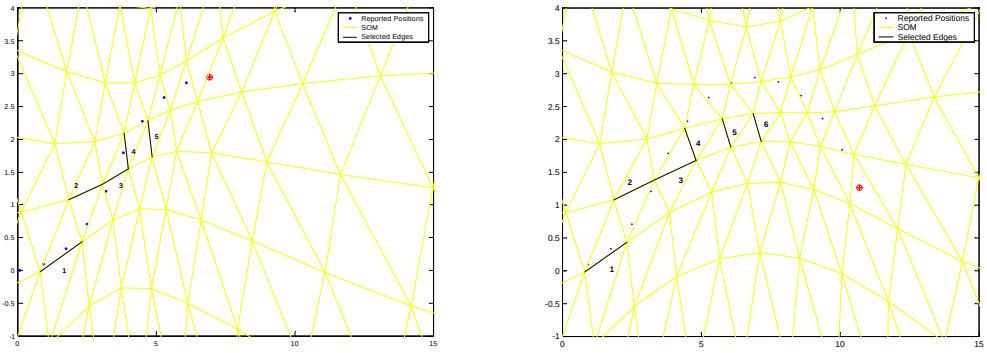
(Figure 10-a) and 14 seconds (Figure 10-b) after t_0, the time of the first reported position. The edges (2), (3), (4) and (5) are stretched during the training. At time t_0+14, the edge (5) was not affected by the training (it remains the same as at time t_0+13). This is the reason why a new edge (6) was added to the list.

4.5 Kinematics smoothing

Each time a new position of the moving entity is reported, the list of edges ξ is used in order to compute an equation of a straight line. This is the current estimation of the direction of advance of the moving entity. To this end, the least-squares regression line (see Annex A for details) is computed by using the centers of the edges in the list ξ (Figure 11). The center c_{ij} of an edge formed by two codebooks c_i = (x_i, y_i) and c_j = (x_j, y_j) is equal to ($\frac{x_i+x_j}{2}$, $\frac{y_i+y_j}{2}$). Each time a new edge is added to the list, a new least-squares regression line is computed. The list ξ is initialized every T seconds, where T is the temporal window described in Section 4.2. In order to avoid a perturbation of the regression line when the list is initialized, this line is computed by using the edges of the latest list before the initialization, and then the edges of the new list after the initialization. Hence, from 0 to T seconds, the regression line is computed by using only one list. From T+1 on, it is computed by using the latest two lists.

4.6 Discussion

The proposed approach to threat stabilization depends on numerous factors. These factors directly influence the performance of any approach to threat stabilization



(a) Time: t_0+9 seconds. (b) Time: t_0+14 seconds.

Figure 10: *The edges of the SOM are stretched by the training*

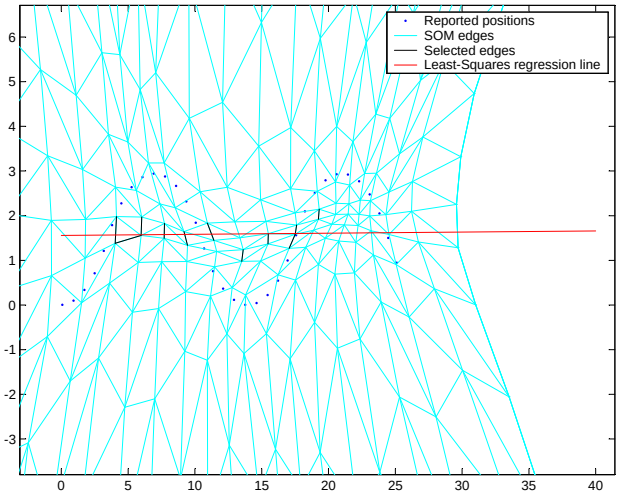


Figure 11: *The regression line computed over the centers of the edges in the list ξ*

(mathematical, neural-based, etc.):

- *Oscillation Frequency*. This parameter depends on the capacity of a missile to make sharp turns within a small period of time. Sharp turns (angles) with different forms increase the difficulty of determining the main features of the trajectory. Hence, stabilizing the direction of advance of the missile may take much more time than that of smooth and regular turns.

- *Observation Frequency*. The reported positions are generally generated from sensors such as radars. The number of reports within a period of time has a direct impact on the time of stabilization of the direction of advance. If the trajectory is over-sampled, some performance and computational issues may arise. On the other hand, if it is under-sampled, it may be too difficult to characterize the trajectory within a short period of time.

