INTERNATIONAL JOURNAL of COMPUTERS COMMUNICATIONS & CONTROL

ISSN 1841-9836

A Bimonthly Journal
With Emphasis on the Integration of Three Technologies

Year: 2018 Volume: 13 Issue: 6 Month: December

This journal is a member of, and subscribes to the principles of, the Committee on Publication Ethics (COPE).

C O P E
Member since 2012
JM08090

http://univagora.ro/jour/index.php/ijccc/

CCC Publications

BRIEF DESCRIPTION OF JOURNAL

Publication Name: International Journal of Computers Communications & Control.
Acronym: IJCCC; **Starting year of IJCCC:** 2006.
ISO: Int. J. Comput. Commun. Control; **JCR Abbrev:** INT J COMPUT COMMUN.
International Standard Serial Number: ISSN 1841-9836.
Publisher: CCC Publications - Agora University of Oradea.
Publication frequency: Bimonthly: Issue 1 (February); Issue 2 (April); Issue 3 (June); Issue 4 (August); Issue 5 (October); Issue 6 (December).
Founders of IJCCC: Ioan DZITAC, Florin Gheorghe FILIP and Misu-Jan MANOLESCU.

Indexing/Coverage:

- Since 2006, Vol. 1 (S), IJCCC is covered by Clarivate Analytics and is indexed in ISI Web of Science/Knowledge: Science Citation Index Expanded.

  2018 Journal Citation Reports® Science Edition (Clarivate Analytics, 2017):
  Subject Category: (1) Automation & Control Systems: Q4(2009, 2011, 2012, 2013, 2014, 2015), Q3(2010, 2016, 2017); (2) Computer Science, Information Systems: Q4(2009, 2010, 2011, 2012, 2015), Q3(2013, 2014, 2016, 2017).

  Impact Factor/3 years in JCR: 0.373(2009), 0.650 (2010), 0.438(2011); 0.441(2012), 0.694(2013), 0.746(2014), 0.627(2015), 1.374(2016) **1.29 (2017)**.
  Impact Factor/5 years in JCR: 0.436(2012), 0.622(2013), 0.739(2014), 0.635(2015), 1.193(2016), **1.179(2017)**.

- Since 2008 IJCCC is indexed by Scopus: **CiteScore 2017 = 1.04**.
  Subject Category:
  (1) Computational Theory and Mathematics: Q4(2009, 2010, 2012, 2015), Q3(2011, 2013, 2014, 2016, 2017);
  (2) Computer Networks and Communications: Q4(2009), Q3(2010, 2012, 2013, 2015), Q2(2011, 2014, 2016, 2017);
  (3) Computer Science Applications: Q4(2009), Q3(2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017).

  SJR: 0.178(2009), 0.339(2010), 0.369(2011), 0.292(2012), 0.378(2013), 0.420(2014), 0.263(2015), 0.319(2016), 0.326 (2017).

- Since 2007, 2(1), IJCCC is indexed in EBSCO.

Focus & Scope: International Journal of Computers Communications & Control is directed to the international communities of scientific researchers in computers, communications and control, from the universities, research units and industry. To differentiate from other similar journals, the editorial policy of IJCCC encourages the submission of original scientific papers that focus on the integration of the 3 "C" (Computing, Communications, Control).

In particular, the following topics are expected to be addressed by authors:

(1) Integrated solutions in computer-based control and communications;

(2) Computational intelligence methods & Soft computing (with particular emphasis on fuzzy logic-based methods, computing with words, ANN, evolutionary computing, collective/swarm intelligence, membrane computing, quantum computing);

(3) Advanced decision support systems (with particular emphasis on the usage of combined solvers and/or web technologies).

EDITORIAL STAFF OF IJCCC

EDITORIAL ADDRESS:

Agora University, Cercetare Dezvoltare Agora, Tineretului 8, 410526 Oradea, Bihor, Romania,
Tel./ Fax: +40 359101032, E-mail: ijccc@univagora.ro, rd.agora@univagora.ro
URL: http://univagora.ro/jour/index.php/ijccc/

EDITORIAL BOARD OF IJCCC

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, Volume 13, Issue 6, December 2018.

