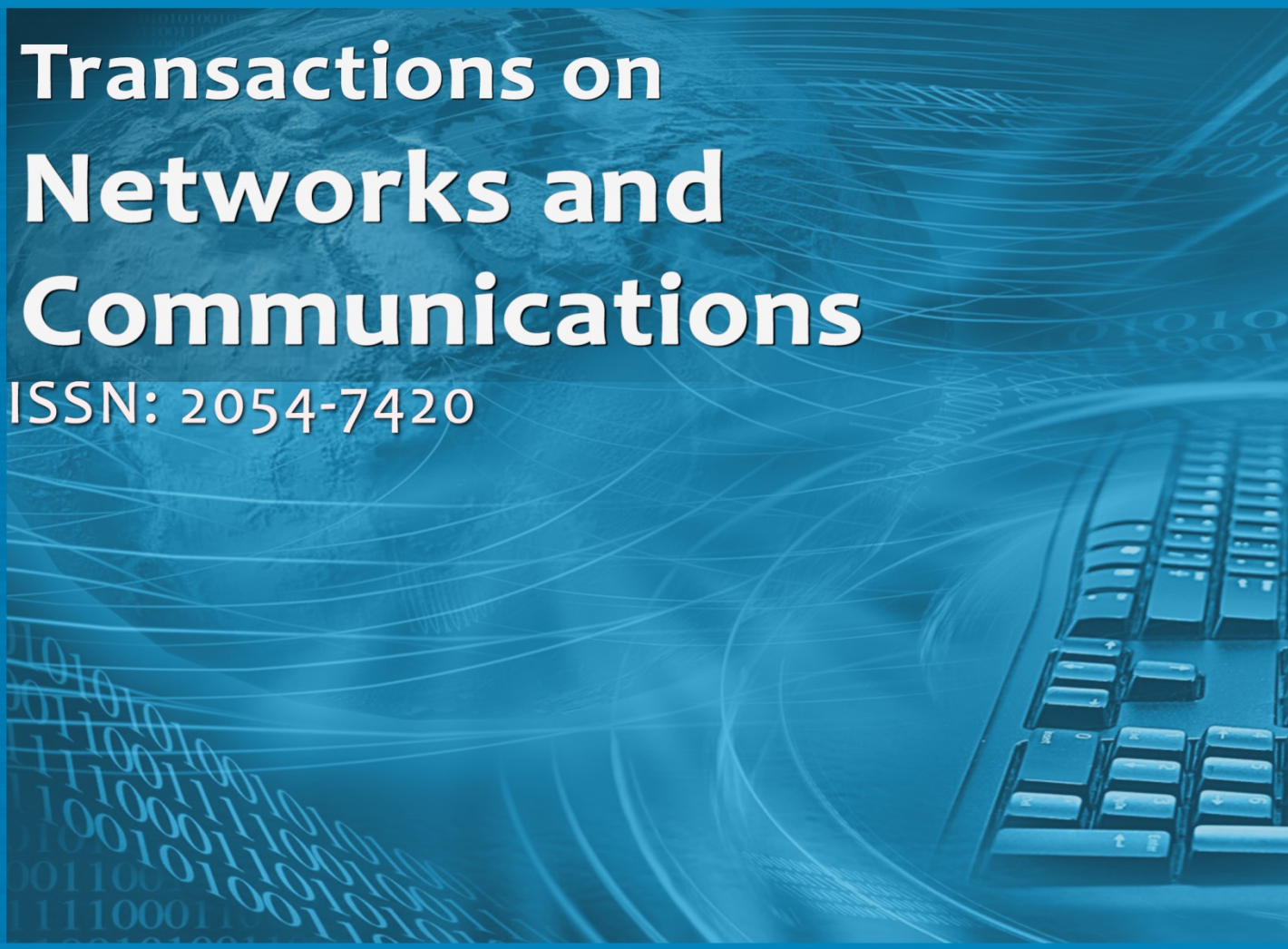
Volume 2 . Issue 2 - April 2014

Transactions on Networks and Communications

ISSN: 2054-7420



Society for Sciences and Education - United Kingdom

TABLE OF CONTENTS

EDITORIAL ADVISORY BOARD

DISCLAIMER

All the contributions are published in good faith and intentions to promote and encourage research activities around the globe. The contributions are property of their respective authors/owners and the journal is not responsible for any content that hurts someone's views or feelings etc.

TNC Transactions on Networks and Communications
SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
VOLUME 2, ISSUE 2
ISSN: 2054 - 7420

Enhancing the Capability of IDS using Fuzzy Rough Classifier with Genetic Search Feature Reduction

[1]Ashalata Panigrahi and [2]Manas Ranjan Patra
Department of Computer Science & Engineering, SMIT, Berhampur
India
[1]ashalata.panigrahi@yahoo.com; [2]mrpatra12@gmail.com

ABSTRACT

Rapid expansion of computer network throughout the world has made security a crucial issue in a computing environment. In the recent past several cyber-attacks have corrupted data of many organizations and creating serious problems. Intrusion Detection System which is increasingly a key part of system defense is used to identify abnormal activities in a computer system. The success of an intrusion detection system depends on the selection of the appropriate features in detecting the intrusion activity. Experiments have been conducted using four classifier techniques, viz, Fuzzy NN, Fuzzy Rough NN, VQNN, Fuzzy Rough Ownership NN. We have studied the accuracy, recall, precision, false alarm rate, error rate of all the classifier techniques

Keywords : Genetic search, Fuzzy rough theory, Hybrid Intrusion Detection System, False alarm rate.

1. INTRODUCTION

Network Intrusion Detection Systems are increasingly demand today due to the continuous increase in number of network attacks in networks. The primary aim of Intrusion Detection System (IDS) is to protect the availability, confidentiality and integrity of critical networked information system. IDS may perform either misuse detection or anomaly detection and may be deployed as either a network-based system or a host-based system. Misuse detection systems are most widely used and they detect intruders with known patterns as: network packet, like source address, destination address, source and destination ports or even some key words of the payload of a packet. Anomaly detection systems identify deviations from normal behavior and alert to potential unknown or novel attacks without having any prior knowledge of them. They have the ability of detecting unknown attacks [1]. The performance of the current intrusion detection system can be improved by utilizing hybrid intrusion detection

DOI: 10.14738/tnc.22.97
Publication Date: 3rd April 2014
URL: http://dx.doi.org/10.14738/tnc.22.97

techniques. The performance of hybrid classification techniques is better than distinct classification methods**.** Yang et al. [2] proposed a wrapper-based feature selection algorithm to find most important features from the training dataset by using random mutation hill climbing method, and then employs linear support vector machines (SVM) to evaluate the selected subset-features. M.Govindarajan et al. [3] have proposed the hybrid architecture. It has proved that, the performance is better for distinct classification methods**.** Horng et al. [4] have proposed an IDS that combines a hierarchical clustering algorithm, a simple feature selection procedure, and the SVM technique. At first, the hierarchical clustering algorithm is used to generate training instances. Then, the simple feature selection procedure was applied to eliminate unimportant features from the training set. Finally, the obtained SVM model classifies the network traffic data. Tsang et al. [5] in their work defines attribute reduction with fuzzy rough sets and analyzes its structures in details and they have developed a formal definition of reduction with fuzzy rough sets.

2. PROPOSED HYBRID MODEL

Hybrid classifiers enhance the accuracy of classification. The objective of hybrid classifier is to merge few machine learning techniques. All the attributes may not contribute in the analysis process for identifying intrusive behavior. In this work we have used genetic search as feature reduction method. After attribute reduction the data are classified by four classification techniques namely, Fuzzy NN, Fuzzy Rough NN, Fuzzy Rough Ownership NN, and Vaguely Quantified NN. The performance of these classifiers is evaluated in terms of their detection accuracy, precision, recall, fitness value, false alarm rate, error rate.

3. METHODOLOGY

3.1 Hybridization of Rough Sets and Fuzzy sets

Fuzzy Set

A fuzzy set in X is an X → [0, 1] mapping, while a fuzzy relation in X is a fuzzy set in X × X. For all y in X, the R-forest of y is the fuzzy set R_y is defined by

$$R_y(x) = R(x,y) \tag{1}$$

For all x in X, if R is reflexive and symmetric fuzzy relation, that is

$$R(x,x) = 1 \tag{2}$$

$$R(x,y) = R(y,x) \tag{3}$$

holds for all x and y in X, then R is called a "fuzzy tolerance ratio.

Rough Set

Rough Set Theory is a mathematical tool to deal with imprecise and insufficient knowledge. In rough set theory, membership is not the primary concept unlike fuzzy sets. It deals with

inconsistency, uncertainty, and incompleteness by imposing an upper and a lower approximation to set membership.

Let (X, A) be an information system where X is the universe of discourse and A is a non-empty finite set of attributes such that a : X → V_a for every a ∈ A. The set V_a is called the "value set of a". Given B ⊆ A there is an associated equivalence relation R_B:

$$R_B = \{ (x,y) \in X^2 \mid \forall a \in B, a(x) = a(y) \} \quad (4)$$

If (x,y) ∈ R_B, then x and y are indiscernible by attributes from B. The equivalence classes of the B- indiscernibility relation are denoted by $[x]_B$.

Let A be a subset X. A can be approximated using the information contained within B by constructing the B-lower and B-upper approximations of A.

$$R_B{\downarrow}A = \{ x \in X \mid [x]_B \text{ subset } A \} \quad (5)$$

$$R_B{\uparrow}A = \{ x \in X \mid [x]_B \cap A \neq \emptyset \} \quad (6)$$

The tuple ($R_B{\downarrow}A$, $R_B{\uparrow}A$) is called a rough set.

3.2 Fuzzy Nearest Neighbor (FNN) Classification

The Fuzzy K-Nearest Neighbor (FNN) algorithm [6] was introduced to classify test objects based on their similarity to a given number K of neighbors, and these neighbors' membership degree to (crisp or fuzzy) class labels. For the purpose of (FNN), the extent C(y) to which an unclassified object y belongs to a class C is computed as:

$$C(y) = \sum_{x \in N} R(x,y)\, C(x) \quad (7)$$

where N is the set of object y's K nearest neighbors, and R(x,y) is the [0,1]-valued similarity of x and y.

```
FNN ( X, C, y, K )
   X, the training data set;  C,  the set of decision classes;
   y, the objects to be classified;
   K, the number of nearest neighbors.
   Begin
     N ← get Nearest Neighbors ( y, K )
     For each  C ∈  C  do
        C' ( y)  =  ∑x∈N R(x,y) C(x) C(x)
        Output  arg max ( C' (y) )
   End
```

Figure 1: The Fuzzy K-Nearest Neighbor (FNN) Algorithm

3.3 Fuzzy-Rough nearest Neighbor Algorithm (FRNN)

In FRNN algorithm the nearest neighbors are used to construct the fuzzy lower and upper approximations of decision classes, and test instances are classified based on their membership to these approximations. FRNN algorithm combines fuzzy-rough approximation with the

classical FNN approach [7]. The rationale behind the algorithm is that the lower and upper approximation of a decision class, calculated by means of the nearest neighbors of a test object y, provides good clues to predict the membership of the test object to that class. The algorithm is dependent on the choice of a fuzzy tolerance relation R. Given the set of conditional attributes A, the fuzzy tolerance relation R is defined by

$$R(x,y) = \min_{a \in A} R_a(x,y) \quad (8)$$

in which $R_a(x,y)$ is the degree to which objects x and y are similar for attribute a. Here we choose

$$R_a(x,y) = 1 - \frac{|a(x) - a(y)|}{|amax - amin|} \quad (8)$$

If (R↓C) (y) is high , it reflects that all of y's neighbours belong to C. A high value of (R↑C) means that at least one neighbor belongs to that class.

```
FRNN ( X, C, y )
    X, the training data set;  C, the set of decision classes;
    y, the objects to be classified;
Begin
      N ← get Nearest Neighbors ( y, K )
       τ←   0   ,  Class ←  Ø
       for each  C ∈  C  do
              if (( R↓C )(y) +   ( R↑C)(y)) / 2  ≥ τ then
              τ  ←   (( R↓C )(y) +   ( R↑C)(y)) / 2
              end
         end
  output Class
End
```

The Fuzzy Rough nearest Neighbor Algorithm

3.4 Vaguely Quantified Nearest Neighbors (VQNN)

VQNN depends only on the summation of the similarities of each class. It uses the linguistic quantifiers "most" and "some". Given a couple (Q_u , Q_l) of fuzzy quantifiers that represent "most" and "some" respectively, the lower and upper approximation of C. VQNN assigns a class to a target instance y as follows:

Determine NN , the K nearest neighbors of y.

Assign y to the class C for which ($R\downarrow^{Qu} C$)(y) + ($R\uparrow^{Ql} C$)(y) is maximal.

The upper and lower approximation of Vaguely Quantified rough sets are defined as

$$((R\downarrow^{Qu} C)(y)) = Q_u(\frac{\sum_{x \in X} \min(Rx,y),C(x)}{\sum_{x \in X} R(x,y)}) \quad (9)$$

$$((R\uparrow^{Ql} C)(y)) = Q_l(\frac{\sum_{x \in X} \min(Rx,y),C(x)}{\sum_{x \in X} R(x,y)}) \quad (10)$$

The fuzzy quantifiers Q_u , Q_l are increasing [0,1] → [0,1] mapping such that Q_u (1) = Q_l(1)=1 and Q_u (0) = Q_l(0)=0. This classifier based on rough set theory is capable of handling noise data.

3.5 Fuzzy Rough Ownership Algorithm

A fuzzy-Rough ownership is an attempt to handle both "fuzzy uncertainty" and "rough uncertainty". The fuzzy-rough ownership function τ_c of class C defined for an object y as,

$$\tau_c(y) = \sum_{x \in X} \frac{R(x,y)C(x)}{|X|} \quad (11)$$

The fuzzy relation R is determined by :

$$R(x,y) = \exp(-\sum_{a \in A} Ka(a(y) - a(x))2/(m - 1) \quad (12)$$

where m controls the weighting of the similarity and K_a is a parameter that decides the bandwidth of the membership. K_a is defined as

$$K_a = \frac{|X|}{2 \sum_{x \in X} || a(y) - a(x) ||2/(m-1)} \quad (13)$$

$\tau_c(y)$ is interpreted as the confidence with which y can be classified to class C. The algorithm does not use fuzzy lower or upper approximations to determine class membership.

```
FROWNN( X, A, C, y )
 X the training data set; A the set of conditional features;
 C the set of decision classes; y the object to be classified.
begin
      for each a ∈ A do
            Ka = |X| / (2 Σ_{x∈X} || a(y) − a(x) ||2/(m−1))
      end
N ← | x |
for each A ∈ C do τc(y) = 0
for each x ∈ N do
            d = Σ_{a∈A} Ka(a(y) − a(x))^2
     for each A ∈ C do
       τc(y) + = C(x).exp(− d^{1/(m - 1)} ) / | N |
     end
end
output    arg max τc(y)
              A ∈ C
```

The Fuzzy Rough Ownership Algorithm

4. EXPERIMENTAL SETUP

4.1 NSL-KDD Dataset

NSL- KDD is a dataset proposed by Tavallace et al. [8]. NSL-KDD data set is a reduced version of the original KDD 99 dataset. NSL-KDD consists of same features as KDD 99. The data set consists of 41 feature attributes out of which 38 are numeric and 3 are symbolic. Total number of records in the data set is 125973 out of which 67343 are normal and 58630 are attacks. The dataset contains different attack types that could be classified into four main categories namely, Denial of Service (DOS), Remote to Local (R2L) ,User to Root (U2R), and Probing.

Denial of Service (DOS) : A DOS attack is a type of attack in which an attacker overwhelms the victim host with a huge number of requests, example ping-of-death, smurf, etc.

Remote to Local Attacks (R2L) : A remote to local attack is an attack in which the intruder tries to exploit the system vulnerabilities in order to control the remote machine through the network as a local user, for example guessing password etc.

User to Root Attacks (U2R) : These attacks are exploitations in which the attacker starts off on the system with a normal user account and attempt to abuse vulnerabilities in the system in order to gain super user privileges, for example phf, etc.

Probing : Probing is an attack in which the attacker scans a machine or a networking device in order to determine weakness or vulnerabilities that may later be exploited so as to compromise the system, for example port-scan.

The percentage distribution of data under different categories is depicted in table I and Figure 2.

Table: 1 Data Distribution & Percentage of NSL-KDD Dataset

Class	Number of Records	% of occurrence
Normal	67343	53.48%
DOS	45927	36.45%
R2L	995	0.78%
Probes	11656	9.25%
U2R	52	0.04%
Total	125973	100%

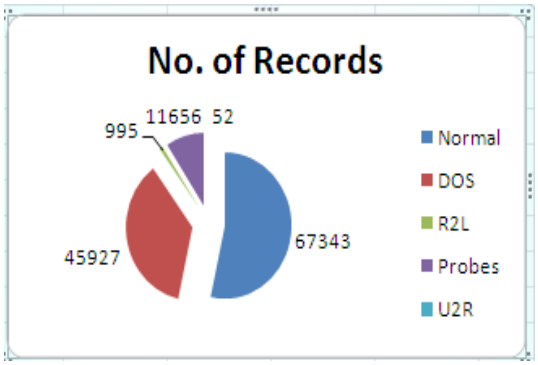


Figure 1: Distribution of Records

4.2 Feature Selection

Redundant and irrelevant attributes of dataset may lead to reduce detection accuracy. Effective input attributes selection from intrusion detection dataset is one of the important challenge for constructing high performance intrusion detection system. Feature selection is the process of finding a subset of features from the original dataset. The basic objective of feature selection method is to remove noise attributes and find important attributes which can represent data as a whole can improve the performance of intrusion detection and decrease the computation time. In this study Genetic search technique applied as feature selection method.