- *Temporal Window*. Whatever the method used to stabilize the direction of advance, only a limited number of positions can be considered, otherwise one may encounter the same problems as those of over-sampling. Generally, one chooses a temporal window, that is, a period of time during which the new reported positions will be taken into account by the stabilization process. If this window is too small, it will be unsufficient neither to characterize the trajectory nor to stabilize the direction of advance. A trade-off for the size of this window is necessary for optimal results.

All these factors must be taken into account before choosing a stabilization approach. For example, mathematical approaches would be misapplied in the case of irregular and sharp oscillations. On the other hand, they are well adapted to regular and smooth oscillations. Neural approaches are well adapted to irregular oscillations and uncertain data, but mathematical approaches will give better solutions with regular oscillations.

It is also worth mentioning that the SOM approach could be more costly than other mathematical approaches from a computational point of view, but it could be used for other purposes than threat stabilization. For instance, the features extracted from the trajectory could be used in order to characterize the type of maneuver of the moving entity such as an attacking maneuver of an aircraft.

5. SS-N-22 missile simulation

In the next sections, the proposed stabilization method is applied to a set of data simulating the trajectory of a particular missile, the SS-N-22.

5.1 SS-N-22 maneuvering (simulated data)

The trajectory of the SS-N-22 is depicted in Figure 12. These are the parameters used in order to generate the simulated data:

- the minimum angular offset is 0° and the maximum angular offset is 45°. Hence, the angular variation is between 0 and 45°. The amount and duration of this variation are chosen randomly;
- the initial speed of the entity is 1650 Kts;
- the initial heading is 90° from the y axis (horizontal heading);
- the maximum number of Gs caused by the maneuvers is 10; this is realistic for a launched missile such as the *SS-N-22*;
- the total duration of the trajectory is 45 seconds;
- finally, the frequency of the reported positions is 1, that is, one position is reported every second.

In this example, the temporal window T is 15 seconds, that is, 3 self organizing maps (SOM) are needed to cover the whole trajectory (45 seconds). The choice of a 45-second trajectory is motivated by the fact that the operators in a ship are generally interested in watching all moving entities that are closer than 20 km to the ship. This is approximately the distance crossed by an SS-N-22 missile in 45 seconds.

Figure 13 shows a comparison with the least squares (LS) algorithm for the same temporal window T (15 seconds). It shows the real heading of the missile and the stabilized heading of both the SOM and the LS algorithms. The SOM algorithm gave better results than the LS algorithm.

In order to compare these results, the closest-points of approach (CPA) of the SS-N-22 were computed according to two different points of interest (POI), poi_1 = (40, 30) and poi_2 = (40, 24). The distance between these two points was 6 km. In Figure 14 are represented the changes of threat values (computed by ordering the CPAs) of the moving entity according to the two points of interest. A change in the value (from 0 to 1 or from 1 to 0) of the represented functions corresponds to a change in the order of the CPAs. The results are represented for the reported trajectory of the missile and the stabilized headings of both the SOM and the LS approaches. Let us notice that the SOM offers more stability than the LS, even if the two points of interest are separated by 6 kms.