Contents

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(6), 915-926, December 2018.

Defense Scheme to Protect IoT from Cyber Attacks using AI Principles

T. Ahamed

Tariq Ahamed Ahanger*
College of Computer Engineering & Sciences,
Prince Sattam Bin Abdulaziz University, KSA
*Corresponding author: t.ahanger@psau.edu.sa

Abstract: Even in its infancy, the internet of things (IoT) has enticed most of the modern industrial areas like smart cities, automobiles, medical technology. Since IoT connects everything together, it is vulnerable to a variety of devastating intrusion attacks. Being the internet of different devices makes it easy for attackers to launch their attacks. Thus, to combat all these attacks, an attack analysis is presented in this article using the basic principles of Artificial Neural Networks. Internet packet traces are used to train to the supervised ANN (Multilevel Perceptron) and evaluated after the training to decline the DDoS Attacks. This research article mainly focuses on the categorization of traffic patterns into legitimate traffic and attack traffic patterns in IoT network. The ANN processes are evaluated and tested in a simulated IoT network. The experimental results show a greater accuracy in detection of various DDoS attacks.

Keywords: ANN, IoT, DDoS, Security, IDS, AI.

1 Introduction

In the current world more objects are associated with the Internet than the people. And this difference will pursue to grow further, as the objects increases in their capability to directly interact with the Internet or evolves into the physical objects of data which can be accessible through the Internet systems. This scenario approaching towards greater independence in object interaction with the Internet is generally interpreted as the (IoT) Internet of Things [4,26]. The greater number of devices which avails the internet services is growing faster and to have them connected every time by wire or otherwise will establish a mighty source of data information at anyone's finger tips [17]. To enable communication between smart machines is an advanced technology, but the technologies which compose the IoT are not fresh for us. IoT, as understood by its name, is the method of converting data to any virtual-platform on current internet infra, which is retrieved from diverse class of things [3]. The main concept of IoT is to grant independent exchange of important information between unseen embedded distinctly identifiable devices in the real world around us, overwhelmed by the dominant technologies like RFID (Radio-Frequency Identification) and WSNs (Wireless Sensor Networks) sensed by the sensor nodes and processed further for decision building [1], in order to perform an automated action.

2 Security threats

The complexity in the nature of IoT security rotates around the reality that, since, it is a great challenge to combine several technologies into one; the system tries to connect devices securely which have limited computation capability, storage, and power [2]. Few of the devices utilized by IoT can hold only a little basic mechanism of security measures, some of which are not capable to maintain the confidentiality and integrity of the users' information data [19]. Furthermore, the devices-for example, RFIDs, and sensors has inadequate user interfaces, such as an On-Off button or status signal, hence represents a limited psychological visual for the owners when it approaches towards trusting these type of devices [13]. Presently, privacy is considered as a major concern; which slows down the advancement of many upcoming technologies. Moreover, it has been proved that the technology which doesn't provide enough trust and exposes the individual's identity diminishes [7]; in the recent past, many new technologies failed to maintain adequate privacy and security mechanisms, which causes pain and tremendous sufferings to the distressed. For the sake of gaining trust of a common man in the IoT, the technologists need to make sure the same kind of failures with regard to security and privacy do not appear in the system, by safeguarding the required mechanisms to ensure such things existed from the beginning.

3 Sources of threats

Subsequently, discussing the IoT features and how they can be used in several scenarios, it is now time to discuss and to identify the potential threats confronting the communication mechanism in IoT. There are three primary entities which poses threats to the privacy and security in IoT:

(i) Malicious User: It is the possessor of the IoT-device with power to carry out attacks to acquire the secrets of the device manufacturer, and also to acquire access to secret functionality. The Malicious user reveals the shortcomings in the system to retrieve information, selling secrets to 3 rd parties, even launches an attack on the systems [12].