4.3 Genetic Search

Genetic algorithm is used as a search method. GA is based on principles of evolution and natural selection. Genetic search performs a global search. A Genetic algorithm mainly composed of three operators: reproduction, crossover, and mutation. Reproduction selects good string; crossover combines good strings to try to generate better offspring's; mutation alters a string locally to attempt to create a better string. In each generation, the population is evaluated and tested for termination of the algorithm. If the termination criterion is not satisfied, the population is operated upon by the three genetic algorithm operators and then re-evaluated. This procedure is continued until the termination criterion is met. [9]

Table 2: List of Selected Features

Feature Selection Method	No. of features selected	Feature Names
Genetic Search	16	Service, Flag, Src_bytes. Dst_bytes, Land, Urgent,Logged_in, Srv_count, Serror_rate, Srv_serror_rate, Rerror_rate, Same_srv_rate, Diff_srv_rate, Dst_host_count, Dst_host_same_srv_rate, Dst_host_srv_serror_rate.

4.4 Cross-Validation

The 10-fold cross-validation method is used to estimate the performance of different techniques. The entire dataset is divided into two different subsets, namely, training set and testing set. The training set is used to perform the analysis and the test set is used to validate

the analysis. In 10-Fold cross validation given dataset is partitioned into 10 subsets. From these 10 subsets 9 subsets are used to perform a training fold and a single subset is used as the testing data. The process is repeated 10 times such that each subset is used as a test subset once. The estimated accuracy is then the mean of the estimates for each of the classifiers.

4.5 Evaluation Measurement

The performance of IDS is measured and evaluated by the value of precision, accuracy, recall, false alarm rate.

TP (True Positive): The number of malicious records that are correctly identified.

TN (True Negative): The number of legitimate (not attacks) records that are correctly classified.

FP (False Positive): The number of records that are incorrectly identified as attacks though they are actually the legitimate ones.

FN (False Negative): The number of records that are incorrectly classified as legitimate activities though those are actually malicious.

Accuracy measure the probability that the algorithm can correctly predict positive and negative examples which are given by:

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \tag{14}$$

Precision is a measure of the accuracy provided that a specific class has been predicted which is given by:

$$\text{Precision} = \frac{TP}{TP+FP} \tag{15}$$

Recall measure the probability that the algorithm can correctly predict positive examples which is given by:

$$\text{Recall} = \frac{TP}{TP+FN} \tag{16}$$

False Alarm Rate is defined as the number of normal instances incorrectly labeled as intrusion divided by the total number of normal instances which is given by:

$$\text{False Alarm Rate} = \frac{FP}{FP+TN} \tag{17}$$

F- Value is the harmonic mean of Precision and Recall which measure the quality of classification which is given by

$$\text{F - Value} = 2 * \frac{(Precision * Recall)}{(Precision + Recall)} \tag{18}$$

$$\text{Fitness Value} = \frac{TP}{TP+FP} * \frac{TN}{TN+FP} \tag{19}$$

$$\text{Error Rate} = 1 - \text{Accuracy} \tag{20}$$

Kappa Statistic: Kappa is a chance-corrected measure of agreement between the classification and true classes. Kappa statistic is used to access the accuracy of any particular measuring case. It is used to distinguish between the reliability of the data collected and their validity [10]. The value of kappa is less than or equal to 1. The value of 1 indicates perfect agreement.

Mean Absolute Error (MAE): The Mean Absolute Error measures the average magnitude of the errors.

Root Mean Squared Error (RMSE): Root Mean Squared Error is a quadratic scoring rule which measures the average magnitude of the errors. Measures the difference between forecast and corresponding observed values, are each squared and then averaged over the sample. Finally, the square root of the average is taken.

5. RESULTS AND DISCUSSIONS

Genetic search technique selected 16 attributes from 41 attributes from the data set. Four classification techniques namely, Fuzzy NN, Fuzzy Rough NN, VQNN, Fuzzy Ownership-NN were used for comparison. All the classification techniques are tested using 10-fod cross-validation. Table 3 depict the performance of four classifier techniques in terms of correctly classified instances and incorrectly classified instances. Fuzzy ownership NN technique identifies highest number of correctly classified instances and less number of incorrectly classified instances. Table 3 shows Fuzzy NN has highest mean absolute error rate.

Table : 3 Comparison of Different Parameters

Feature Reduction Method	Test Mode	Classifier Techniques	Correctly Classified Instances	Incorrectly Classified Instances	Kappa Statistic	MAE	RMSE
Genetic Search	10-Cross Validation	Fuzzy NN	94.606%	5.394%	0.9074	0.0216%	0.1469%
		Fuzzy Rough NN	99.0236%	0.9764%	0.9829	0.0109%	0.0768%
		VQNN	98.772%	1.228%	0.9785	0.0054%	0.0649%
		FROWNN	99.4173%	0.5565%	0.9903	0.0028%	0.0413%

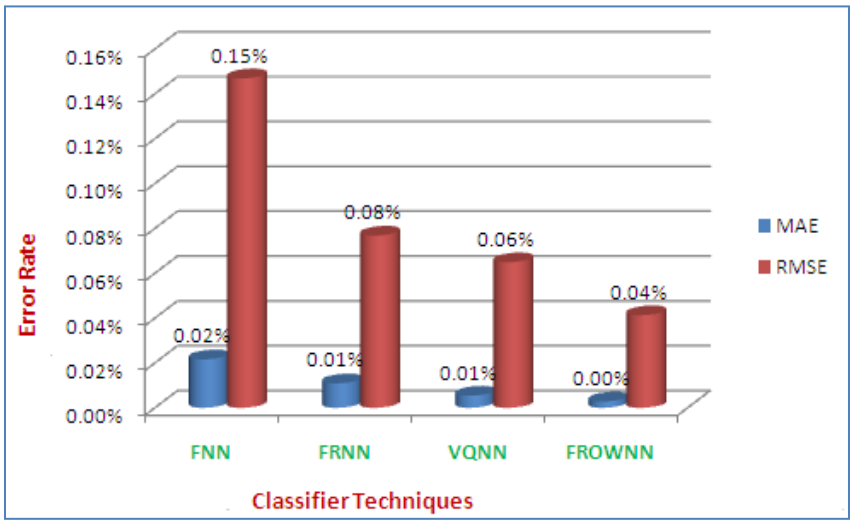


Figure 2: Mean Absolute Error vs Root Mean Squared Error

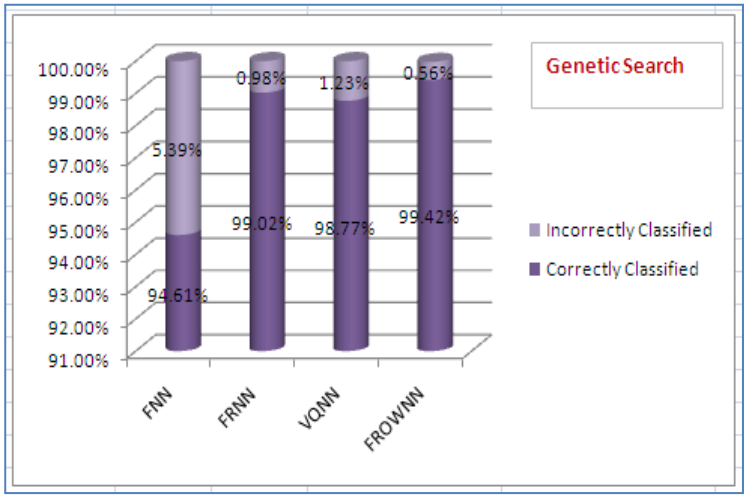


Figure 3 Correctly vs. Incorrectly Classified Instances

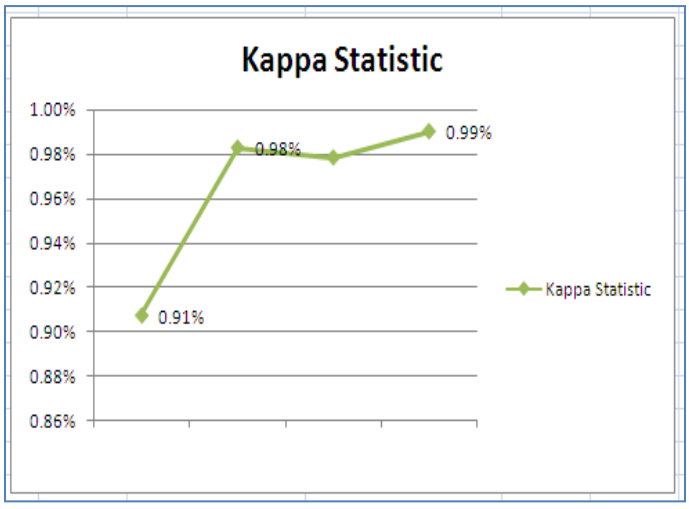


Figure 4: Comparison of Kappa Statistic of Four Classifier Techniques

Table : 4 Comparison of Accuracy, Precision , Recall, F-Value of Four Classifier Techniques

Feature Reduction Method	Test Mode	Classifier Techniques	Accuracy	Precision	Recall	F-Value
Genetic Search	10-Fold Cross-Validation	FNN	98.0845%	97.5875%	98.3149%	97.9499%
		VQNN	98.9069%	99.3382%	98.3063%	98.8198%
		FRNN	99.0736%	99.2458%	98.76002%	99.00234%
		FWNN	**99.5197%**	**99.5054%**	**99.5054%**	**99.5053%**

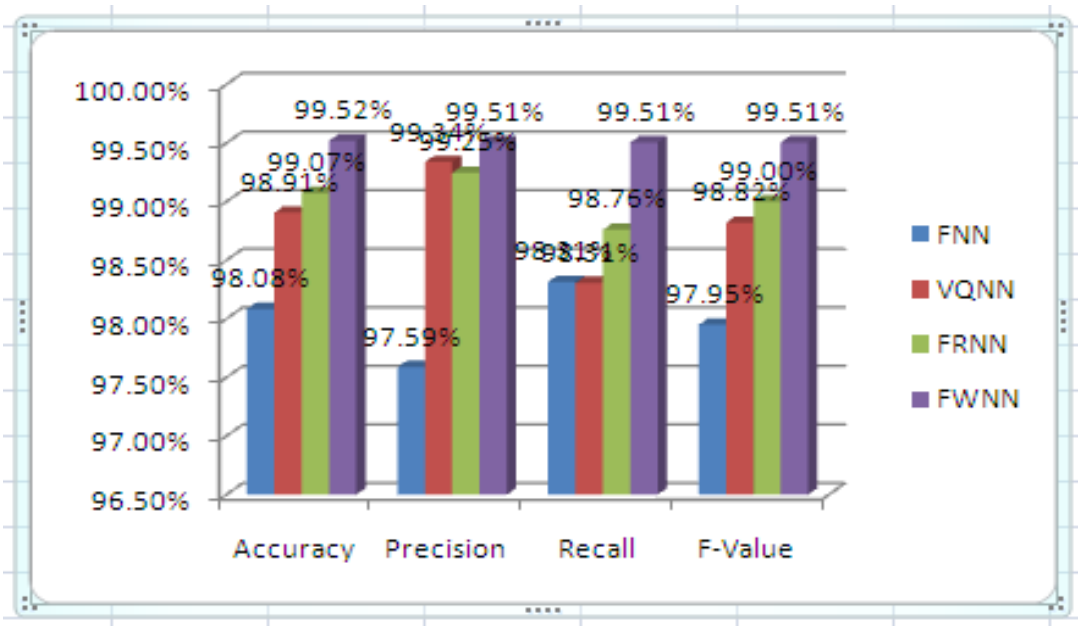


Figure 5 : Comparison of Accuracy, Precision , Recall, F-value

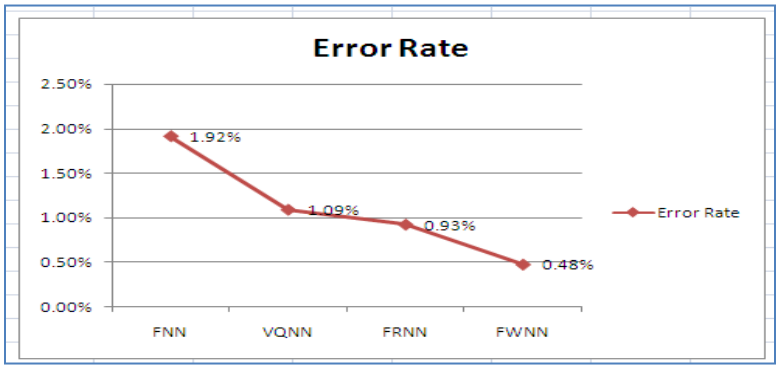


Fig. 6 Error Rate of Four classifier Techniques

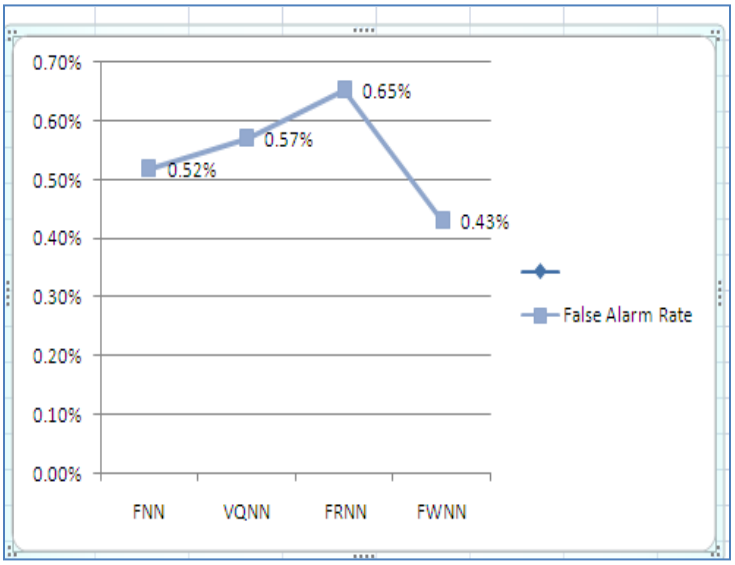


Fig. 7 Comparison of False alarm Rate of classifier techniques

6. CONCLUSION

Intrusion detection in large networks has been a challenging task. In this paper we have proposed hybrid model for intrusion detection. Experiments in our present work show that Fuzzy Ownership NN classifier technique provides best performance compared to other three classifier techniques. In future, we will explore other hybrid techniques with different feature selection methods and study their performance using different parameters.

REFERENCES

[1]. Zorana Bankovic, Dus an Stepanovic, Slobodan Bojanic, Octavio Nieto-Taladriz, " Improving network security using genetic algorithm approach" . Computers and Electrical Engineering, pp. 438-451, 2007.

[2]. Yang Li, J. L. Yang, Z. H. Tian, T. B. Lu, and C. Young, " Building lightweight intrusion detection system using wrapper-based feature selection mechanisms", Computer and Security, Vol. 28, pp. 466-475, September 2009.

[3]. M.Govindarjan , and R.M. Chandrasekran , " Intrusion Detection Using Neural Based Hybrid Classification Methods", Computer networks . 55(8): 1662-1671, 2011.

[4]. S.Horng, M.Su, Y.Chen, T.Kao, R.Chen, J.Lai and C.D.Perkasa, " A Novel Intrusion Detection System Based on Hierarchical Clustering and Support Vector Machines" , Expert Systems with Applications, vol.38, no.1, pp.306-313, 2011

[5]. C. C. Tsang, Degang Chen, and D. S. Yeung. Attribute Reduction using Fuzzy Rough Sets,. In IEEE Transaction on Fuzzy Systems, vol. 16, pp. 1130-1140, oct. 2008.

[6]. J.M. Keller , M. R. Gray, J. A. Givens : A Fuzzy K-Nearest Neighbour Algorithm, IEEE Trans. Systems Man Cybernet. 15(4), pp.580-585, 1985

[7]. Jesen,R. and Cornelis,C. “A new approach to fuzzy-rough nearest neoghbour classification”, LNAI 5306, Springer-Verlag, pp. 310-319 (2008).

[8]. M. Tavallaee, E Bagheri; Wei Lu; and A. Ghorbani, A detailed analysis of the KDD CUP 99 data set. Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2009), 1-6 (2009)

[9]. D. Goldberg, Genetic Algorithm in Search, Optimization, and Machine Learning, Addison Wesley. 1989.

[10]. kappa at http://www.dmi.columbia.edu/homepages/chuangi/kappa

TNC Transactions on Networks and Communications
SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
VOLUME 2, ISSUE 2
ISSN: 2054 - 7420

The effect of properties of human body on statistical characteristics channel on-body communication

Esraa H.Kadum[1] and Haider M. AlSabbagh[2]
[1]esraahabeeb63@yahoo.com ;[2]haidermaw@ieee.org,
[1,2]*Department of Electrical Engineering, College of Engineering, University of Basra, Basra, Iraq*

ABSTRACT

In wireless body area networks , on-body radio propagation channels are typically time-varying, because of the frequent body movements , the one case of human body state is statically case when the body is stand up with its hand along the body , the model taking into account the places of nodes as its shown in this paper, the channel characteristics for statistical one case which may birthing can effect on the design ,four channel model line of sight and one sensor (antenna) put on back which make the channel as non – light of sight .the destine between transmitter and receiver make a very important rule because the path loss dependent on it directly , the system design is many transmitter and one receiver (Master and Slave) node, single input multi output channel design (SIMO).