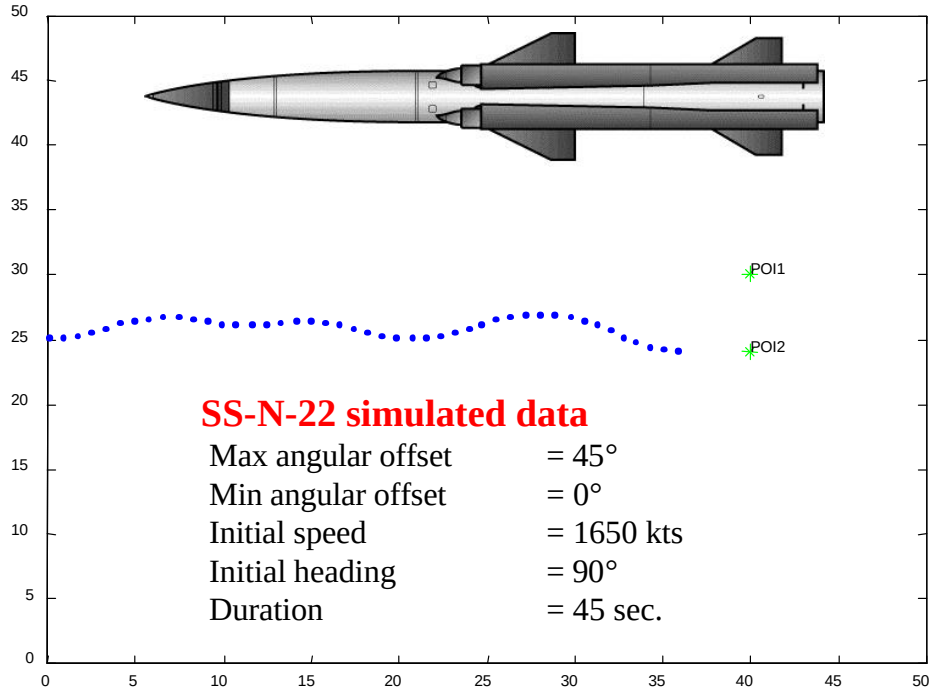


Figure 12: *A simulated trajectory of an SS-N-22 missile*

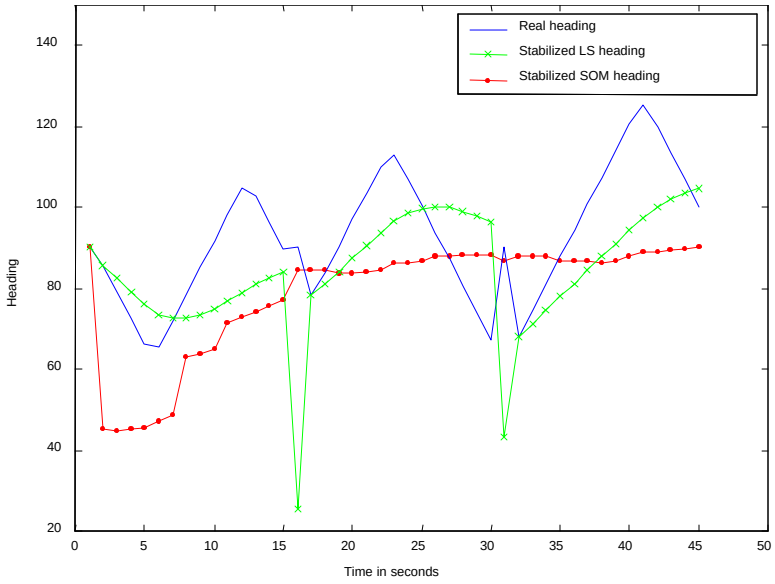


Figure 13: *Comparison between the stabilization of the least squares (LS) algorithm and the self organizing maps (SOM) algorithm*

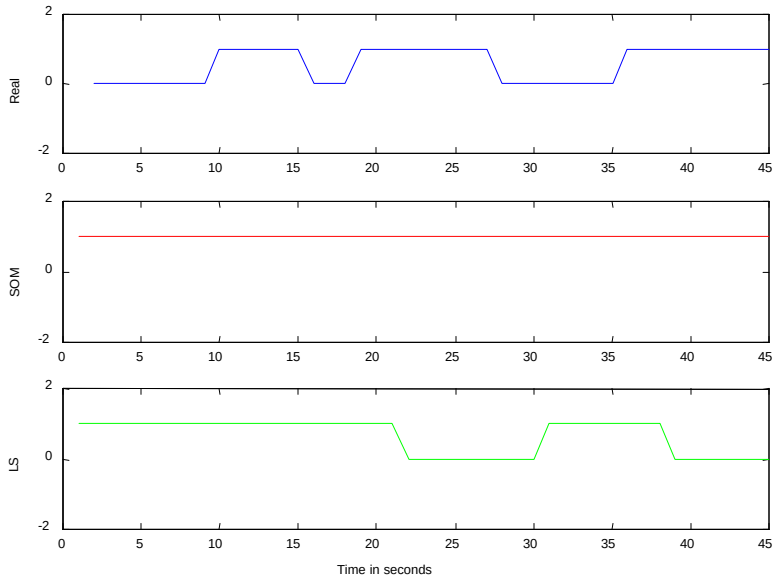


Figure 14: *Changes of the threat values due to a maneuvering behaviour of the moving entity*

5.2 Adding uncertainty to the initial data set

In this section, some noise is added to the initial trajectory of the missile. In real situations, it is impossible to obtain the real positions of such a moving entity as shown in Figure 12. Even if the positions are reported by a sophisticated radar, they are subject to uncertainty. In order to measure its impact on the stabilization results, an uncertainty factor is added to each reported position. A reported position is randomly chosen in a region defined by a circle of 1 km-radius. The result is shown in Figure 15.