(ii) Bad Manufacturer: It is the builder of the device with the capability to explore the underlying technology to retrieve the information of the IoT devices, or users. Aforementioned, a manufacturer can intendedly introduce holes in the security design which can be exploited subsequently to access the user's secret data and also revealing it to 3rd parties. Similarly, the manufacturing of badly secured things results in the compromise of their users' privacy. Adding to that, in the context of IoT where different devices connect one another, a device manufacturer just to harm the reputation of their competitors; can attack their devices [6].

(iii) External Adversary: It is an external entity that is not considered as a member of the system and has no authorization to access it. An adversary aims to gather the information about the users' of the system with the malicious intentions for example, creating financial losses and subverting the user's reliability. It also, causes flaw in the system by maneuvering the data sensed and transmitted [11].

4 Classes of attacks

To understand the risks attached, classification of attacks is essential to every system. We opted to focus on many categories to determine threats.

- Device Tampering: As we know IoT devices are smaller devices which are integrated in many other systems, for example; light, switches, TVs, cars, ovens, and many more. Few of the IoT devices remains unattended most of the time, therefore, they could be stolen easily irrespective being noticed by anyone. If a device got into the wrong hands, several types of attacks can be executed like software manipulation, hardware tampering, and secret stealing [20]. It's essential to indicate that an attacker could tamper with the vulnerable device and utilize it to inject fake data into the system, utilize the device for attacking purpose or deviate it to its expected functionality.

- Information Disclosure: It's an act of disclosing the information to an object which lacks the permission to access it. This comprises targeted attack, accidental exposure, and correlation or inference [18]. An adversary can retrieve information from the network links by eavesdropping, physically accessing the device.
- Privacy Breach: Contradictory to Information Disclosure, an attacker doesn't essentially require access to the secret information of the user to learn about him. The attacker could ascertain confidential information from several other sources like traffic analysis and meta data [5].
- Denial-of-Service: DoS is associated to the characteristic of being not accessible when an authorized user requests. The underlying system should be capable enough to continue operating even though some unintended action is being carried out by some adversary. DoS attacks can be carried out by device stealing, disrupting the communication links, manipulating the software [14].
- Spoofing: It is used to steal the credentials which belong to others so as to gain access to not accessible services. These credentials could be retrieved form a device directly, eavesdropping a channel of communication, or phishing [15].

5 Intrusion detection system

Any kind of unapproved or unauthorized activities in a network or a system are called intrusions. An IDS (Intrusion Detection System) is a group of the tools, mechanism, resource to identify, assessment which describes intrusions [8]. Intrusion detection is usually a part of a comprehensive protection system which is installed in a device or around a system and it can't be taken as a stand-alone measure of protection. Intrusion can be defined as: "any type of activities which tries to manipulate the truth, secrecy, or the resource availability" [9]. In the vice-versa intrusion prevention techniques are referred to as the first line of defense against these intrusions. Nevertheless, as in any type of security system, total prevention of intrusion is impossible. The node compromise and intrusion heads to secret information like security keys being disclosed to the intruders in the system, which results collapse of the security mechanism [10]. Consequently, IDSs are developed and designed to make intrusions public, before disclosing the system resources which are secured. IDSs are consistently acknowledged as a second line of defense from the view point of security. IDSs are considered as web equivalent of the robbers' alarms that are being utilized in physical security systems currently [16]. The normal functional requirement of IDSs are; "high true positive rate, measured as the percentage of anomalies detected, and low false positive rate, measured as the percentage of normalcy deviations detected as anomalies".

According to architecture, IDS can be divided into two groups:

Network Based IDS: TheIDS which is network based is responsible for the protection of the whole network environment from any type of intrusions. This type of IDS architecture requires comprehensive knowledge of the system status and also monitors the different components of the network as well as the transactions which carry out between them [11]. A network IDS watches the actions on an entire network and carries the traffic analysis for possible security breaches or threats. Agent technology performs a primary role in this type of IDS architecture.