Keywords: on-body communication, channel model, statistical channel, path loss and power delay profile.

1. INTRODUCTION

For a Body Area Network (BAN) channel model, it is required to determine the electromagnetic field at each point on the body, for a given position of the transmitter on the body, this is a big problem numerically, Therefore, it is desirable to drive an analytical expression which performs this objective resolving this problem means solving the Maxwell's equations for each point of the body [1]. Wireless Body Area Networks (WBAN) consist of a number of units placed inside or in proximity of the human body (such as in everyday clothing), and are a natural progression of the WPAN concept [2]. Also known as IEEE 802.15.6, WBAN is a low-frequency technology intended to endow a future generation of short-range electronics for exchanging information [3] [4]. This paper takes in hand the study one of the body communication (on–body), the channel characterization in the statistical case of human body ,this is will do for standing where two hands along the body , the Characterization of the Radio Channel in On-Body Communication[5].

DOI: 10.14738/tnc.22.83
Publication Date: 3rd April 2014
URL: http://dx.doi.org/10.14738/tnc.22.83

The study of statistical channel characteristics like power delay profile, RMS delay spreads, coherence bandwidth and time coherence are performed. Many UWB components and systems are already in the testing and demonstration phases, with actual release dates for final consumer products expected in early 2005. Intel Corporation is working with the industry to enable this exciting technology and help ensure its success. On-body area ultra wideband (UWB) communication is of high importance for promising new biomedical applications [6,7].

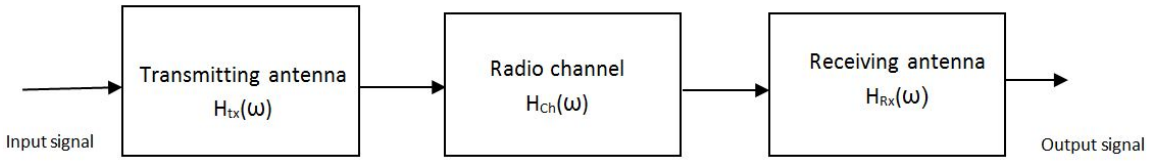


Fig (1) Block diagram representing the radio channel

By taking a single-input, single-output system Fig (1), in which an input signal x(t) is introduced, causing an output signal y(t) to come out. Common assumptions for the system are to be linear and time-invariant, but the ability to release these constraints in the following. Inside the system, some noise could be generated, and added to the "deterministic" part of the output signal. Usually this noise is assumed to be white Gaussian noise, completely uncorrelated with the input signal [8].

A convenient characterization of multi-path propagation , In this model, the time axis is divided into small time intervals called "bins". The received power is integrated within each bin to obtain the energy received as function of the excess delay [9,10]. The statistics of the Saleh-Valenzuela model and using the measured Bicone NLOS channel parameters for different case. Specifically the examine of the cumulative distribution function CDF of the mean excess delay, RMS delay spread and the number of paths for the artificial channel [11].

The power delay profile gives information about the delay which a transmitted signal experiences in multipath internment, the PDP is often empirically mo deled by an exponentially decaying [12]. The calculation of the PDP starts from a ray optical point of view. In fig (2), if the positions R_x of the receiver and T_x of the transmitter are given, the locations of the radiating virtual sources can be calculated using the method of images each virtual source is supposed to transmit at zero delay a Dirac pulse at a carrier frequency f_c ; the pulse needs a delay time until it reaches the receiver [13].

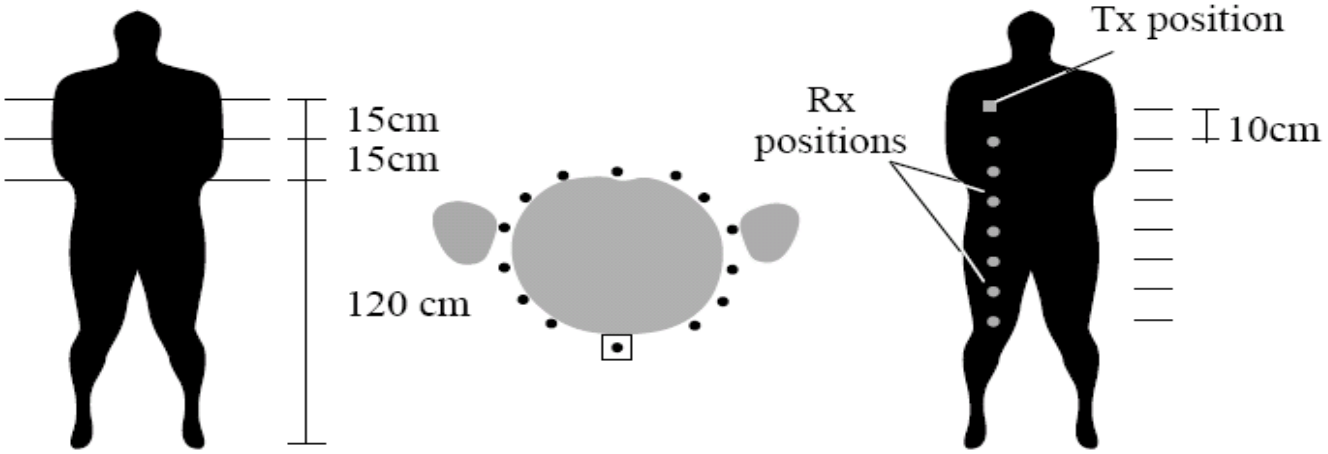


Fig (2) path loss model for standing case

For .. R_x = directional antenna , T_x = Omni-directional antenna ,the fig (3) shows the principle work of line of sight channel model on body taking account the other obstacles [14].

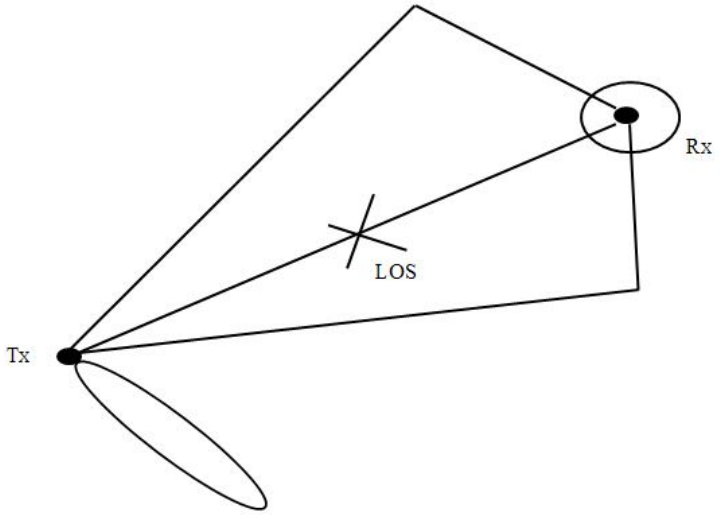


Fig (3) The principle line of sight on body channel (LOS)

2. EXPERIMENTATION

2.1 Multipath Channel Model

In wireless communications, the multipath propagation phenomenon is when the transmitted signal travels along many different paths to reach the receiver. The presence of multiple paths between transmitter and receiver introduces complexity in the channel modeling [15]. The complexity depends on the distribution of the multipath intensity, relative propagation time of the waves and the bandwidth of transmitted signal. Therefore the time – varying properties of the channel must be taken into account in the channel model. The

presented of a body area UWB multipath channel model based on the classical Saleh - Valenzuela model[16] [17] .

The impulse response model is a convenient model to characterize the multipath channel and any deterministic impulse response can be represented by discrete tapped delay line model as long as the system is band limited.

The Saleh-Valenzuela model can be represented as:

$$h(t)_i = X \sum_{l=0}^{L-1} \sum_{k=0}^{k-1} \alpha_{K,1}^{i}\, \delta\left(t - T_1^i - \tau_{k,1}^i\right)$$

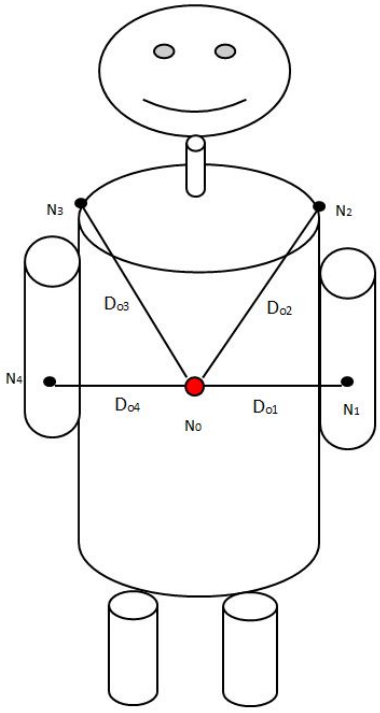
Where $\alpha_{k,1}^{i}$ represents the multipath gain coefficients, $T_{k,l}^{i}$ represents the delay of the $l^{th}\ path$, $\tau_{k,l}^{i}$ represents the delay k^{th} multipath component relative to the l^{th} path arrival time (T_1^i). X_i represents the log normal shadowing , and (i) refers to the i^{th} realization.

For the S-V Channel Model the arrival time and the ray arrival time are Poisson processes:

$$p\left(\frac{T_l}{T_{l-1}}\right) = \Lambda \exp[-\Lambda(T_l - T_{l-1})] \ldots\ldots\ldots \text{ l } > 0$$

$$p\left(\tau_{k,l}/\tau_{(k-1),l}\right) = \lambda\, exp\left[-\lambda\left(\tau_{k,l} - \tau_{(k-1),l}\right)\right] \ldots\ldots \text{ k} > 0$$

So, the design is shown in fig. below:



Fig(4) Human body with positions of nodes (antennas) and the desistance between sensors

For different channel design which are : 4 channel are line- of- sight (LOS) for distance between transmitter and receiver (node) equal to D_1, D_2, D_3 , D_4 .and there is one node which is Non – line – of sight (NLOS) .

CM1: Line Of Sight (LOS) model for 0 - 30 cm

CM2 : Line Of Sight (LOS) model for 0 - 30 cm

CM3 : Line Of Sight (LOS) model for 0 - 70 cm

CM4 : Line Of Sight (LOS) model for 0 -70 cm

CM5 : Non Line of sight (NLOS) for 0 - 140 cm

For model with parameters $\theta = \{\theta_1 ,\theta_2 ,.....\theta_p\}$:

D: the distance between every node and the master node (N_o), p = No. of parameters which are used for optimization and the impulse response of the system.

In practice, the output signal can be written as the sum of the generated noise and a deterministic function of the input signal:

$$y(t) = n(t) + F[x(t)]$$

If the system is linear and time-invariant, the function F assumes the form of the convolution between the input signal and the system's impulse response h(t):

$$y(t) = n(t) + x(t) \otimes h(t)$$

By using the software MATLAB the impulse response for four different modified S-V models as shown in fig () .it is apparent that the paths in CM1 and CM2 tend to concentrate in two or three paths with smaller delay , whereas in CM3,CM4 have bigger delay because of the more distance .

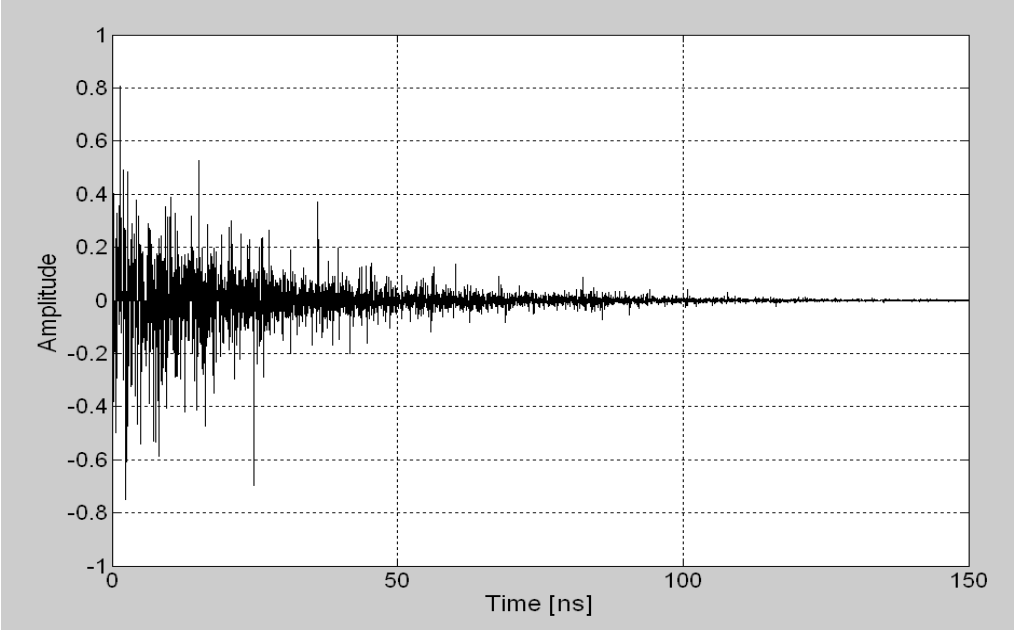


Fig (5) Delay for Saleh –Valenzuela channel model on-body communication CM1-4

For communication between two sensors on the human body, transmitted signals can arrive at the receiver in 3 ways:

- propagation through the body
- diffraction around the body
- Reflections off of nearby scatterers and then back toward the body.

2.2 Path loss model

This model represents the exponential decay with distance (expected with diffraction around a cylindrical body), followed by a flat saturation point due to energy received from multipath reflections off nearby scatterers.

$$P_{dB}(d) = -10\log_{10}\left(P_0\ e^{-m_0 d}\ + P_1\right) +\ \sigma_P n_P$$

The parameters are explained as:

- P_0 is the average loss close to the antenna. It will depend on the type of antenna.
- M_0 represents the average decay rate in dB/cm for the surface wave travelling around the perimeter of the body.
- P_1 is the average attenuation of components in an indoor environment radiated away from the body and reflected back towards the receiving antenna.
- σ_p is the log-normal variance in dB around the average representing the variations measured at different body and room locations. This parameter will depend on variations in the body curvature, tissue properties and antenna radiation properties at different body locations.

Wireless dynamic channels can be modeled as linear time- varying systems:

$$y(t) = \sum a_i(t)\, x(t - \tau_i(t))$$

where a_i (t) and τ_i (t) are the gain and delay of path i.

The time-varying impulse response:

$$h(t,\tau) = \sum a_i\,(t)\delta(\tau - \tau_i(t))$$

So wireless channels are characterized by severe fluctuations in the receive power, i.e., in the strength of the electromagnetic field at the receiver position. The receive power is usually modeled as a combination of three phenomena: path loss, large-scale fading, and small-scale fading.

The path loss describes the distance-dependent power decay of electromagnetic waves.

Let us model the attenuation factor as d, where d is the distance of the wave has traveled and denotes the path loss exponent, which is typically assumed to lie between 2 and 4. The path loss in decibels is then obtained as:

$$PL_{dB}(d) = PL_{0,dB} + 10n\log_{10}\frac{d}{d_0}$$

In this equation, the path loss exponent (n) is taking as 3.8 where the path loss reference distance (d_0) is 0.1m and finally the value the path loss in reference $PL_{0,dB} = 43$.

Path loss is the attenuation of an electromagnetic wave as it propagates through space, depending on the environment in which the radio channel is contained, different propagation models are used and consequent path loss will be obtained in a different way. These

models are mainly empirical with equations based in measurements within a certain frequency band. For urban environments, for example, several models can be used depending on the situation, such as Okumura-Hata Model. Some path loss measurements have already been conducted for on-body elements and some conclusions were taken. As quoted in , some studies have been developed at 2.45GHz with the body in different positions, propagation paths can be with LOS or NLOS, in other words, transmitter and receiver can "see" each other or do not, respectively.

3. CONSOLATION

The distance between antennas (sensors) which is part of channel model and the principles of channel which are line of sight or non-line of sight ,this is have direct effect on path loss either the channel was in free space or on body ,for channel (1) and channel (2) the delay is smaller than in channel (3) and (4) , the effect of human body make the path loss of statistical channel more less than when the channel in free space .The putting of the antennas on the two shoulder and on the two hands make the model more sufficient for this case (standing) ,when we change the places(legs, knee, waist,….) the model all change and also its statistical characteristics.

REFERENCES

[1]. A. D. Droitcour, O. Boric-Lubecke, V. M. Lubecke, J. Lin, and G. T. A. Kovacs, "Range correlation and I/Q performance benefits in single-chip silicon Doppler radars for noncontact cardiopulmonary monitoring," IEEE Trans. Microw. Theory Tech., vol. 52, no. 3, pp 838–848, Mar. 2004.

[2]. A. A. Serra, P. Nepa, and G. Manara, "On-body antenna input-impedance phase-modulation induced by breathing and heart activity," in Proc. URSI, Jul. 5–11, 2008.

[3]. R. F. Dubrovka and I. B. Shirokov, "On-body antenna for the miners cardiac rhythm sensor," in Proc. Antennas Propag. Conf., Loughborough, 2009, pp. 581–584.