The new stabilization results of the SOM and LS algorithms are shown in Figure 16.

The changes of threat values according to the same two points of interest (poi_1 = (40, 30) and poi_2 = (40, 24)) are represented in Figure 17. It is interesting to notice how the added uncertainty factor has impacted the stabilization of the threat values in comparison with the results shown in Figure 14. The stabilization by the LS algorithm was more impacted than that of the SOM due to the introduction of uncertainty to the initial data. However, the SOM stabilization algorithm still gives better results than the LS algorithm. The introduction of uncertainty will probably have more impact on the stabilization method based mathematical approaches since they try to approximate the trajectory by an analytical function based on its shape.

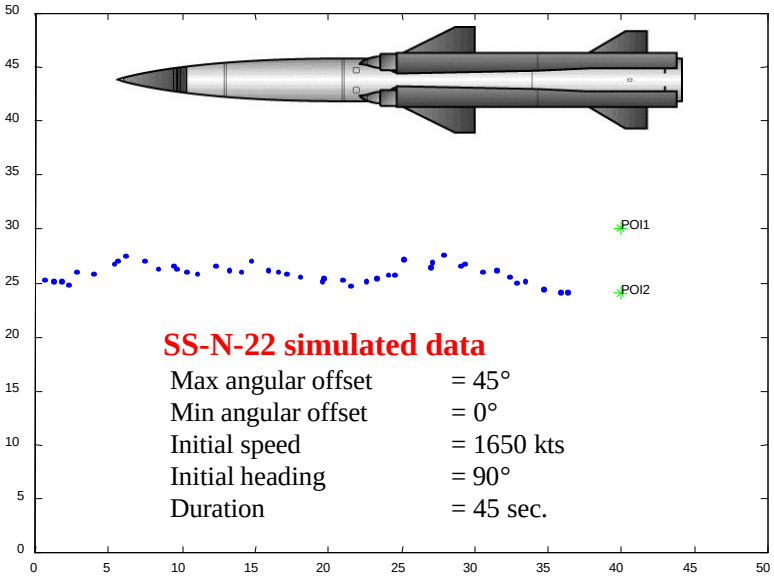


Figure 15: *Adding uncertainty to the first data set*

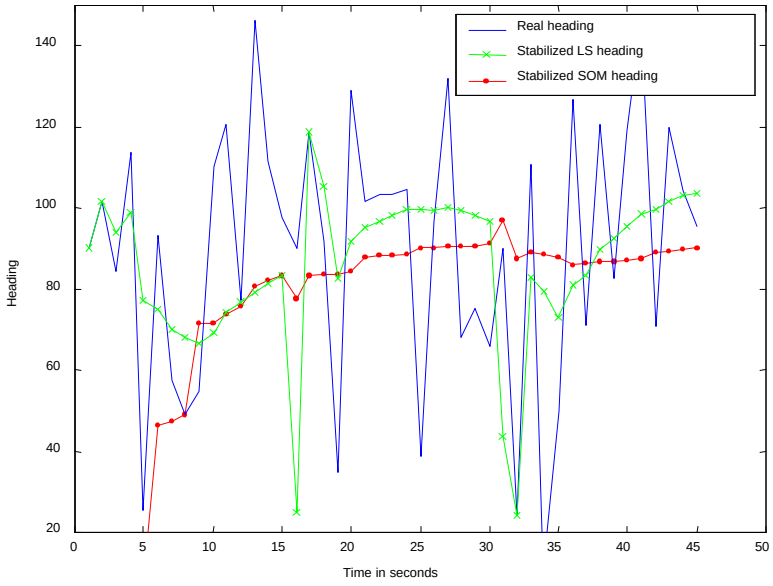


Figure 16: *The new results of stabilization for the new data set*

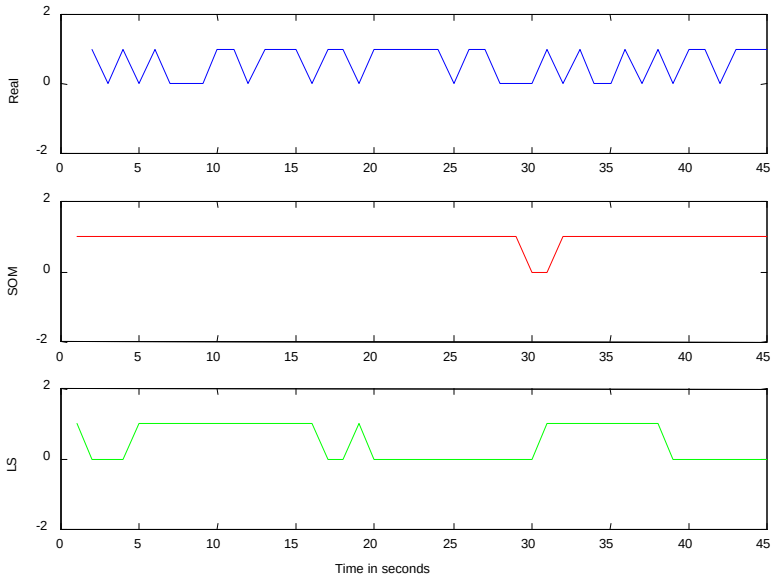


Figure 17: *New changes of the threat values for the new data set*

6. Conclusions and future work

As an important part of the situation analysis process, the threat evaluation process is always dependent on the real time evolution of a situation in the environment. In this work, the real time aspect is met by the (rapid) change in the kinematics of a moving entity. Based on the closest point of approach (*CPA*) concept, the evaluation of a stable threat value for such an entity remains a difficult task, and needs a stabilization process.

An algorithm for kinematics smoothing was developed based on the self organizing maps (SOM) neural network. This approach is very innovative since it uses a SOM in real time. Furthermore, such a network is well adapted to uncertain data corresponding to irregular trajectories that could be done by the moving entity. In fact, the topological relation between the output nodes makes it possible for the network to give the same result for a large set of similar trajectories. This makes the network less sensitive to noisy data, which is inevitable when the data are collected from sensors.