Host Based IDS: The IDS which is host based watches all the ongoing activities on an individual information system host. It makes sure that not a single security policy of information system is being disrupted [12]. The host based IDS are installed only on a single terminal/host and are given the responsibility of observing the status of that specific host(or server) only.

6 IDS patterns

The primary purpose of developing an Intrusion Detection system is to detect and identify the potential threats efficiently. There IDS system can be classified into two categories:

Misuse detection: or signature-based detection. The behavior of the system is compared with previously known types of attack patterns (or signatures). Those action patterns which may present a security threat should be described and stored in the system. Afterwards, the misuse detection technique attempts to recognize any type of bad behavior with respect to these stored patterns in the system [18].

Anomaly detection: targets on normal behaviors, instead of attack behaviors. Firstly, these types of systems define what comprises a normal behavior which is normally carried out by an automated training and then intrusion activities are flagged that differ from this normal behavior by a specified threshold [22].

7 Intrusion detection techniques

There is a diverse class of IDS methods which are established on different formats, compositions and schemes. Following are the description of those frequently used techniques:

- Evolutionary Algorithm: The algo produces an application path that contributes standard operational models. This algorithm is designed to detect standard operational behavior, attempted intrusion, and error state by categorizing archetype which depends upon Non-Identical conditions [13].

- Rule Based: This technique compares data against signatures with state transition analysis. Every data packet is practiced to FSM (Finite State Machine) and follows transitions till the ultimate state reaches, resulted in the detection an attack [8, 10].

- Statistical Analysis: This technique involves the comparison of present set of data pattern with the predefined set of basic criteria. The normal data behavior is compared with the deviations over a temporal period. In Anomaly Detection system this technique is used [21].

- Protocol Verification: This technique is based upon extensively checking the protocol behavior and their fields in comparison with the pre- defined set of standards. The data which violates the defined standards is considered to be malicious. This approach has good success rate in lucrative systems but the flaws are producing erroneous positives for undefined protocols [10].

8 Artificial neural network (ANN)

ANN neurons are utilized to model complicated hypotheses. Complexity of the hypotheses depends upon the number of neurons involved. The hypotheses evaluation is carried out by the setup of input nodes in the feedback process and the propagation of event data across the network towards output nodes where it is categorized as legitimate or suspicious [3]. Gradient descents are used at this stage to thrust the error back from the output node across to the network using BPA(Back Propagation Algorithm) to scan the hidden nodes for error measurement. Thus the cost of function in terms of gradient descent can be calculated. To assimilate the structure and sequence developed in the system, neural network systems go through the training phase. A typical structure of an ANN neuron is shown in Fig. 1.

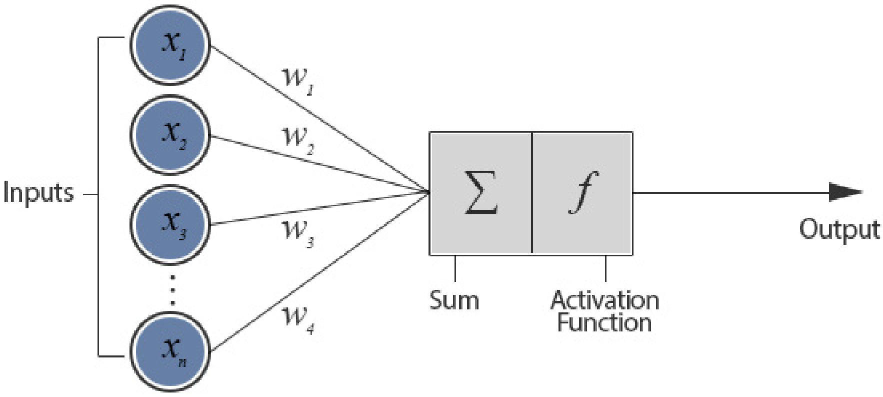


Figure 1: ANN neuron

9 ANN learning process

Two types of learning procedures for ANN's are as follows:

- Unsupervised Learning Procedure: The ANN in this learning has an input

  $$g = \{a_i\}_{i=1}^{N}$$

  i.e., unlabeled data set; need to search certain patterns in that data.