[4]. P. Salonen, Y. Rahmat-Samii, H. Hurme, and M. Kivikoski, "Dual-Band Wearable Textile Antenna", 2004 IEEE AP-S Int. Symp. Dig., volume 1, pp. 463-466,2004.

[5]. N. Jin, and Y. Rahmat-Samii, "Parallel particle swarm optimization and finite-difference time-domain (PSO/ FDTD) algorithm for multiband and wide-band patch antenna designs" IEEE Trans. Antennas Propagat., volume 53, No. 11, pp. 3459-3468, (2005).

[6]. Cheng Tao, Li Jin. Analysis and simulation of RFID anti-collision algorithms .IEEE Proceeding Advanced system. Communi-cation Technology. Phoenix Park,Korea. 2007,pp:697-701 .

[7]. H. Sawada, T. Aoyagi, J. Takada, K. Y. Yazdandoost, and R. Kohno, "Channel model for wireless body area network," in Proc. 2nd Intl. Symp. on Med. Info. and Comm. Tech. (ISMICT), Oulu, Finland, Dec.2007.PP 243-248.

[8]. T. Zasowski, G. Meyer, F. Althaus, and A. Wittneben, "Propagation effects in uwb body area networks," in IEEE Intl. conf. on Ultra -wideband, Sept. 2005.

[9]. A. Fort, J. Ryckaert, C. Desset, P. D. Doncker, P. Wambacq, and L. V. Biesen, "Ultra-wideband channel model for communication around the human body," IEEE journal on selected areas in comm., vol. 24, pp. 927–933, 2006.

[10]. B. Zhen, M. Kim, J. Takada, and R. Kohno, "Characterization and modeling of dynamic on-body propagation at 4.5 GHz," IEEE Antennas Wireless Propagat. Lett., vol. 8, pp. 1263–1267, 2009.

[11]. J. Hagedorn, J. Terrill, W. Tang, K. Sayrafian, K. Y. Yazdandoost, and R. Kohno, "A statistical path loss model for MICS," IEEE 802.15-08- 0519-01-0006, Sept. 2008.

[12]. S. L. Cotton and W. G. Scanlon, "Characterization of the on-body channel in an outdoor environment at 2.45 ghz," in European conf. on antennas propagation, 2009, pp. 722–725.

[13]. M. Kim and J. Takada, "Statistical model for 4.5 GHz narrowband on- body propagaton channel with specific actions," IEEE Antennas Wireless Propagat. Lett., vol. 8, pp. 1250–1254, 2009.

[14]. A. Fort, J. Ryckaert, C. Desset, P. D. Doncker, P. Wambacq, and L. V. Biesen, "Ultra-wideband channel model for communication around the human body," IEEE journal on selected areas in comm., vol. 24, pp. 927–933, 2006.

[15]. R. Istepanian , E . Jovano v, Y. Zhang , "Guest Editorial Introduction to the Special Section on M-Health: Beyond Seamless Mobility and Global Wireless Health-Care Connectivity " in IEEE Transactions on Informati on Technology in Biomedicine, 8(4): 405 - 414 , Dec . 2004 .

[16]. D. Raskovic, T. Martin , E . Jovanov, "Medical Monitoring Applications for Wearable Computing," in The Computer Journal, July 2004, 47(4):pp. 495-504.

[17]. Mehmet. Yuce , Ho . Keon g, and Moo . Chae, "Wideband Communication for Implantable and Wearable Systems ", in IEEE TRANSACTIONS ON MI CROWAVE THEORY AND TECHNIQUES, VOL. 7 , NO10, OCTOBER 2009.

TNC Transactions on Networks and Communications
SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
VOLUME 2, ISSUE 2
ISSN: 2054 - 7420

Unified Transition to Cooperative Unmanned Systems under Spatial Grasp Paradigm

Peter Simon Sapaty
Institute of Mathematical Machines and Systems, National Academy of Sciences of Ukraine;
sapaty@immsp.kiev.ua, peter.sapaty@gmail.com

ABSTRACT

A novel philosophy, ideology, methodology, and supporting high-level networking technology will be revealed capable of guiding gradual transition to intelligent unmanned systems with a variety of important practical applications. The approach is based on a completely different type of high-level language capable of grasping top semantics of complex spatial operations in dynamic and unpredictable environments while shifting numerous technical details to effective automatic implementation. The language is based on holistic and gestalt principles rather than traditional multi-agent organizations, providing high integrity and super-summative features of the solutions described. Cooperative networked interpretation of the language in distributed systems and different parallel and distributed scenarios in it will be demonstrated that can be performed by any combination of manned and unmanned components under unified command and control provided by the technology described.

Keywords: Distributed Systems; Gestalt Philosophy; Spatial Grasp Language; Networked Interpretation; Spatial Scenarios; System Integrity; Unmanned Systems; Human-Robotic Integration.

1. INTRODUCTION

Unmanned systems are widely used today, and their role will definitely be increasing in the future. Their further development and engagement will, however, largely depend on how efficiently and naturally robotic means unite with manned systems within overall human activity. There should be clear philosophical, methodological, linguistic and technological grounds for manned-unmanned integration, division of jobs between humans and robots, and common command and control for combined missions.

Within this global context we are pursuing a *tasking approach* aimed at formalizing and describing problems to be solved and tasks to be executed in physical and virtual environments which may need robotic involvement. Effective tasks definitions may help to define which

DOI: 10.14738/tnc.22.83
Publication Date: 3rd April 2014
URL: http://dx.doi.org/10.14738/tnc.22.83

human and which robotic components should be used, and how they must be organized as a system to perform the tasks needed in most efficient way.

And this approach should allow for a general enough, semantic level of task presentations to allow maximum flexibility of their implementation with possibly unknown in advance and scarce resources, as well as for easiness of their redefinition when conditions, goals, and states of environment change. This will allow us to be in line with growing world dynamics and withstand numerous asymmetric situations and threats the humankind is facing, which may need asymmetric solutions too, and with broad engagement of advanced robotic means.

As we will be touching system organizations throughout this paper, let us provide some comments on and comparison of different system organizations.

The *traditional* approach to system design, development and management supposes the system structure and system organization to be predominantly primary, created in advance, whereas global function and overall behavior appearing as secondary, like in Figure 1.

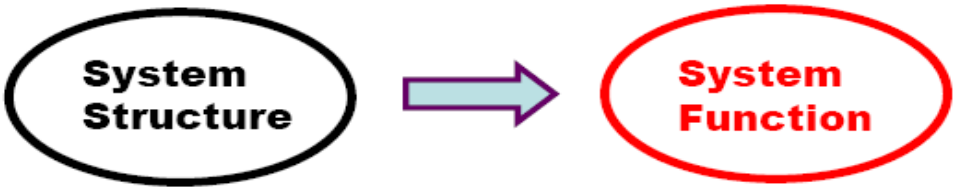


Figure 1: Traditional Approach to System Design

The systems based on this vision, ideologically relevant to multi-agent systems [1] still prevailing nowadays, are often clumsy and static, they may fail to quickly adapt to dynamic and asymmetric situations. If the initial goals change, the whole system may have to be partially or even completely redesigned and reassembled. Adjusting the already existing systems to new goals and functionality may result in a considerable loss of their integrity and performance.

With global goals changed, the whole projects based on creating structures and overall system organizations first may become not needed at all despite huge investments made into them like, for example, the famous robotized Future Combat Systems project, or FCS [2]. The latter, designed mainly for classical battlefields, became obsolete even in its infancy after the main operations changed towards terrorism fight, for which quite different system ideology and technical equipment appeared to be needed.

We are pursuing an *alternative* system approach, where the global function and overall behavior should be considered, as much as possible, primary, and the system structure and organization as secondary, the latter as a dynamic derivative of the former (see Figure 2).

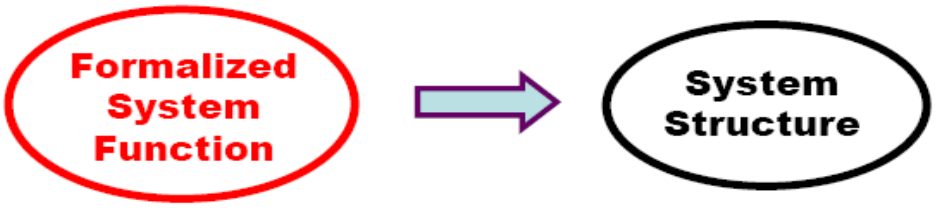


Figure 2: The Alternative System Organization Considered

The advantages of such (actually, the other way round) organization may include high potential flexibility of runtime system creation and management, especially in quick responses to asymmetric events. This quality allows us formulate top semantics of the needed reaction on world events in a special high level language, shifting most of traditional organizational routines to automated up to fully automatic implementation, with effective engagement of unmanned systems evolutionally and most naturally.

The related paradigm and accompanying networking technology we are developing is based on formalized wavelike seamless navigation, coverage, or grasping of distributed physical and virtual spaces, as symbolically shown in Figure 3. This believably inherits and psychologically matches of how human mind operates [3], especially in comprehension of distributed environments, in a holistic [4], gestalt-based [5-7], and integral way, and finds complex spatial solutions in them. These features are placed in our case on advanced highly parallel and fully distributed networking platforms often exhibiting clear advantages before intelligent biological systems in specific, especially distributed applications.

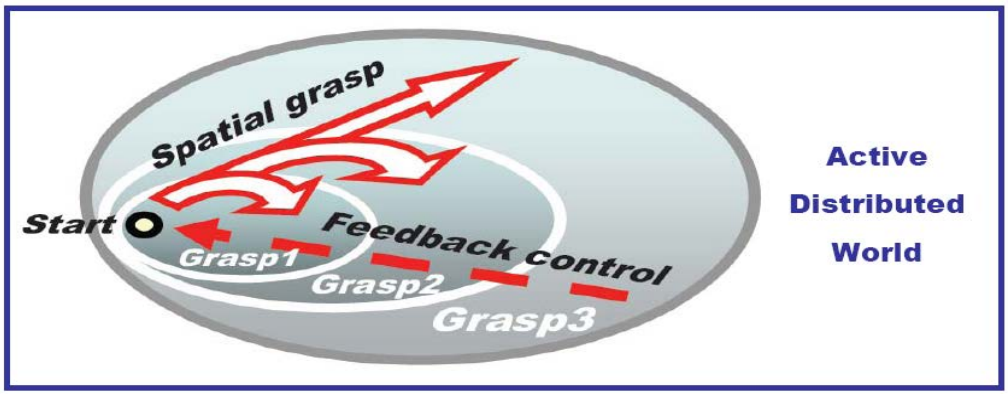


Figure 3: Incremental Spatial Grasp of Distributed Worlds

The approach in general works as follows. A network of universal control modules U, embedded into key system points (like humans, robots, smart sensors, mobile phones including), collectively interprets mission scenarios expressed in a special high-level Spatial Grasp Language (SGL), as shown in Figure 4. These scenarios, capable of representing any parallel and distributed algorithms, can start from any node while covering the whole system or its parts needed at runtime.

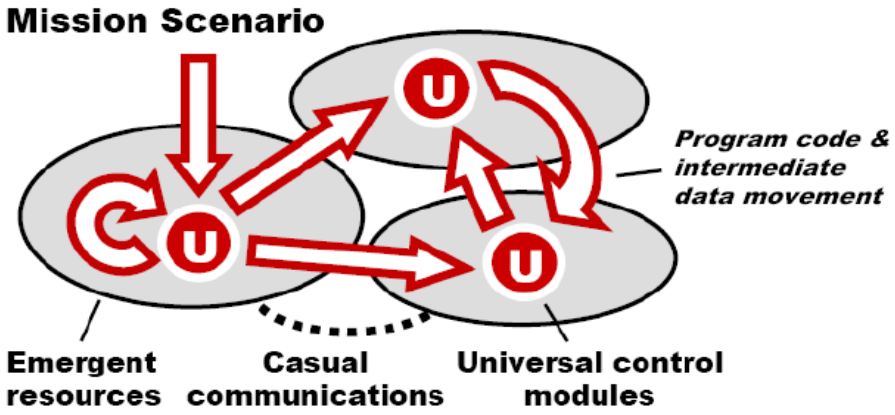


Figure 4: Collective Scenario Execution in Distributed Dynamic Environments

SGL scenarios, often expressing top semantics of spatial operations, are very compact and can be created on the fly. Different scenarios can cooperate or compete in a networked space as overlapping fields of solutions. Self-spreading scenarios can create runtime knowledge infrastructures distributed between system components, as shown in Figure 5. These can effectively support distributed databases, advanced command and control, global situation awareness, as well as any other computational or control models.

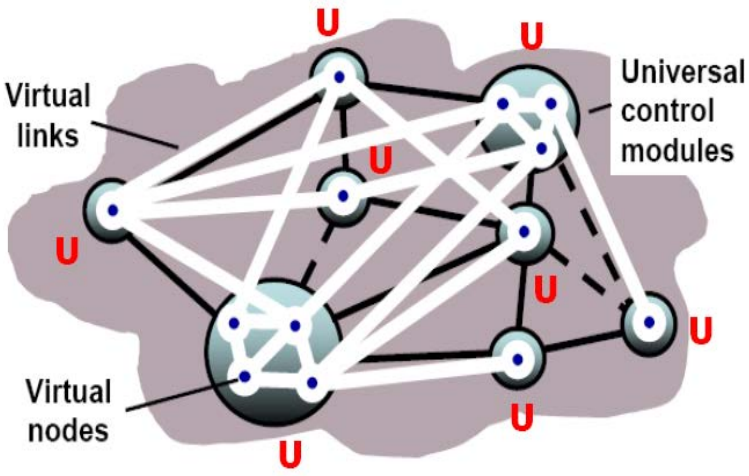


Figure 5: Creating Distributed Knowledge Infrastructures

This paper represents the first report on the latest version of SGL with extended repertoire of its functions (or rules). It also describes the related updated structure of the SGL interpreter as well as presents a number of cooperative robotic applications of SGT confirming simplicity and compactness of SGL-based scenarios (usually at least an order of magnitude more compact than in other known languages like Java for general data processing or BML [8] for command and control applications).

The development history, various philosophical and technological aspects of this Spatial Grasp Technology (SGT) as well as detailed descriptions of its researched areas can be found in many existing publications, including [10-20].

2. SPATIAL GRASP LANGUAGE

2.1 SGL Orientation and Peculiarities

SGL differs fundamentally from traditional programming languages. Rather than working with information in a computer memory, as usual, it allows us to directly *move through*, *observe*, and make *any actions and decisions* in fully distributed environments, whether physical or virtual. In general, the *whole distributed world*, which may be dynamic and active, is considered in SGL as a substitute to traditional computer memory. An SGL program (rather: *scenario*, to highlight its generality and orientation on direct problem solving in real worlds by both manned and unmanned components) can be viewed from different angles:

- As the first linguistic means to describe and formalize the notion of *gestalt* [5-7] allowing us to effectively grasp top semantics, integrity, and super-summative features of large complex systems.
- As an active recursive *self-matching pattern* applied against distributed physical, virtual, executive, or combined worlds, ruling and changing these worlds appropriately.
- As a universal *genetic* mechanism, expressed in a special integral formalism, allowing any distributed systems, whether passive or active, to be created, grown, evolved, and modified while starting from any world point. This also relates to the relatively new concept called *memetics* [9].
- As a sort of a *"soul"* (very symbolically, however) to be injected into and spread in a controlled manner throughout the distributed world, giving it new life, breath, and consciousness providing the needed both local and global awareness and control.

2.2 The SGL Worlds

SGL directly operates with:

- *Virtual World* (VW), which is finite and discrete, consisting of nodes and semantic links between them, both nodes and links capable of containing any information, of any nature and volume.
- *Physical World* (PW), infinite and continuous, where each point can be identified and accessed by physical coordinates expressed in a proper coordinate system, and with the precision given.
- *Execution world* (EW), consisting of active doers with communication channels between them, where doers may represent humans, robots, laptops, smartphones, any other devices or machinery capable of operating on the previous three worlds.

Different combinations of these worlds can also be possible, for example, *Virtual-Physical World* (VPW) allowing not only for a mixture of the both worlds but also their deeper integration where VW nodes can be associated with certain PW coordinates, thus making their presence in physical reality too. Another possibility is *Virtual-Execution World* (VEW), where doer nodes may be associated with virtual nodes like having special names (or nicknames) assigned to them, having now semantic relations between them too, like between pure VW nodes. *Execution-Physical World* (EPW) can pin some or all doer nodes permanently to certain PW coordinates, and *Virtual-Execution-Physical World* (VEPW) can combine features of the previous two variants.

2.3 Top SGL Syntax

SGL has recursive structure top level of which is shown in Figure 6. Such organization allows it to express any spatial algorithm, create and manage any distributed structures with any topologies, static as well as dynamic, also solve any problem in, on, and over them—and all this can be expressed in a very compact, transparent, and unified way.