The proposed algorithm was tested on two sets of data. The first set is a simulation of a "SS-N-22" missile trajectory, and the second is built up by adding an uncertainty factor to the first. The results were conclusive. The algorithm succeeded in stabilizing the direction of advance of the missile. It also gave better results in comparison with another algorithm based on the least-squares method.

As future work, one could investigate the stabilization of the direction of advance in a three-dimensional space. This is due to new generations of missiles able to make maneuvers along x, y and z axes. In order to tackle such a problem, the use of a three-dimensional SOM will become necessary.

Finally, the proposed threat stabilization process could be used in another important part of the threat evaluation process, the *intention recognition*. This could be based on the extraction of specific features from the SOM in order to characterize the maneuvers of the moving entity.

Annex A
Least-Squares Regression Line

The least-squares regression line approximates a set of points $(X_1, Y_1), \ldots, (X_n, Y_n)$ (See Figure A.1). In order to determine the equation of such a line, $y = ax + b$, we use the least-squares criterion. It aims at determining the cœficients a and b as

$$Q = \sum_{i=1}^{n} [y_i - (ax_i + b)]^2$$

has minimum value.

Let $\frac{\partial Q}{\partial a} = 0$ and $\frac{\partial Q}{\partial b} = 0$ to find a and b.

$$\begin{aligned} Q &= \sum_{i=1}^{n} [y_i - (ax_i + b)]^2 \\ &= \sum_{i=1}^{n} [y_i^2 + (a + bx_i)^2 - 2y_i(a + bx_i)] \\ &= \sum_{i=1}^{n} [y_i^2 + a^2 + 2abx_i + b^2x_i^2 - 2y_ia - 2bx_iy_i] \end{aligned}$$

$$\begin{aligned} \frac{\partial Q}{\partial a} &= \sum_{i=1}^{n} (2a + 2bx_i - y_i) \\ &= 2na + 2(\sum_{i=1}^{n} x_i)b - 2\sum_{i=1}^{n} y_i \end{aligned}$$

$$\frac{\partial Q}{\partial b} = 2a\sum_{i=1}^{n} x_i + 2b\sum_{i=1}^{n} x_i^2 - 2\sum_{i=1}^{n} x_iy_i$$

$$\begin{aligned} \frac{\partial Q}{\partial a} = 0 \Rightarrow \quad & na \quad + (\sum_{i=1}^{n} x_i)b = \sum_{i=1}^{n} y_i \\ \frac{\partial Q}{\partial b} = 0 \Rightarrow \; & (\sum_{i=1}^{n} x_i)a \quad + (\sum_{i=1}^{n} x_i^2)b = \sum_{i=1}^{n} x_iy_i \end{aligned} \tag{A.1}$$