  SOM is a category of Artificial Neural Network which is trained by unsupervised learning method to generate less dimensional, discrete representation input space of samples of training called map.

- Supervised Learning Procedure: In this type of learning, the ANN is presented with labelled set of training data which learns the mapping from data inputs a to outputs b, provided a labelled input-output sets of pairs,

  $$g = \{(a_i, b_i)\}_{i=1}^{N}$$

  where g is called the training set and N is the total number of training samples. It is presumed that b_i is a group variable from an infinite set

  $$b_i \in \{1, \dots, X\}$$

The MLP (Multi-layer Perceptron) is a category of artificial neural network trained by supervised learning procedure. The MLP is utilized for the detection of intrusions which were established on an offline analysis approach. Also, in a diverse outlook, it was used to detect intrusion in a data on a network by comparing it with the SOM Self-Organizing Maps.

10 MLP architecture

A three layer MLP architecture with feed-forward ANN is shown in Fig. 2. The network holds a function of unipolar transfer sigmoid in every output and hidden layer neurons. Also, an algorithm with stochastic learning and a function of mean square error is utilized. The input nodes which represent neural network with labels $x_1, \ldots, x_3$ have been implemented, and +1 as bias units.

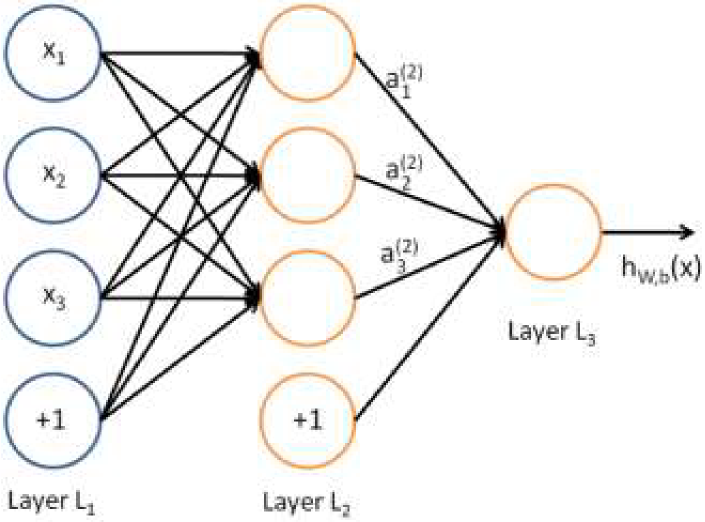


Figure 2: Feed-forward neural network

The ANN architecture has total of three input units (L_1), three hidden units (L_2) and finally one output (L_3) composed of the three layers of neural network.

Feed forward Learning Algorithm: The feed forward algorithm is the simplest form of devised artificial neural network. The information moves in only forward direction, which starts from the input layer, the hidden layer and finally to the output layer respectively. The algorithm states as follows; Let V_l is the number of total elements which excludes the bias elements. Hence, the network parameters are

$$\{W, b\} = \{W^1, b^1, W^2, b^2\}$$

where W_{ij}^{lr} represents the combined parameter connected with the element j in layer lr and element i in $lr + 1$. Likewise, b_i^{lr} is the bias connected with I unit in the layer $lr + 1$. Therefore above mentioned elements and functions reflect $W^1 \in O^{3*3}$ and also, $W^2 \in O^{3*1}$. Suppose, a_i^1 represent the output element in layer lr.