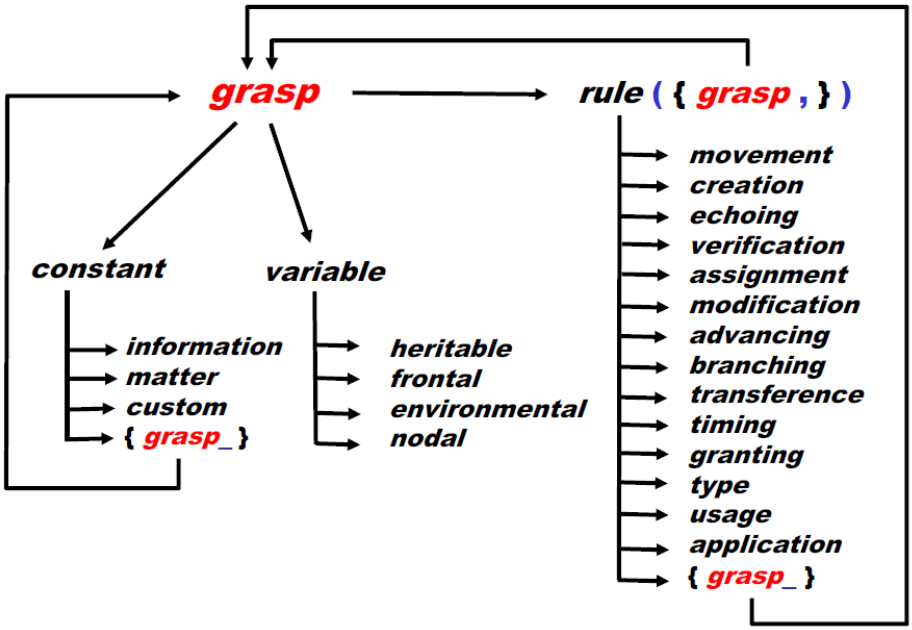


Figure 6: SGL Recursive Syntax

Let us explain the language basics in a stepwise top-down manner.

The SGL topmost definition, where scenario in it is named as *grasp* (reflecting the spatial navigation-grasp-conquest model explained in previous chapters) can be expressed as follows:

grasp → *constant* | *variable* | *rule* ({ *grasp*, })

with syntactic categories shown in italics, vertical bar separating alternatives, braces to indicate repetitive parts with the delimiter (here comma) between them shown at the right, whereas parentheses and commas being the language symbols.

As follows from this notation, an SGL scenario, applied in a certain world point (i.e. of PW, VW, EW or their combinations), in the simplest form can just be a constant defining the result explicitly. It can also be a variable containing certain data, say, assigned to it previously by some or other SGL scenarios, which may have happened to visit this point of the world before.

The next option may be one or more constants, variables, or recursively grasps again (treated as operands and separated by a comma if more than one), which are embraced by a certain operation, or *rule*, with the use of parentheses. The rules, starting in the current world position, can be of most diverse natures -- from local matter or information processing to global management and control. The rules can produce results (which may be single or multiple) in the *same or other* world locations.

Due to recursion in the language definition, the results obtained and world locations reached by rules may, in their turn, become, respectively, operands and/or starting places for other rules, with new results and new locations (single or multiple too) obtained after their completion, and so on.

The scenario can thus dynamically spread & process & match the world or its parts needed, with the scenario code capable of virtually or physically moving (the local data too, as will be clear afterwards) in the distributed space, matching the latter and possibly losing utilized parts if of no need any more. This movement can take place in single or multiple, parallel, same or different scenario parts/copies, dynamically linking with each other within automatic spatial control, spreading and covering the navigated world too.

SGL constants can represent *information*, physical *matter* (or physical objects), or *custom* defined data items extending the language for specific applications, as follows:

constant → *information* | *matter* | *custom* | { *grasp_* }

The word “constant” is used rather symbolically in the SGL definition, as the last option shown above is recursively defined as *grasp* again (possibly, even an aggregate of grasps separated by underscore), thus capable or representing any complex objects, passive or with embedded activities, for their partial or complete processing.

SGL variables, called “spatial”, which may be stationary or mobile and contain information or matter, are serving different features of distributed scenarios. As follows, they may be of the four types: *heritable* (stationary), *frontal* (mobile), *environmental* (stationary or mobile), and *nodal* (stationary), with their semantics and usage explained later.

variable → *heritable* | *frontal* | *environmental* | *nodal*

And rules can belong to the following main classes (to be explained in detail afterwards):

rule → *movement* | *creation* | *echoing* | *verification* | *assignment* | *modification* | *advancing* | *branching* | *transference* | *timing* | *granting* | *type* | *usage* | *application* |{ *grasp_* }

The final rule's option, *grasp* again, brings another level of recursion into SGL where operations may not only be explicit but can also represent results of spatial development of corresponding SGL scenarios of any world coverage and complexity. This option also offers composition or aggregates (separated by underscore) of different operations to jointly work on the operands they commonly embrace.

2.4 SGL Main Features

Here are some general aspects of SGL scenarios explaining their evolution in distributed worlds.

The basic concept is *progress point*, or *prop* (the latter word is used here as an abbreviation and differs from traditional meaning of the word "prop"). The prop identifies a *combined scenario development and control step* in the united space & time continuum. By using the notion of props, which proved to be very useful on both conceptual and implementation levels, we can clearly explain how SGL scenario operates and evolves, with key points as follows.

- Applied to some point of the world (which may be of different natures as explained before), an SGL scenario is considered to be in the *starting prop* associated with this entry point.
- When activated, the scenario develops in a stepwise manner, generally as a parallel transition between consecutive sets of props (the initial set containing the starting prop only). Each new set (or sets, as scenario may branch) identifies the final result of the current step of scenario evolution having started from the previous set (or its subsets).
- Starting from a prop, a scenario action may result in new props (which may be multiple, as a set) or remain in the same prop. In the latter case, this prop may again be added to the resulting set of props obtained from other starting props, for further common activities from all these, if multiple space & time propagations occur.
- Each prop has a resulting *value*, which may be single one representing information or matter or a list of values (potentially: nested), and a resulting control *state* (one of thru, done, fail, or fatal, with their meanings explained later).
- Different operations (represented by arbitrary scenarios) may evolve *independently* or *interdependently* in space and time from the same prop, and in *ordered*, *unordered*, or *parallel* manner.
- Operations may also *spatially succeed* each other, with new ones applied from the props reached by the previous actions. This potentially parallel wavelike evolution in space-time continuum may take place in *synchronous* or *asynchronous* mode.

- Operations and decisions represented by rules can use states and values associated with props reached by other operations, *whatever complex and remote* the latter might be.
- Any prop is always *associated* with a point of the world (i.e. physical, virtual, execution or combined node) the related scenario branch is currently developing in.
- *Any number* of props can be simultaneously associated with the *same* world points, sharing local information at them, if needed.
- Staying with world points, it is possible to *directly* access and change local parameters in them, whether physical or virtual, thus impacting the worlds (or trying to do so) via these points.
- Overall organization of the breadth and depth world navigation and coverage is provided by a variety of powerful SGL rules, which may be arbitrarily *nested* within complex processing, evolution, control, management, and supervision structures.

As was shown in previous publications, any sequential or parallel, centralized or distributed, stationary or mobile algorithm operating with information and/or physical matter can be written in SGL on any levels. The latter ranging from top semantic (also close to what is called "command intent") to those detailing system partitioning, composition, infrastructures, subordination between active components, and overall management and control.

2.5 The Sense and Nature of SGL Rules

Explaining the language basics further, let us shed some light on the sense and nature of rules, to be explained later in detail. A rule, representing in SGL any action or decision, may, for example, be as follows:

- Elementary arithmetic, string, or logic operation.
- Move or hop in a physical, virtual, execution, or combined space.
- Hierarchical fusion and return of (potentially remote) data.
- Distributed control, both sequential and parallel, and in breadth or depth.
- A variety of special contexts detailing navigation in space while influencing embraced operations and decisions.
- Type or sense of a value, or its chosen usage, guiding and simplifying automatic language interpretation.
- Creation or removal of nodes and/or links in distributed knowledge networks.
- Result of local or global operations of arbitrary complexity and space coverage, which can find, select, or produce the rule needed.

- As already mentioned, a rule can also be a compound one integrating a number of other rules.

All rules, regardless of their nature, sense, or complexity, are obeying the same ideology and organization, as follows:

- Starting from a certain space location, initially linked to it.
- Performing certain operations in a distributed space.
- Producing final results in the resultant set of props with their states and values.
- Linking to same or new world positions reached by the rule's activity.

This uniformity allows us to effectively compose highly integral and transparent spatial algorithms of any complexity and any world coverage, which can operate altogether under fully automatic, parallel and distributed control.

2.6 SGL Spatial Variables

Let us consider some details on the nature and sense of spatial variables, stationary or mobile, which can be used in fully distributed physical, virtual, or executive environments, effectively serving multiple cooperative and integral processes:

- *Heritable variables* – stationary, starting in a prop and staying with this prop permanently (even though the prop has become a past history only, with active processes already gone with other props) and serving all subsequent props, which can share them in read & write operations.
- *Frontal variables* – mobile, temporarily associated with currently active props (not being shared with other props), and then moving with the scenario evolution to subsequent props, accompanying scenario activity. These variables replicate if from a single prop a number of other props emerge.
- *Nodal variables* – stationary, being a private, direct property of the world locations/nodes reached by the scenarios. Staying at world nodes, they can be accessed and shared by all activities having reached these nodes under same scenario identity and at any time (their life span will be explained later).
- *Environmental variables* – these allow us to access different features of physical and virtual words during their navigation, also internal parameters of the distributed SGL interpretation system, to assess, guide, and optimize scenario execution. Most of them are stationary, associated with stationary world positions, but some, related to the execution system itself, can be mobile.

These types of variables, especially when used together, allow us to create advanced spatial algorithms working *in between* components of distributed systems rather than *in* them, providing flexible, robust, and self-recovering solutions. Such algorithms can freely replicate, partition, spread and *migrate* in distributed environments (partially or as an *organized whole*), preserving global integrity and overall control.

2.7 Control States and Their Hierarchical Merge

The following control states appear in props during scenario evolution in distributed space-time continuum. They are used for distributed control of multiple sequential and parallel processes, with making intelligent decisions at different levels.

- `thru` – indicates full success of the current branch of the scenario with capability of further development (i.e. indicating successful operation not only in but also *through* this stage of control). Next scenario stages, if any, will be allowed to proceed from the current prop.
- `done` – indicates success of the current stage with its planned termination after which no further development of this particular branch from the current prop will be possible (unless this status is subsequently changed by a special higher-level rule).
- `fail` – indicates non-revocable failure of the current branch, with no possibility of further development. This state relates to the current branch/prop only, not influencing directly the development of other branches of the scenario. It, however, same as the previous states, can influence decisions on higher levels by control rules which can allow or block development of other branches.
- `fatal` – reports fatal, terminal failure with nonlocal effect, triggering abortion of all evolving processes and associated temporary data, which may be parallel and distributed, also active, regardless of their current world locations and their success or failure. The scope of this global cancellation process may be the whole scenario or only its part embraced by a special rule (explained later) supervising the area in which this state may happen to occur.

These control states appearing in different branches of a parallel and distributed scenario at bottom levels can be used to obtain generalized control states for higher scenario levels, up to the whole scenario, for making proper decisions. The hierarchical bottom-up merge & generalization of states is based on their comparative importance, where the stronger state will always dominate when ascending towards the root.

For example, the merge of states thru and done will result in thru, thus generally classifying successful development at a higher scenario level with possibility of further expansion from all or at least some of its branches. Merging thru and fail will result in thru too, indicating general

success with possibility of further development despite some branch (or branches) terminated with failure, but others remained open to further evolution. Merging done and fail will result in done indicating successful termination in general while ignoring local failures, without possibility of further development in this direction.

And fatal will always dominate when merging with any other states unless its influence is restricted at top by a special rule which, in case of discovering state fatal under its supervision, will itself result with fail for higher assessment and control. So ordering these states by their powers from maximum to minimum will be as follows: fatal, thru, done, fail.

2.8 The Use of Conventional Notations

To simplify SGL programs, traditional to existing programming languages abbreviations of operations, also conventional delimiters can be used too, substituting certain rules as in numerous examples throughout this book, always remaining, however, within the general syntactic structure shown in Fig. 6. A number of such code simplifications will be used in the following sections when describing different scenarios in SGL for solving concrete problems.

2.9 Some Elementary Examples in SGL

- Just representing result directly, as a numerical, string, or custom constant:
  `77, 'Peter', Peter`
- Multiplication of two constants with the result as an open value :
  `multiply(34, 5.5)` or `34 * 5.5`
- Assigning a sum of values to variable Result:
  `assign(Result, add(27, 33, 55.6))` or
  `Result = 27 + 33 + 55.6`
- Moving to two physical locations (x1, y3) and (x5, y8) in parallel:
  `move(location(x1, y3), location(x5, y8))` or in a shortened way:
  `move(x1_y3, x5_y8)`
- Creating isolated virtual node Peter:
  `create('Peter')` or `create(Peter)` if Peter is a custom name.
- Extending node Peter as father of Alex, the latter to be a new node:
  `advance(hop('Peter'), create(+'fatherof','Alex'))` or
  `hop(Peter); create(+fatherof, Alex)` – shortened, and for custom names.
- Tasking of doer D1 to shift in physical space on coordinate deviation (dx, dy):
  `advance(hop(D1), increment(WHERE,(dx,dy)))` or
  `hop(D1); WHERE += dx_dy`

(WHERE is a special environmental variable, explained later, keeping physical coordinates of the node, here D1, in which scenario control is currently staying.)

- Tasking D1 to move directly to new physical coordinates (x, y) will be as follows:
  `advance(hop(D1), assign(WHERE,(x, y)))` or
  `hop(D1); WHERE = x_y`

2.10 Full SGL Summary

SGL full description summarizing the listed above language constructs is as follows where, as already mentioned, syntactic categories are shown in italics, vertical bar separates alternatives, and parts in braces indicate zero or more repetitions with a delimiter at the right. The remaining characters and words are the language symbols (including braces shown in bold).

grasp → *constant* | *variable* | *rule* ({ *grasp*, })
constant → *information* | *matter* | *custom* |{ *grasp_* }
variable → *heritable* | *frontal* | *nodal* | *environmental*
rule → *movement* | *creation* | *echoing* | *verification* | *assignment* | *modification* | *advancing* | *branching* | *transference* | *timing* | *granting* | *type* | *usage* | *application* | { *grasp_* }
information → *string* | *number* | *special*
string → ‘{*character*}’ | **{**{*character*}**}**
number → *standard* | `zero` | `one` | `two` | `three` | `four` | `five` | `six` | `seven` | `eight` | `nine` | `plus` | `minus` | `dot`
matter → “{*character*}”
movement → `hop` | `move` | `shift`
creation → `create` | `linkup` | `delete` | `unlink`
echoing → `state` | `order` | `rake` | `element` | `content` | `index` | `count` | `sum` | `first` | `last` | `min` | `max` | `random` | `average` | `access` | `sortup` | `sortdown` | `reverse` |`add` | `subtract` | `multiply` | `divide` | `degree` | `separate` | `unite` | `attach` | `append` | `common`
verification → `equal` | `notequal` | `less` | `lessorequal` | `more` | `moreorequal` | `none` | `empty` | `nonempty` | `belongs` | `notbelongs` | `intersects` | `notintersects`
assignment → `assign` | `remove` | `withdraw` | `assignpeers`
modification → `inject` | `replicate` | `split`
advancement → `advance` | `slide` | `repeat` | `fringe`
branching → `branch` | `sequence` | `parallel` | `if` | `or` | `and` | `choose` | `firstrespond` | `cycle` | `loop` | `sling` | `whirl` | *empty*
transference → `run` | `call` | `input` | `output` | `transmit` | `send` | `receive`

timing → `sleep` | `remain`
granting → `supervise` | `release` | `free` | `blind` | `lift` | `none` | `stay` | `seize`
type → `heritable` | `frontal` | `nodal` | `environmental` | `matter` | `number` | `string`
usage → `address` | `coordinate` | `content` | `index` | `time` | `speed` | `name` | `place` | `center` | `range` | `doer` | `node` | `link` | `unit` | `scenario` | `world` | *empty*
heritable → H {*alphameric*}
frontal → F {*alphameric*}
nodal → N {*alphameric*}
environmental → `TYPE` | `NAME` | `ADDRESS` | `QUALITIES` | `WHERE` | `BACK` | `PREVIOUS` | `PREDECESSOR` | `DOER` | `RESOURCES` | `LINK` | `DIRECTION` | `WHEN` | `TIME` | `SPEED` | `STATE` | `VALUE` | `COLOR` | `IN` | `OUT` | `STATUS` | *specific*
special → `thru` | `done` | `fail` | `fatal` | `infinite` | `nil` | `nodes` | `links` | `any` | `all` | `allother` | `passed` | `existing` | `neighbors` | `direct` | `noback` | `firstcome` | `forward` | `backward` |

3. DISTRIBUTED SGL INTERPRETER

3.1 The Interpreter Components and Structure

The internal organization of SGL interpreter (in software, hardware or both) is shown in Figure 7. The interpreter consists of a number of specialized modules working in parallel and handling & sharing specific data structures supporting both persistent virtual worlds and temporary data and hierarchical control mechanisms.