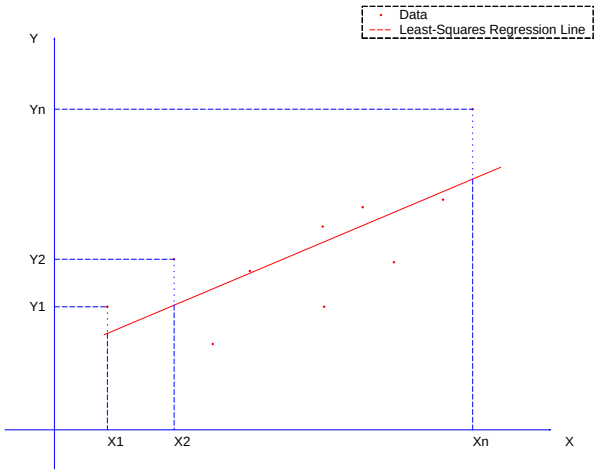


Figure A.1: *The least-squares regression line*

The solution of Equation A.1 is:

$$a = \frac{\sum_{i=1}^{n} y_i - (\sum_{i=1}^{n} x_i)b}{n}$$
$$b = \frac{n\sum_{i=1}^{n} x_i y_i - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2} \qquad \text{(A.2)}$$

References

1. Steinberg, A., Bowman, C. L., and White, F. E. (1998). Revisions to the JDL Data Fusion Model. In *Proceedings of the third NATO/IRIS Conference*, Quebec, Canada.

2. Roy, J. (2001). From Data Fusion to Situation Analysis. In ISIF, (Ed.), *Proceedings of the 4th Annual Conference on Information Fusion*, Montreal, Canada.

3. Waltz, E. and Llinas, J. (1990). Multisensor Data Fusion, ARCHTECH HOUSE Inc.

4. Roy, J., Breton, R., and Paradis, S. (2001). Human-Computer for the Study of Information Fusion Concepts in Situation Analysis and Command Decision Support Systems. In *Proceedings of SPIE on Signal Processing, and Target Recognition X*, Orlando, FL.

5. Kohonen, T. (1984). Self-Organization and Associative Memory, Springer-Verlag.

6. Anderson, J. A. (1995). An Introduction to Neural Networks, Cambridge, MA: The MIT Press.

7. Fausett, L. (1994). Fundamentals of Neural Networks: Architectures, Algorithms, and Applications, New Jersey, USA: Prentice Hall.

8. Ridder, D. (1996). Shared weights neural networks in image analysis, Delft, Netherlands: Delft University of Technology.

9. Bishop, C. M. (1995). Neural Networks for Pattern Recognition, Oxford, UK: Oxford University Press.

10. Ripley, B. D. (1996). Pattern Recognition and Neural Networks, Cambridge, MA: Cambridge University Press.

11. Cichocki, A. and Unbehauen, R. (1993). Neural Networks for Optimization and Signal Processing, New York, NY: John Wiley & Sons.

12. Hinton, G. E. (1992). How Neural Networks Learn from Experience. *Scientific American*, **267**, 144–151.

13. Kohonen, T., Hynninen, J., Kangas, J., and Laaksonen, J. (1996). SOM PAK: The Self-Organizing Map program package. (Technical Report A31).

This page intentionally left blank.

Distribution List

INTERNAL DISTRIBUTION

DRDC Valcartier TR 2002-239

1 - Director General

3 - Document Library

1 - Head/Decision Support Systems

1 - J.M.J. Roy

1 - Dr R. Breton

1 - Dr P. Valin

1 - Head/Information and Knowledge Management

1 - R/Visualization & Geo-spatial Systems

1 - R/Knowledge Management Systems

1 - R/Informations System Warfare

1 - Head/System of Systems

1 - R/Metrics & Experimentation

1 - R/Distributed Synthetic Environment

1 - R/System Engineering & Architecture

1 - Dr M. Allouche (author)

1 - Dr A. Guitouni

1 - Mr A. Sahi

1 - LCdr É. Tremblay

1 - Maj. B. Deschênes

1 - LtCol. B. Carrier

1 - M. Blanchette

1 - S. Paradis

INTERNAL DISTRIBUTION (cont'd)