For $lr = 1$, we just supposed that $a_i^1 = x_i$ represent the ith input. Thus, the ANN model will derive hypotheses $h_{(w,b)}(x)$ which give a real number as output. Thus we can represent the above model in a mathematical form as:

$$a_1^{(2)} = f\{W_{11}^1 x_1 + W_{12}^1 x_2 + W_{13}^1 x_3 + b_1^1\} \tag{1}$$

$$a_2^{(2)} = f\{W_{21}^1 x_1 + W_{22}^1 x_2 + W_{23}^1 x_3 + b_2^1\} \tag{2}$$

$$a_3^{(2)} = f\{W_{31}^1 x_1 + W_{32}^1 x_2 + W_{33}^1 x_3 + b_3^1\} \tag{3}$$

$$h_{(w,b)}(x) = f\{W_{11}^2 a_1^{(2)} + W_{12}^2 a_2^{(2)} + W_{13}^2 a_3^{(2)} + b_1^2\} \tag{4}$$

Eq. 4, weighted total sum of the inputs from I in lr is the feedforward algorithm.

Backward Learning Algorithm: The process of learning in involves 4 stages, which are as follows:

- For all the layers in the neural network, the feed forward algorithm calculates the activation.
- The output from layer L_3 computes the measure of error in the output:

$$\begin{aligned}\Delta_i^{(L_3)} &= \frac{\Delta}{\Delta_y^{(L_3)}} \frac{1}{2} \|z \cdot h_{(w,b)}(x)\|^2 \\ &= -\{z_i - a_i^{(L_3)}\} \cdot f\left(y_i^{(L_3)}\right) \end{aligned} \tag{5}$$

  Here, $a_i^{(3)} = f\left(y_i^{(L_3)}\right)$ is the sigmoid function.

- To calculate the errors in $lr = 2$, $lr = 3$ for every node I in layer lr:

$$\Delta_i^{(lr)} \left(\sum_{k=1}^{V_t+1} W_{ki}^{(lr)} \Delta_k^{(l+1)} \right) f'\left(y_i^{(L_3)}\right) \tag{6}$$

- Lastly, the intended partial derivatives are computed as:

$$\frac{\Delta_i^{(lr)}}{\Delta W_{kj}^{(l)}} M(W, b; z, x) = a_j^{(l)} \Delta_k^{(l+1)}$$

$$\frac{\Delta}{\Delta b_k^{(l)}} M(W, b; z, x) = \Delta_k^{(l+1)}$$

11 Proposed system and evaluation

We have used eight node sensors to compose an IoT network. Seven client nodes and one server relay node for data analytical purpose. Network tap is used to capture traffic to avoid any kind of hindrance or change in the live traffic. The sensor node sends data towards server node and server node acknowledges by sending receive data reply that is based on data itself. This process enables the sensor nodes as shown in Fig. 3, to acclimatize their behavior and respond to the ongoing phenomenon.

In our research setup, we have an external intruder who attacks the IoT network as shown in Fig. 4. Server node is the only target of target of attackers because the server node analyzes, keeps the record and responds to the sensor nodes. The attacker launches the DoS attack by