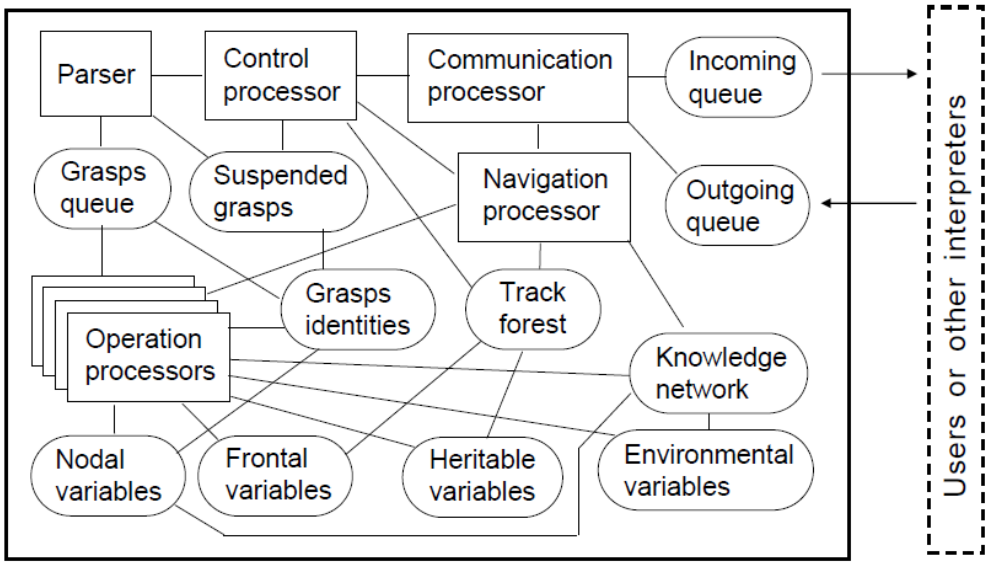


Figure 7: Organization of SGL Interpreter

The "nerve system" of the distributed interpreter is its *spatial track system* with its parts kept in the Track Forest memory of local interpreters. These being logically interlinked with similar parts in other interpreter copies, forming altogether *global control coverage*. This forest-like distributed track structure enables for hierarchical control as well as remote data and code access, with high integrity of emerging parallel and distributed solutions, without any centralized resources.

The dynamically crated track trees (generally: forests), spanning the systems in which SGL scenarios evolve, are used for supporting spatial variables and echoing & merging different types of control states and remote data, self-optimizing in parallel echo processes and providing automatically of what is usually called (adaptive) command and control, or C2. They also route further grasps to the positions in physical, virtual, execution or combined spaces reached by the previous grasps, uniting them with frontal variables left there by the preceding grasps.

3.2 SGL Interpreter as a Universal Spatial Machine

The whole network of the interpreters can be mobile and open, changing at runtime the number of nodes and communication structure between them. Copies of the interpreter can be concealed if to operate in hostile environments, allowing us to analyze and impact the latter in a stealth manner, if needed.

The dynamically networked SGL interpreters are effectively forming a sort of *universal parallel spatial machine* (as shown in Figure 8) capable of solving any problems in a fully distributed mode, without any special central resources. "Machine" rather than computer or "brain" as it can operate with matter too, and can move partially or as a whole in physical environment, possibly, changing its distributed shape and space coverage. This machine can operate simultaneously on many mission scenarios which can be injected at any time from its arbitrary nodes.

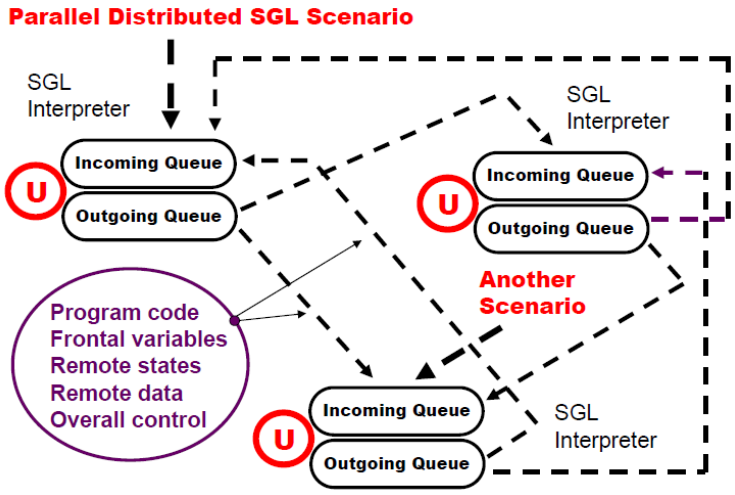


Figure 8: SGL interpretation Network as a Universal Spatial Machine

Installing communicating SGL interpreters into mobile robots (ground, aerial, surface, underwater, space, etc.) on top of their existing functionality allows us to organize effective group solutions of complex problems in distributed physical spaces in a clear and concise way, *effectively shifting traditional management routines to automatic levels*. Human-robot interaction and gradual transition to fully unmanned systems are drastically assisted too.

Some hypothetic integrative scenario skeletons, uniting very dissimilar types of robotic units (ground, surface, underwater, space), all operating under the unified command and control provided by SGT via embedded SGL interpreters communicating with each other, are shown in Figure 9.

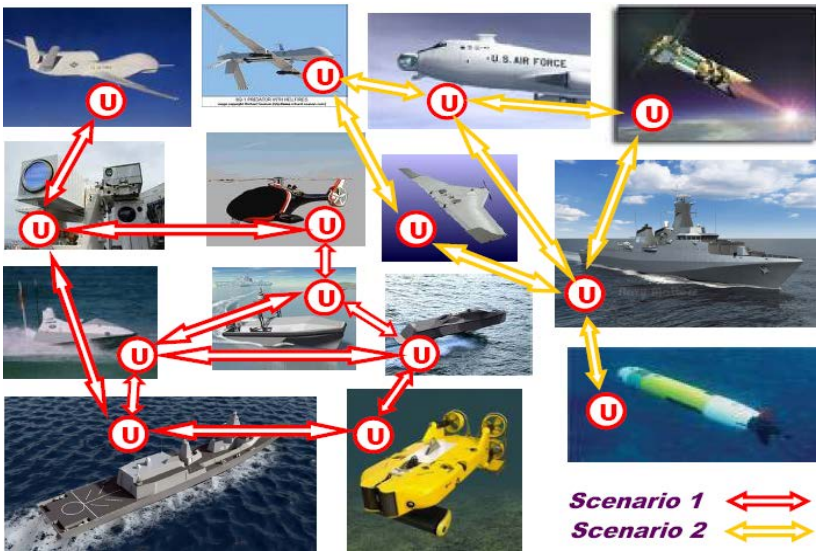


Figure 9: Possible Cooperative Scenario Skeletons

4. COOPERATIVE ROBOTICS

After embedding SGL interpreters into robotic vehicles, we can provide any needed collective behavior of them—from loose swarming to a strictly controlled integral unit obeying external orders. Any mixture of different behaviors within the same scenario can be easily programmed too.

We will consider here some collective robotic scenarios in SGL operating on different organizational levels and their integration under the unified control provided by the automatic language interpretation.

4.1 Integration of Loose Swarming with Hierarchical Command and Control

Imagine that a distributed area needs to be investigated by multiple unmanned aerial vehicles that should randomly search the space, collect information on unwanted objects, classifying them as targets, and organize collective reaction on emerging threats. Different group functionalities for this can be expressed in SGL which can be effectively integrated into the resultant holistic group scenario, as follows.

- Swarm movement scenario, starting from any unit (let us call this *swarm_move*):

```
hop(all_nodes);
Limits = (dx(0,8), dy(-2,5)); Range = 500;
repeat(Shift = random(Limits);
       if(empty(hop(Shift, Range), move(Shift)))
```

A snapshot of such swarm movement is shown Figure 10.

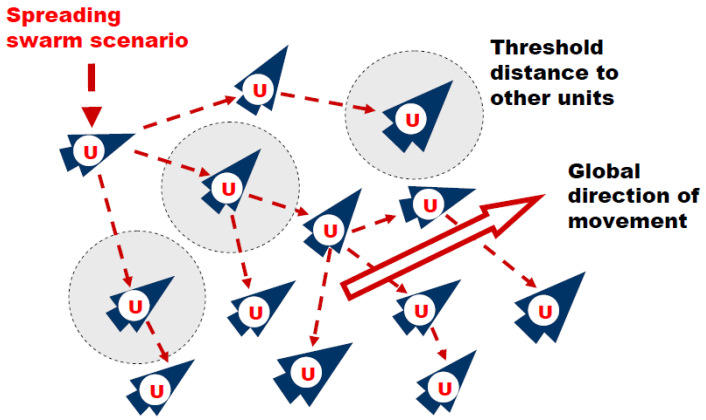


Figure 10: Swarm Movement Snapshot

- Finding topologically central unit and hopping into it, starting from any unit (*find_hop_center*):

```
frontal(Aver) =
    average(hop(all_nodes); WHERE);
hop(min(hop(all_nodes);
distance(Aver, WHERE) & ADDRESS):2)
```

The center found by this scenario is shown in Figure 11.

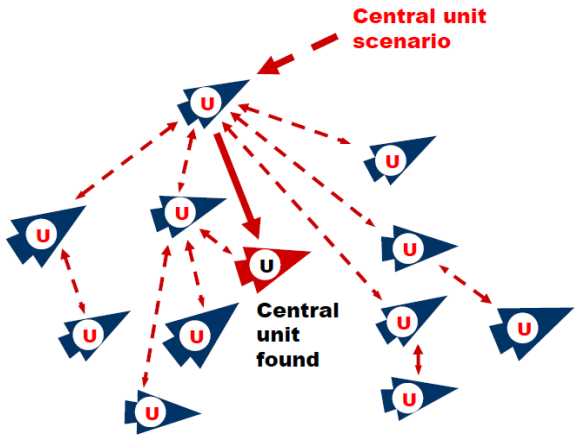


Figure 11: Finding Central Unit

- Creating runtime infrastructure starting from the central unit found (*infra_build*), see Figure 12:

```
stay(repeat(linkup(+infra, first, Depth)))
```

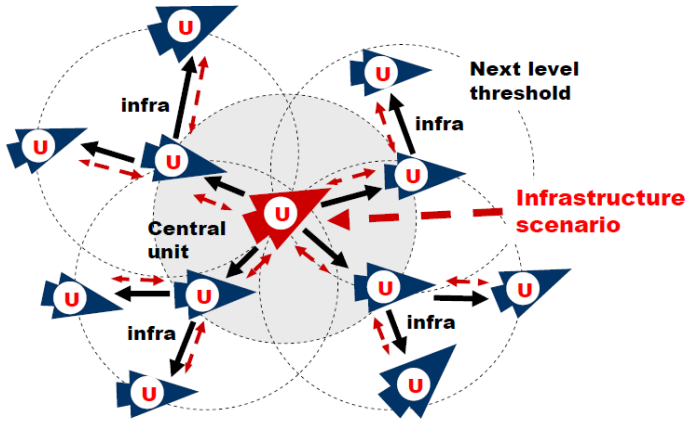


Figure 12: Runtime Infrastructure Creation

- Regular targets collection & distribution & impact (*collect_distribute_impact*) starting from the central unit found, as in Figure 13:

```
loop(
  nonempty(frontal(Seen) =
      repeat(free(detect(targets)), hop(+infra)));
  repeat(free(select_shoot(Seen)), hop(+infra)))
```

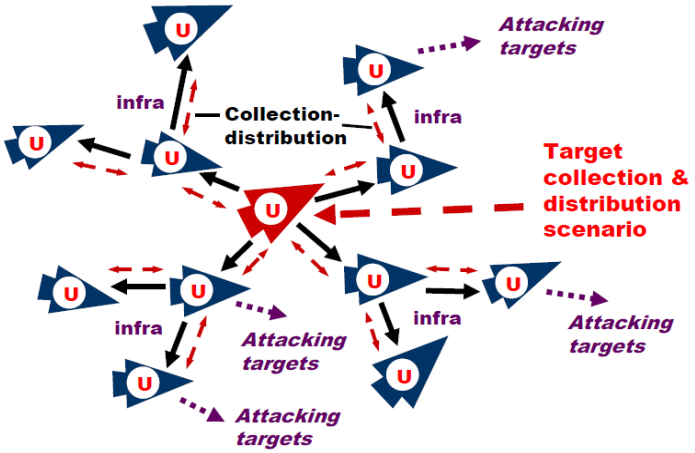


Fig. 13. Targets Collection, Dissemination, and Impact

- Removing previous infrastructure (before creating a new one), starting from any unit (*infra_remove*):

```
stay(hop(allnodes); remove(alllinks))
```

The resultant combined solution (integrating previous SGL programs named in italics), starting from any unit, will be as (where *time* is meant to be a certain time interval):

```
parallel(
  swarm_move,
  repeat(
```

```
find_hop_center;
infra_remove; infra_build;
remain(time_delay, collect_distribute_impact)))
```

The obtained resultant scenario combines loose, randomly oriented swarm movement in a distributed space with periodic updating of topologically central unit (as units are changing distances and relative positions) and updating runtime hierarchical infrastructure between them. This infrastructure controls observation of distributed territory while collecting potential targets, distributing them back to the vehicles for local assessment selection and impact.

4.2 Collective Patrol of Coastal Waters

This scenario may be suitable for both surface and varying depth underwater search of intrusions in the coastline zone, but for simplicity we will be assuming here only two dimensional space to be navigated.

At the beginning let us create a distributed coastal waypoint map in the form of embedded semantic network, as in Figure 14 (r being arbitrary name of links between nodes-waypoints). The corresponding DSL solution is as follows.

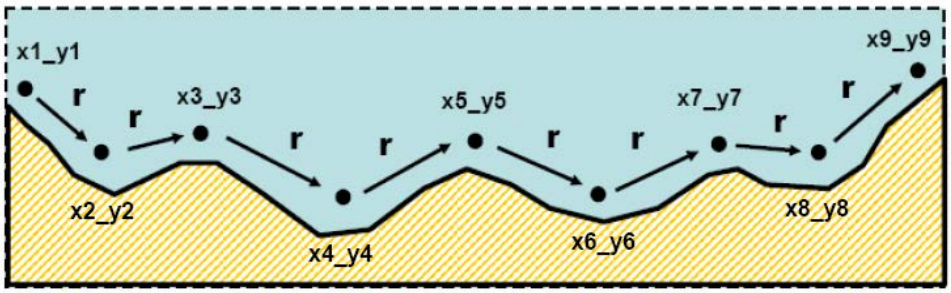


Figure 14: Coastal Waypoint Map

```
create_physical(
   #x1_y1; +r#x2_y2; +r#x3_y3; … +r#x9_y9)
```

where xi, yi represent concrete coordinates.

A single USV (or UUV) solution repeatedly navigating all coastal area by the map created is shown in Figure 15 and DSL program that follows (searching the water space for alien objects by the depth available by vehicle's sensors).

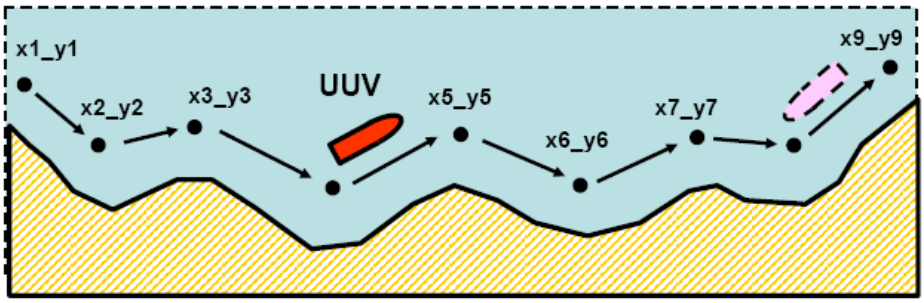


Figure 15: Patrolling Coastal Waters with a Single Vehicle

```
frontal(Link =+r, SearchDepth = …);
move(x1_y1);
repeat(
  repeat(move(Link); check_report(SearchDepth, alien));
  invert(Link))
```

Two-vehicle simultaneous solution is shown in Figure 16 and by the following program, with vehicles moving according to the coastal map independently, assuming each having automatic procedures for avoiding possible collisions with the other vehicle.

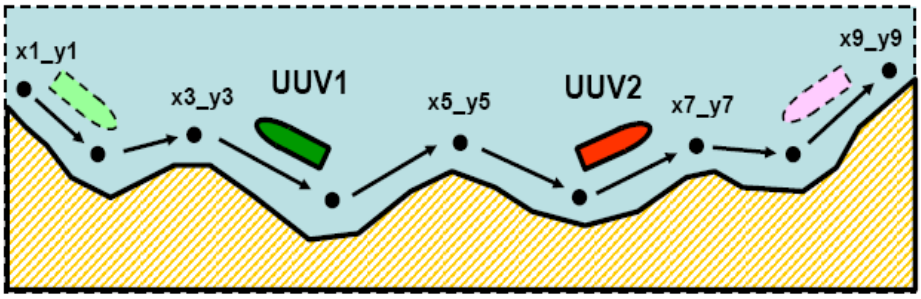


Figure 16: Patrolling coastal waters with two vehicles

```
frontal(Link, SearchDepth = …);
branch(
  (Link = +r; move(x1_y1)),
  (Link = -r; move(x9_y9)));
repeat(
  repeat(check_report(SearchDepth, alien); move(Link));
  invert(Link))
```

Another solution for the two-vehicle case may be when each vehicle turns back if discovers another patrol vehicle on its way, checking for this its vicinity with depth given.

```
frontal(Link, SearchDepth = …, CollisionDepth = …);
branch(
  (Link = +r; move(x1_y1)),
  (Link = -r; move(x9_y9)));
repeat(
  repeat(none_seen(CollisionDepth, patrol);
         check_report(SearchDepth, alien));
         move(Link));
  invert(Link))
```

For the both cases, the whole coastline will always be searched in full if at least a single vehicle remains operational. The current scenario can be easily extended to more then two vehicles searching space cooperatively.