DRDC Valcartier TR 2002-239

1 - Dr A. Benaskeur

1 - F. Rhéaume

1 - M. Bélanger

1 - J. Berger

1 - Dr A. Boukhtouta

1 - Dr A.L. Jousselme

1 - Dr A.C. Bourry-Brisset

1 - Dr H. Irandoust

1 - P. Maupin

1 - C. Daigle

EXTERNAL DISTRIBUTION

DRDC Valcartier TR 2002-239

1 - DRD KIM (PDF)

1 - Director Science & Technology Maritime(DSTM)

1 - Director Science & Technology Land (DSTL)

1 - Director Science & Technology Air (DSTA)

1 - Director Science & Technology C4ISR (DSTC4ISR)

1 - Director Maritime Requirements Sea (DMRS) 4

1 - Director Maritime Requirements Sea (DMRS) 6

1 - Director Aerospace Requirements (DAR) 4

1 - Director Aerospace Requirements (DAR) 4-2

2 - Director Maritime Ship Support (DMSS) 6

2 - Director Maritime Ship Support (DMSS) 8

3 - DRDC - Atlantic:

Attn: Dr. Bruce MacArthur
Attn: Dr. Dr. Bruce Chalmers

2 - CF Maritime Warfare School CFB Halifax
Halifax, Nova Scotia

Attn: TAC AAW
OIC Modeling and Simulation

2 - Canadian Forces Naval Operations School CFB Halifax
Halifax, Nova Scotia

Attn: Tactic
CT AWW

EXTERNAL DISTRIBUTION (cont'd)

DRDC Valcartier TR 2002-239

1 - Canadian Forces Naval Engineering School CFB Halifax
Halifax, Nova Scotia
Attn: CSTC

1 - Operational Requirements Analysis Cell CFB Halifax
Halifax, Nova Scotia
Attn: Commanding Officer

1 - Canadian Forces Fleet School CFB Esquimalt
Esquimalt, British Columbia
Attn: Commanding Officer/WTD

1 - Operational Requirements Analysis Cell CFB Esquimalt
Esquimalt, British Columbia
Attn: Commanding Officer

DOCUMENT CONTROL DATA

1. ORIGINATOR (name and address)	2. SECURITY CLASSIFICATION
Defence R&D Canada - Valcartier 2459 Pie-XI Blvd. North Québec, Quebec G3J 1X5	(Including special warning terms if applicable) Unclassified

3. TITLE (Its classification should be indicated by the appropriate abbreviation (S, C, R or U)
A pattern recognition approach to threat stabilization (U)

4. AUTHORS (Last name, first name, middle initial. If military, show rank, e.g. Doe, Maj. John E.)
Mohamad Khaled Allouche

5. DATE OF PUBLICATION (month and year)	6a. NO. OF PAGES	6b .NO. OF REFERENCES
June 2006	26	13

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. Give the inclusive dates when a specific reporting period is covered.)
Technical report

8. SPONSORING ACTIVITY (name and address)
DRDC-RDDC Valcartier

9a. PROJECT OR GRANT NO. (Please specify whether project or grant)	9b. CONTRACT NO.
11bh	

10a. ORIGINATOR'S DOCUMENT NUMBER	10b. OTHER DOCUMENT NOS
TR 2002-239	N/A

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

- ☒ Unlimited distribution
- ☐ Restricted to contractors in approved countries (specify)
- ☐ Restricted to Canadian contractors (with need-to-know)
- ☐ Restricted to Government (with need-to-know)
- ☐ Restricted to Defense departments
- ☐ Others

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected.)

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

Threat evaluation is of great importance in the command and control decision making process. Due to highly dynamic changes in the environment, it becomes more and more difficult to assign a stable threat value to an entity. The present work aims at stabilizing the threat value assigned to a moving entity having a maneuvering behaviour. To this end, a special type of neural networks, the Self Organizing Maps, is used in order to extract some important features from the kinematics of the moving entity. The direction of advance of such an entity is computed based on the extracted features. The stabilization of such a direction of advance allows the stabilization of the threat value assigned to such an entity, especially when this value is based on the closest point of approach concept. The proposed approach is tested over a simulated data set and compared to other stabilization approaches such as the Least-Squares Regression Line method.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Threat Stabilization, Neural Networks, Kinematics Smoothing, Self-Organizing Maps

Defence R&D Canada

Canada's Leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

WWW.drdc-rddc.gc.ca