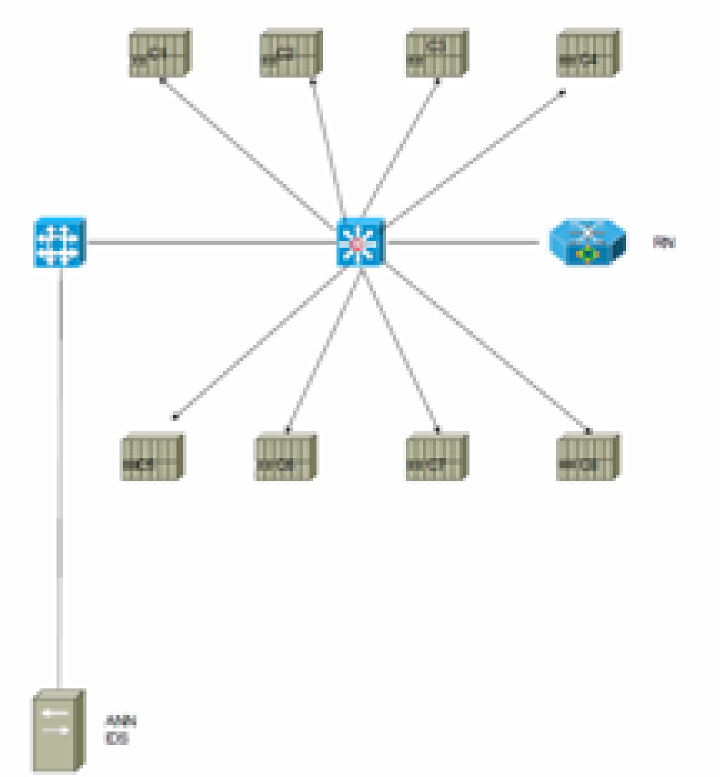


Figure 3: Normal condition

sending 10^7 packets towards the target node from a single host and the same amount of attack traffic was launched from four hosts to launch DDoS towards the target node. We used a custom C script to craft UDP packets to use them as attack packets. Therefore, the server node comes to a halt and do not respond. The sensor nodes are not able to accommodate their changed behavior and finally cause the halt in the monitored system.

Therefore the detection of these attacks at the right time is very important to allow the uninterrupted working of sensor network and assure the reliability of the network.

Table 1: Parameters for Training.

No. of sample patters used for training	2575
No. of sample patters used for Validation	499
No. of sample patters used for test	499

Table 2: Amount of Pattern Traffic Used For Categorization.

Traffic	Size	%
DDoS	2294	63.98
DoS	2294	63.98
Normal	1279	36.02

In this section we will evaluate the performance of designed Artificial Neural Network intrusion detection method as discussed in the previous section. We trained the network with the

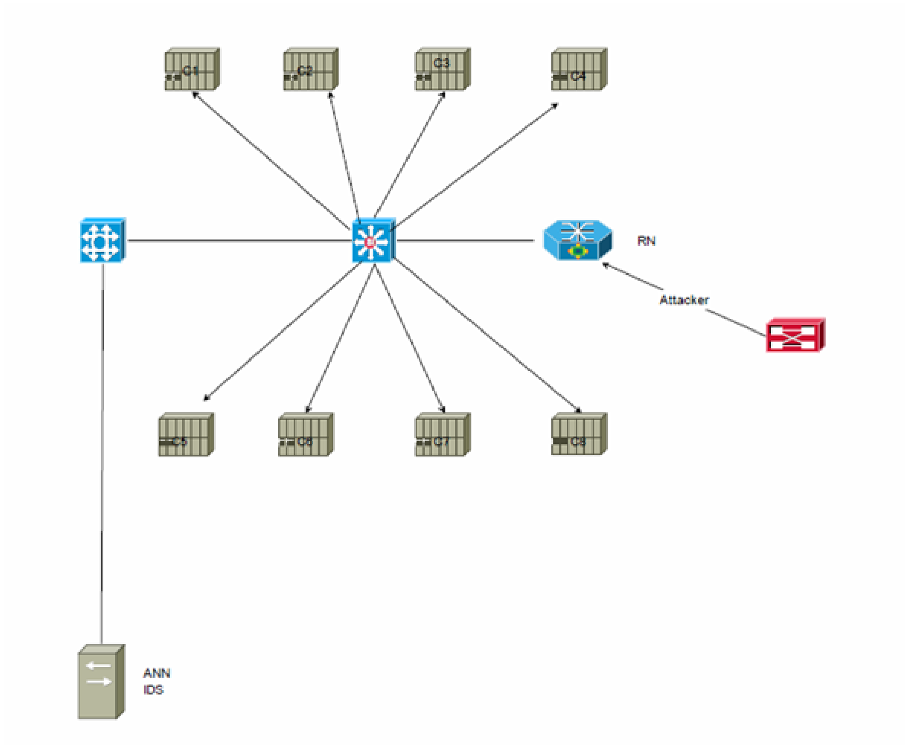


Figure 4: Attack situation

following parameters as shown in Table 1 and the amount of sample patterns used for categorization is shown in Table 2.