4.3 Cooperative Finding of Oil Spill Center

This scenario tries to find oil spill center at sea by cooperating multiple surface or underwater unmanned vehicles distributed initially throughout the polluted region, as shown in Figure 17.

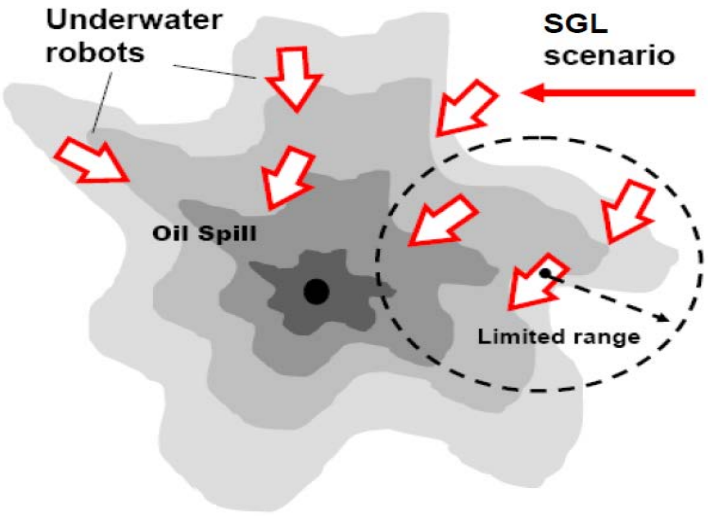


Figure 17: Cooperative Finding of the Spill Center

The distributed scenario operates as follows.

- It starts from any robot, covering the whole group by limited local channels.
- Each robot randomly tries to move toward increasing spill level,
- Each movement is allowed only if some other robots remain in reach, to preserve connectivity between robots as a network.
- By regular local communications between robots the maximum recorded oil spill level is updated in all robots.
- The robot(s), in which maximum level of the whole region corresponds to locally checked level for a threshold period of time, report the center of spill.

The expression of this scenario in SGL will be as follows (words in italics to be substituted by real values):

```
move(coord1, coord2, …, coordn);
nodal(Level = check(spill), Direction,
      Current, Count, Max);
parallel(
   loop(Max = maximum(
          (hop(range, all); Max), Max, Level);
        Level == Max != 0; Count += 1;
        if(Count >= threshold,
```

```
            output('Center', WHERE));
        sleep(Delay)),
    loop(Current = WHERE;
        or((WHERE += random(shift, Direction);
            nonempty(hop(range, all));
            New = check(spill) > Level;
            Level = New; Count = 0;
            Direction = angle(Current, WHERE)),
           WHERE = Current)))
```

5. CONCLUSIONS

We have described a new type of distributed management philosophy and supporting networking technology based on a completely different type of management language than usual. This language, SGL, is oriented on programming and processing of distributed both physical and virtual systems and worlds directly and on a high semantic level, allowing at the same time to express organization, management, and control details on any other levels too, if needed, which are usually shifted to automatic SGL interpretation in collaborative environments.

With the use of SGL, the whole distributed world can be considered as an integral and universal spatial machine capable of solving arbitrary complex problems (*machine* rather than *computer* as it directly operates with physical matter/objects too). Multiple communicating "processors" or "doers" of this machine can include humans, computers, robots, smart sensors, any mechanical/electronic equipment capable of cooperatively solving complex problems formulated in SGL.

Being understandable and suitable for both manned and unmanned components, the language offers a real support for unified transition to robotized systems as within execution of operational scenarios in it any components can easily change, at runtime including, their manned to unmanned status and vice versa, also enabling *fully unmanned solutions* if these may happen to be needed for specific applications.

REFERENCES

[1]. Minsky M (1988), The society of mind. Simon and Schuster, New York.

[2]. Feliciano CN (2009), The army's future combat system program (Defense, Security and Strategy Series). Nova Science.

[3]. Wilber K (2009), Ken Wilber online: waves, streams, states, and self—a summary of my psychological model (or, outline of an integral psychology). Shambhala Publications.

[4]. Smuts JC (2007), Holism and evolution. Kessinger Publishing, LLC.

[5]. Wertheimer M (1924), Gestalt theory. Erlangen, Berlin.

[6]. Sapaty P (2009), Gestalt-based ideology and technology for spatial control of distributed dynamic systems. Proc. International Gestalt Theory Congress, 16th Scientific Convention of the GTA. University of Osnabrück, Germany.

[7]. Sapaty P (2009), Gestalt-based integrity of distributed networked systems. SPIE Europe Security + Defence, bcc Berliner Congress Centre, Berlin Germany.

[8]. Schade U, Hieb MR (2006), Formalizing battle management language: a grammar for specifying orders. 06S-SIW-068, presented at the 2006 Spring Simulation Interoperability Workshop, April 2006, Huntsville, AL.

[9]. Tyler T (2011), Memetics: memes and the science of cultural evolution. CreateSpace Independent Publishing Platform.

[10]. Sapaty PS (2013), The world as an integral distributed brain under spatial grasp paradigm. Proc. International Science and Information (SAI) Conference, 7-9 October, London, UK.

[11]. Sapaty P (2012), Logic flow in active data. Book chapter in: VLSI for Artificial Intelligence and Neural Networks, Springer; Softcover reprint of the original 1st ed. 1991 edition.

[12]. Sapaty PS (2012), Distributed air & missile defense with spatial grasp technology. Intelligent Control and Automation, Scientific Research, Vol.3, No.2.

[13]. Sapaty PS (2011), Meeting the world challenges with advanced system organizations. Book chapter in: Informatics in Control Automation and Robotics, Lecture Notes in Electrical Engineering, Vol. 85, 1st Edition, Springer.

[14]. Sapaty PS (2009), Providing spatial integrity for distributed unmanned systems. Proc. 6th International Conference in Control, Automation and Robotics ICINCO 2009, Milan, Italy.

[15]. Sapaty P, Sugisaka M, Delgado-Frias MJ, Filipe J, Mirenkov N (2008), Intelligent management of distributed dynamic sensor networks. Artificial Life and Robotics, Volume 12, Numbers 1-2 / March, Springer Japan.

[16]. Sapaty P (2008), Distributed technology for global dominance. In: Raja Suresh (ed): Proc. of SPIE, Vol. 6981, Defense Transformation and Net-Centric Systems.

[17]. Sapaty PS (2005), Ruling distributed dynamic worlds. John Wiley & Sons, New York.

[18]. Sapaty PS (1999), Mobile processing in distributed and open environments. John Wiley & Sons, New York.

[19]. Sapaty PS, Corbin MJ, Seidensticker S (1995), Mobile intelligence in distributed simulations. Proc. 14th Workshop on Standards for the Interoperability of Distributed Simulations, IST UCF, Orlando, FL, March.

[20]. Sapaty P (1993), A distributed processing system. European Patent No. 0389655, Publ. 10.11.93, European Patent Office.

TNC Transactions on Networks and Communications
SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
VOLUME 2, ISSUE 2
ISSN: 2054 - 7420

Nanorobots in Medicine-A New Dimension in Bio Nanotechnology

T.Venkat Narayana Rao[1], H. S. Saini[2], Pinnamaneni Bhanu Prasad[3]
[1]*Guru Nanak Technical Campus. Ibrahimpatanm , A.P, India.*
[2]*Guru Nanak Institutions, Ibrahimpatanm , A.P, India*
[3]*Guru Nanak Institutions, Ibrahimpatanm and Vision Specialist, Matrix vision GmbH, Germany*
[1]tvnrbpbby@yahoo.com

ABSTRACT

Bio-nanotechnology has become a hopeful area of research that is bringing radical advancements and changes in the current century of technological mutiny. It is a one part of Nano technology**.** It has shown its clear participation in all fields, there is an integral part of the nanos in human science and medicine. Nanomedicine is the process of diagnosing cancers, treating, preventing disease, relieving pain, and of preserving and improving human health using molecular tools and molecular knowledge of the human body. Most of the symptoms such as fever and itching have specific biochemical reasons that can also be controlled, reduced, and eliminated using the appropriate injected nanorobots. This paper mainly concentrates on reviewing role and implementation of nanorobots in medical field and how it can replace present medical scenarios with cost reduction along with vision to permanent solutions to many human aliments towards surgery less treatments.

Keywords: Bio-nanotechnology; Nanomedicine; Nanorobots, Cancer.

1. INTRODUCTION

A Nanorobot is a miniature machine designed to perform a specific task with exactness of nanoscale dimensions which is precisely dimension of a few nanometers (nm) or less. Nanobots have potential applications in the fields of assembly, maintenance and manufacturing of advanced devices, machines, circuits, at the atomic and molecular level. This process is also termed as molecular manufacturing. Nanobots have given rise to the novel field of nanomedicine. It has been suggested that a convoy of nanobots might serve as antibodies or antiviral agents in patients. There are several other potential medical applications, including repair of damaged tissue, unblocking of arteries affected by plaques. This include building complete replacement of body organs[5]. The major gain of nanobots is considered to be their durability. They can significantly perform the task of self-replication, which is the process of

DOI: 10.14738/tnc.22.131
Publication Date:4th April 2014
URL: http://dx.doi.org/10.14738/tnc.22.131

replicating themselves to replace depleted units. According to the theory, they can remain operational for years, decades, or centuries.

Nanoscale systems can also operate much faster than their larger counterparts because displacements are lesser . This allows mechanical and electrical events to happen in less time at a given speed. Nanotechnology is reasonably straightforward to understand, but developing this widespread knowledge into a nanorobot has been extra complicated. Humans are able to perform one nano-function at a time, but there are plenty of varied applications which require a construction of autonomous robot which would be exceedingly tiresome for us, irrespective of how high-tech the laboratory. So it has become very much necessary to create a whole set of specialized machine tools in order to speed the process of nanobot building[2].

2. COMPONENTS OF NANOROBOTS

Virtual Reality was used for the nanorobot design where is considered as a suitable approach for the use of macro and microrobotics concepts. The nanorobots are made of gold and silver colloid balls, as small as two nanometers along with carbon atoms arranged in a diamondoid structure.

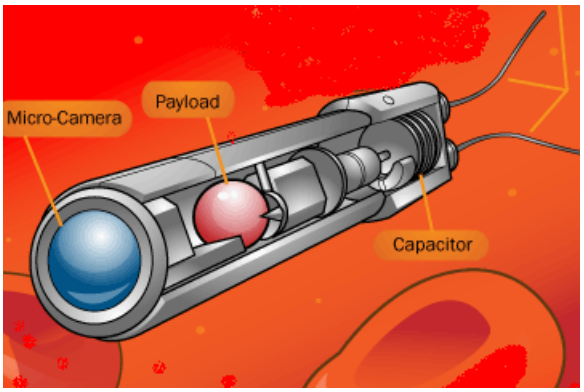


Figure 1: How Nanorobot looks like and its components.

The nanorobot design is inspired by the human body derived from biological models and is comprised by some components such as Molecular Sorting Rotor a robot arm[3]. The nanorobot movements was assumed some concepts provided from underwater robotics. The trajectories and position of each molecule, which has to be captured and assembled. Arbitrarily each one has probabilistic motion of acceleration. The nanorobot uses a macro transponder navigational system in order to address nanorobot positioning, which may keep high positional accuracy to each nanorobot's direction. Such a system might involve externally generated signal from beacons placed at fixed positions outside the skin. Thus, the delivery positions that represent organ inlets requiring proteins to be injected are located in a well-known location for the nanorobot [1].

Sensors that inform if a collision happens and identify when it is an obstacle through mircocamera, which in such cases will require a new trajectory planning, comprise the nanorobot. Plane surfaces (three fins total) and bi-directional propellers are used for the

navigation, which is comprised by two simultaneously counter-rotating screw drives for the propulsion. A binary indication is used to trigger the behavioral response as a common mechanism for action and for prevailing different phases of activity in tasks as done by social insects. In this way, activation of a motor behavior is not reliant on a specific perceptual cue, but rather on the verdict that results from sensor processing. Either touch sensors or infrared sensors can provide the information. The advantage is that the design of the motor behavior does not alter, when different sensor types of feature extraction techniques are used. Since the information needed by the motor behavior is the same binary vector in both the cases for the kinetics assumptions. The nanorobot lives in a world of adhesion, viscosity, where friction and viscous forces are dominant. Observing environmental characteristics related to nano-worlds the gravitational force here is negligible. Propellers are also extremely used to drive forward and backward. They are bi-directional and are fitted along with the propellers used. The fins that are used in the nanobots are precisely used to drive the device. The fins use plane surfaceas shown in the figure 2. In nanobots we make essential usage of the WIFI-CMOS, hence wireless communication is feasible way to interface. It enables nanorobots in the basic operations such as tracking and diagnosis. CMOS is used for the fabrication of high performance nano devices.

3. DESIGN OF NANOROBOTS

There are two basic kinds of nanorobots; assemblers and self replicators. Assemblers are simple cell shaped nanorobots that are able to interpret molecules or atoms of different types, and are controlled by specific specialized programs. Self-replicators are fundamentally assemblers that are capable of duplicating themselves at a very large, fast rate; the facility to self-replicate or clone themselves, and possess subsequent ability to work in unison to build macro-scale devices[1].

3.1 Working of Nanorobots

The nanorobot would probably be a micron-scale robot assembled from nanoscale parts. These parts could range in size from 1-100 nm (1 nm = 10^{-9} meter), and might be built-in jointly to make a working machine measuring perhaps 0.5-3 microns (1 micron = 10^{-6} meter) in diameter. The microns is about the maximum size for blood borne medical nanorobots, due to the capillary passage requirement. Carbon will likely be the chief element comprising the bulk of a medical nanorobot, most likely in the form of diamond or diamonded/fullerene nano composites largely because of the tremendous strength and chemical inertness of diamond. Many other light elements such as nitrogen, fluorine, silicon, hydrogen, sulfur, oxygen etc. will be used for special purposes in nanoscale gears and other components. It works on new concepts of novel concept of software (made up of DNAs) and hardware (made up of enzymes) molecular elements[4].

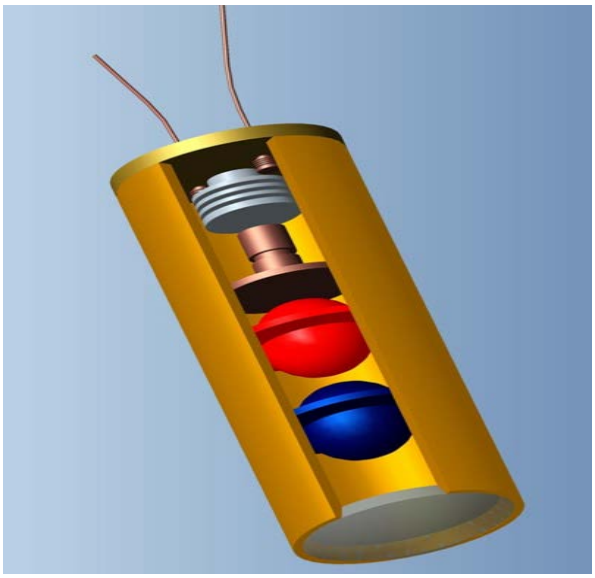


Figure 2. Woking of nanorobot with payload.

It is impossible to say precisely what a general nanorobot would look like. Nanorobots intended to travel through the bloodstream to their target will probably be 500-3000 nanometers (1 nanometer =10^{-9} meter) in characteristic dimension. Non-blood bearing issue-traversing nanorobots might be as bulky as 50-100 microns, and alimentary or bronchial traveling nanorobots may be even larger still. Each species of medical nanorobot is to be designed to achieve a specific task, and many shapes and sizes are likely. The best nanobot consist of a transporting mechanism, an internal processor and a fuel unit that enables it to do task. The main difficulty arises around this fuel unit, since majority of usual forms of robotic propulsion cannot be minimized to nanoscale with existing technology. Scientists have succeeded in reducing a robot to five or six millimeters, but this size still technically qualifies it as a macro-robot. One possible solution is to stick on a fine film of radioactive particles to the nanobot's body. As the particles decay and release energy the nanobot would be able to harness this power source; radioactive film can be enlarged or reduced to any scale without a drop in efficiency occurring[7].

3.2 The External Control Mechanism

Consists of control mechanism employs affecting the dynamics of the nanorobot in its work environment through the application of external potential fields. Researchers are keenly looking at using MRI as an external control mechanism for guiding the nano particles. An MRI system is capable of generating variable magnetic field gradients which can exert force on the nanorobot in the three dimensions and control its movement and direction. Professor Martel's laboratory is exploring effect of such variable magnetic field on a ferromagnetic core that could probably be embedded into the nanorobots[1].

Other possibilities being explored are in the category of 'hybrid' control mechanisms where the target is located and fixed by an external navigational system but the behavior of the nanorobot is determined locally through an active internal control mechanism. The use of nano

sensors and evolutionary agents to determine the nanorobots behavior is suggested by the mentioned reference.

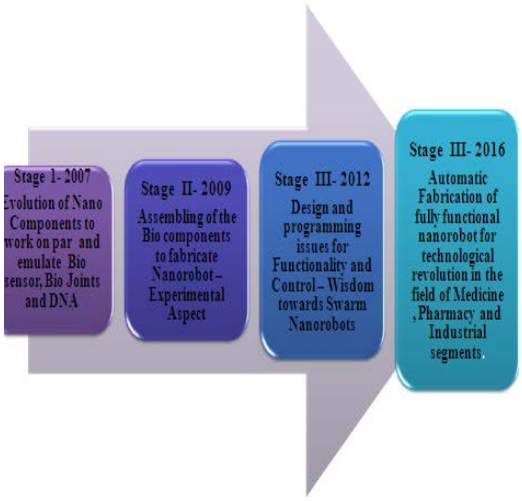


Figure 3: The Roadmap, illustrating the system capabilities improving in progression with Bio-Nanorobotic Components with time.

3.3 Internal Control Mechanism

The Internal Control Mechanism is both active and passive. This type of control relies on the mechanism of bio chemical sensing and selective binding of various bio molecules with a variety of other elements. This is a traditional method, which has been in use since quite some time for designing bio molecules. Using the properties of the various bio molecules and combining with the knowledge of the target molecule that is to be influenced, these mechanisms could be valuable. However, this is a passive control mechanism where in, at run time these bio molecules cannot change their behavior. Once programmed for a particular kind of molecular interaction, these molecules stick to that. The key issues is controlling the nanorobots which are supposed to be intelligent and hence programmed and controlled to be effective in ever dynamic environment. The question of actively controlling the nanorobots using internal control mechanism is a difficult one. We require an 'active' control mechanism for the designed nanorobots such that they can vary their behavior based on situations they are subjected to, similar to the way macro robots perform. For achieving this internal control, the idea of molecular computers could be utilized. Leonard Adleman (from the University of Southern California) introduced DNA computers a decade ago to solve a mathematical problem by utilizing DNA molecules. Professor Ehud Shapiro have recently developed and has been successful in programming the biomolecular computer to analyze the biological information, which could detect and treat cancer. The molecular computer has an input and output module, that acts together, can diagnose a particular disease and in response produce a drug to cure that disease. It uses novel concept of software (made up of DNAs) and hardware (made up of enzymes) molecular elements. This molecular computer is in generalized form and can be used for any disease which produces a particular pattern of gene expression related to it[4].

3.4 The Current Start of Art in Nanorobots

Much progress has been made in the field of nanotechnology and nanomedicine, which has instigated the study of the use of robots in the nanometer scale known as nanorobots. The technology of nanorobots has grow to be a raging topic and advanced research is being carried out for the use of robots. The therapeutics of diverse fatal diseases is also in consideration for various biomedical applications and experiments in nanomedicine. The building of biosensors and the nanokinetic devices are a main requirement in the process and locomotion of nanorobots. Though, nanorobots remain to be a part of scientific fiction but this would be a clinical aspect in reality fir the future medical diagnostics. Manipulation of nanorobots is a technology that is facilitated by the Nano-Electro-Mechanical Systems or NEMS. With various new materials and structures in nanoscale, NEMS will assist in the development of new nanosensors and nanoactuators. The science of nanorobotics plays a vital role in the expansion of robots, whose structure is built by using nanoscale components and objects. The nature of the components being in the nano scale allows the researchers for the engineering of the imitation of human beings. The assembly of the various complex parts, which comprise the robots have been possible due to nanorobotics. Nanobots, nanoids, nanites, or nanomites are some of the theoretical devices created with the knowledge of nanorobotics. Development of bio-nano components from biological systems is the fore most step towards the design and development of an complex bio-nanorobot, which could be used in future applications. Since the planned systems and devices will be composed of these components, one must have a good understanding of how these perform and how could they be controlled. From the simple elements such as structural links to more advanced concepts such as motors, each component must be carefully studied and possibly manipulated to understand the practical limits of each one of them. DNA and carbon nanotubes are being fabricated into various shapes and sizes, enabling possibilities of constructing newer and complex devices. These nano-structures are potential candidates for amalgamating and accommodating the bio-nano components within them. Proteins such as rhodopsin and bacteriorhodopsin are a few examples of such bio-nano components. Both these proteins are naturally found in biological systems as light sensors. They can essentially be used as solar collectors to gather abundant energy from the sun. This energy could be produced (in terms of proton force) either for later use or could be consumed straight away by other components, such as the ATP synthase nano rotary motor. The initial work is planned to be on the bio-sensors i.e. in terms of heat shock factor. These sensors will form an integral part of the proposed bio-nano assemblies, where these will be integrated within a nano structure and will be activated, as programmed, for gathering the required information at the nano scale. The tools and techniques from molecular modeling and protein engineering will be used to design these modular components as shown in Figure 3. Bio-nanorobotics is a truly multidisciplinary field and how it has evolved step by step. DNA which may be used in a variety of ways such as a structural element and a power source, hemagglutinin virus can be used as a motor and bacteriorhodopsin could be used as a sensor or a power-source [6].

3.5 Bio to Bio Robotic Mapping Aspects

This section introduces a mapping of Bio natural characteristics with that of capabilities of proposed nanorobot functionalities.

Bio Code Notation and Functionality	Capabilities Targeted Applications for the proposed fabrication of Nanorobot
E-Energy Storage and Carrier	-Ability to store energy from various sources such as, Solar, chemical for future usage and for its own working - Required for the working of all the bio-chemical mechanisms of the proposed bio-nano-robotic systems
M-Mechanical	-Ability to precisely move and orient other molecules or modules at nano scale. This includes ability to mechanically bind to various target objects, and carry them at desired location. - Carry drugs and deliver it to the precise locations. - Move micro world objects with nano precision. For example, Parallel platforms for nano orientation and displacements.
S-Sensory	-Sensing capabilities in various domains such as, chemical, mechanical, visual, auditory, electrical, magnetic -Evaluation and discovery of target locations based on either chemical properties, temperature or others characteristics.
G-Signaling	- Ability to amplify the sensory data and communicate with bio-systems or with the micro controllers. - Capability to identify their locations through various trigger mechanisms such as fluorescence - Imaging for Medical applications or for imaging changes in Nano Structures
F-Info. Storage	-Ability to store information collected by the sensory element. Behave similar to a read and write mechanism in computer field - Store the sensory data for future signaling or usage - Read the stored data to carry out programmed functions. - Back bone for the sensory bio-module - Store nano world phenomenon currently not observed with ease
W-Swarm Behavior	- Exhibit binding capabilities with --to perform distributive sensing, intelligence actions -Performed by the bio-nano robots will be planned and programmed keeping in mind the swarm behavior and capabilties
I-Bio Nano Intelligence	-Capability of making decisions and performing Intelligent functions -Ability to make decision
R-Replication	-Replicate themselves when required - Replicate at the target site and -Replication of a particular bio-module as per the demand of the situation

3.6 Nanorobots in Medical Field

The rapidly increasing field of nanotechnology has many useful and direct applications for the medical industry. The **nanorobots** are no exception to this rule. The medical science wants to create nanobots that can repair damaged tissue without hurt and trauma. Many of the medical procedures we employ today are very painful to the human body and do not work in agreement with our natural systems. Chemotherapy wreaks havoc on humans and nearly kills them in the pursuit to kill off their hateful cancer cells. Persistent surgical procedures are also quite common now, with associated traumas that cause several patients to die rather than healing. Nanorobots are so small that they actually relate on the same level as bacteria and viruses do, and so they are competent of blending with the very particles of our bodies like atoms and molecules. The ideal nanobot has not yet been fully fabricated, but when this microscopic robot makes its inevitable introduction it will be hailed as a lifesaver by the world of medicine. We can list out numerous uses of the nanobots in the following section[2].

3.7 Simulation of Nanorobotic Environment

Simulation is an indispensable tool for exploring alternatives in the configuration, motion planning and control of nanomachines exploring the human body which is practically not feasible during testing period of the proposed technology. The chief purpose of this work is to give a better understanding on the operational principle of the nanorobot through simulated environment. The technique, used will be explained with reference to the working principle of the nanorobot in the human body. The simulation approach is to enable the development of nanorobots operating in a fluid environment pertinent for medical applications. We propose a practical simulator that allows fast design methods comparing various control algorithms for nanorobots and their appropriateness for different tasks. The simulator includes identifiable targets, obstacles and thereby providing a fitting environment for a characteristic nanorobotic tasks and administrating the desired chemical/drug compositions near specific areas in human body. By the usage of simulated bio-nano environment in the ambit of virtual reality, the operator can control, design and characterize through physical simulation and 3D visualization. Based on the simulated results the real scenario of drug administration through nanrobots can be envisaged and drug testing on animals can be minimized.

3.8 Applications of Bio-Nanorobots

3.8.1 Microbivore nanobots: These nanobots would function similarly to the white blood cells in our bodies, but they are designed to be much faster at destroying bacteria. This type of nanobots should be able to eliminate bacterial infections in a patient within minutes, as opposed to the weeks required for antibiotics to take effect. Microbivore nanobots are designed so that antibodies attach to the particular bacteria the robot is seeking. Once bacteria attaches to an antibody, an arm grabs the bacteria and moves it to the inside of the nanorobot,

where it is destroyed. After the task the bacteria is then discharged into the bloodstream as harmless fragments.

3.8.2 Respirocyte nanobots: These nanobots would function in a similar way to the red blood cells in our bodies; however, they are designed to carry much more oxygen than natural red blood cells. This plan could be very useful for patients suffering from anemia. These reciprocate nanobots would enclose a tank in which oxygen is held at a high pressure, sensors to determine the concentration of oxygen in the bloodstream, and a valve that releases oxygen when sensors determine that additional oxygen is needed.

3.8.3 Clottocyte nanobots: These robots function likewise to the platelets in our blood. Platelets stick jointly in a wound to form a clot, preventint blood flow. Depending on the size of the wound, significant blood loss can transpire before a clot is formed A system of clottocyte nanobots would store fibers until they encounter a wound. In such case, the nanobots would disperse their fibers, which would then come closer to create a clot in a fraction of time that platelets do.

3.8.4 Cellular repair nanobots: By working at the cellular level, such nanobots could prevent much of the damage caused by comparatively clumsy blades. The nature of the components being in the nano scale allows the researchers for the engineering of the mimicking of human beings and can repair the vital body parts, similar to the natural heal process. The construction of the various complex parts, which compose the robots have been possible due to nanorobotics, nanites, nanobots, nanoids or nanomites are some of the theoretical devices created with the knowledge of nanorobotics to do such operations.

3.8.5 Nanodentistry: One of the unique applications, whereby nanorobots aid in different processes involved in dentistry. They assist in inducing oral anaesthesia, desensitization of tooth, manipulation of the tissue for the re-allignment and straightening of the irregular set of teeth \, major tooth repair , the improvement of the teeth durability, generation of nanofiller and improvement of appearance of teeth, etc.

3.8.6 Treating arteriosclerosis: Can be applied in the therapeutics for atherosclerosis. Arteriosclerosis refers to a state ¸ where plaque builds along the walls of arteries as shown in figure 4. Nanorobots could conceivably treat the condition by cutting away the plaque, which would then enter into the bloodstream.

Nanorobots may treat circumstances like arteriosclerosis by physically cutting away the plaque along artery walls.

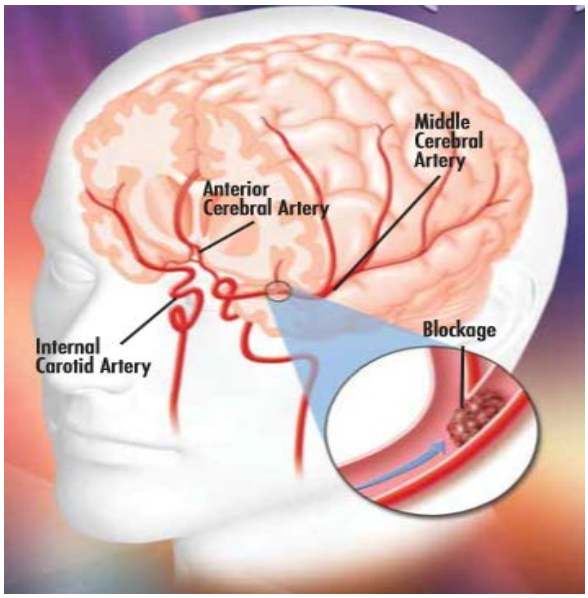


Figure 4. State of arteriosclerosis.

3.8.7 Cleaning and Breaking up blood clots: Blood clots can source complications ranging from muscle death to a stroke. Nanorobots could travel to a clot , break it up and clean as well . One particular kind of nanorobot is the clottocyte, or artificial platelet. The clottocyte carries a small mesh net that dissolves into a sticky membrane upon contact with blood plasma. This application is one of the most dangerous uses for nanorobots -- the robot must be able to remove the blockage without losing small pieces in the bloodstream, which could then travel away in the body and cause more problems. The robot must also be minute enough so that it doesn't block the flow of blood itself.

3.8.8 Parasite Removal: Nanorobots could wage micro-war on bacteria and small parasitic organisms inside a patient. It might take numerous nanorobots working together to destroy all the parasites.

3.8.9 Gout and kidney stones: Gout is a condition where the kidneys lose the ability to eradicate waste from the breakdown of fats from the bloodstream. This ruin sometimes crystallizes at points close to joints such as the ankles and knees . Patients who suffer from gout experience intense pain at these joints. A nanorobot could rupture up the crystalline structures at the joints, providing respite from the symptoms, though it would not be able to repeal the condition permanently. Kidney stones can be extremely painful , the larger the stone the more complex it is to bypass. Doctors break up large kidney stones using ultrasonic frequencies, but it's not always successful. A nanorobot could break up a kidney stones using a small laser.

Nanorobots would carry small ultrasonic signal generators to deliver frequencies directly to kidney stones.

Other related Applications

1) Self-directed assembly: Nanorobots would have self-assembled lipidic micelles, self-assembled monolayers, and vesicles, which pursue the Brownian revolution in the medical world.

2) DNA-directed assembly: Using part of DNA for assembling, which works on the self-assembly theory of complementary base pairing and has application in the DNA based rotary motors.

3) Protein-directed assembly: Genetically engineered chaperon proteins that help in the assembly of gold nanoparticles and semiconductor quantum dots into arrays in the nanoscale range. The Ratchet action protein based molecular motors have also found much application in biology.

4) Microbes and virus directed assembly: Includes various bacteria that are included into microelectromechanical systems (MEMS) and help in acting as pumps and living motors etc. Viral capsid shells have also found application in acting as scaffolds for the assembly of the nanoparticles.

5).Transmigration of the WBC: The inflammatory cells to can be healed to accelerate the healing process.

6). Drug delivery nanorobots: Known as 'pharmacytes' will be applied in future therapeutics related to cancer in chemotherapy for exact dose administration of the chemicals as well as in the anti-HIV-therapeutics.

7). Ancillary devices: For processing different chemical reactions in the injured organs.

8). Control and monitor of glucose: In diabetic patients.

9). Surgical nanorobots for nanomanipulation: In this the target site will be subjected to programming based on the guidance from a surgeon.

10). Gene therapy and DNA analysis: For different genetic diseases and disorders.

4. CONCLUSION

The future of bio robots is bright. We are at the dawn of a new era in which many disciplines atreams of technology will amalgamate which include robotics, mechanical, chemical ,biomedical engineering, physics , mathematics , chemistry and biology,. This is required so as to develop fully functional systems. Research considers that the nanobots will be able to act as antibodies for patients with weak immune systems, they will be able to move in the bloodstream searching for harmful bacteria and viruses and eliminating them before they cause the patient any harm. This will become extremely useful when tackling more common ailments otherwise treating them is complex and expensive. Nanobots will not only be able to help doctors and surgeons fight life threatening diseases but also more common bacterial and viral

cases. In this paper we have focused on the revolution of nanorobits, its functions, fabrication details along with the future research and product design avenues in the medicine field.

REFERENCES

[1]. Cavalcanti A, Freitas Robert A. Jr., Kretly Luiz C. Nanorobotics control design: a practical approach tutorial. *ASME 28th Biennial Mechanisms and Robotics Conference*, Salt Lake City Utah, USA, September 2004.

[2]. Zhang L, Abbott JJ, Dong L, Kratochvil BE, Bell D, Nelson BJ: Artificial bacterial flagella: fabrication and magnetic control. Appl. Physics Lett. 94 (064107), 1–3 (2009).

[3]. Ghosh A, Fischer P: Controlled propulsion of artificial magnetic nanostructured propellers. Nano Lett. 9(6), 2243–2245 (2009).

[4]. Cavalcanti A., "Assembly Automation with Evolutionary Nanorobots and Sensor-Based Control applied to Nanomedicine", IEEE Transactions on Nanotechnology, Vol. 2, no. 2, pp. 82-87, June 2003.

[5]. Venkatesan, M. ; Jolad, B. "Nanorobots in cancer treatment " , INTERACT 2010, Ladjal, H. ; Hanus, J.(L. Ferreira, A. 2009)

[6]. O. Ozcan and M. Sitti, "Modeling of Conductive Atomic Force Microscope Probes for Scanning Tunneling Microscope Operation," Micro/Nano-Letters, vol. 7, no. 4, pp. 329-333, April 2012.

[7]. Diller, C. Pawashe, S. Floyd, and M. Sitti, ''Assembly and Disassembly of Magnetic Mobile Micro-Robots towards Deterministic 2-D Reconfigurable Micro-Systems,'' International Journal of Robotics Research, vol 31, 1667-1680, 2011.