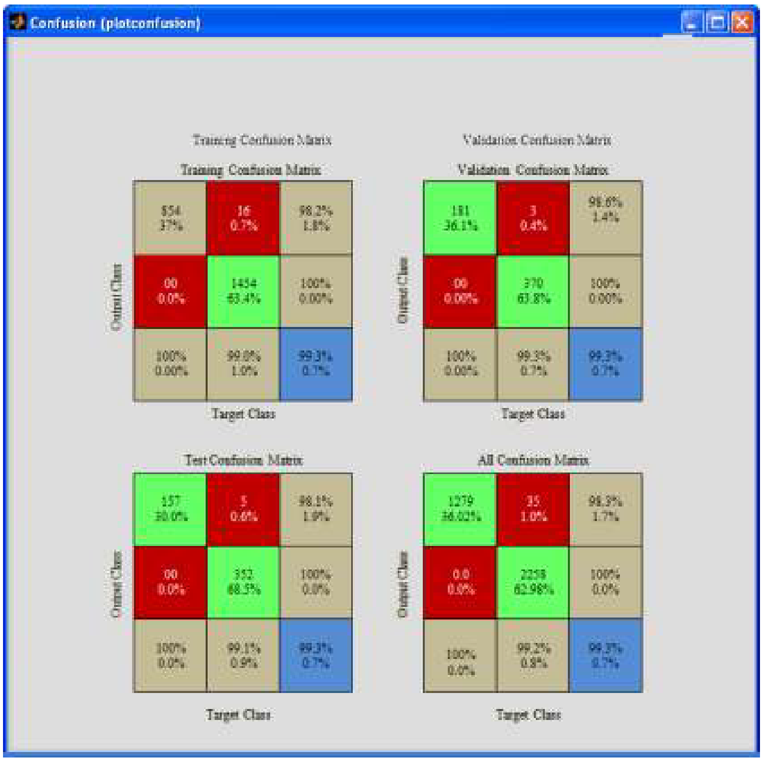
Neural network confusion matrix is shown in Fig. 5, is used to plot

- Training Set
- Testing Set
- Validation Set
- The Final All Confusion Matrix

The output of the system is of two types:

1. True Positive
2. False Positive

The attack traffic that is correctly categorized is measured as True positive as shown in green block and the correctly categorized normal traffic is measured as False positive as shown in red block. After all the evaluation and tests as shown in 4th final confusion matrix in Fig. 5, our proposed system showed the detection accuracy greater than 99% in the categorization of network traffic. Our method proved that the artificial neural network algorithm used is capable of detecting DDoS and Dos attack traffic during the flow of genuine IoT network traffic. It also improves the stability and reliability of the IoT network by signaling the response system at the right time to avoid the network disruptions, thus improving and enhancing the performance of the IoT network.



Training Confusion Matrix

Training Confusion Matrix

554 37%	16 0.7%	98.2% 1.8%
00 0.0%	1454 63.4%	100% 0.00%
100% 0.00%	99.0% 1.0%	99.3% 0.7%

Validation Confusion Matrix

Validation Confusion Matrix

181 36.1%	3 0.4%	98.6% 1.4%
00 0.00%	370 63.8%	100% 0.00%
100% 0.00%	99.3% 0.7%	99.3% 0.7%

Test Confusion Matrix

157 30.0%	5 0.6%	98.1% 1.9%
00 0.0%	352 68.5%	100% 0.0%
100% 0.0%	99.1% 0.9%	99.3% 0.7%

All Confusion Matrix

1279 36.02%	35 1.0%	98.3% 1.7%
0.0 0.0%	2258 62.98%	100% 0.0%
100% 0.0%	99.2% 0.8%	99.3% 0.7%

Figure 5: Confusion matrix

12 Conclusion

A DDoS detection method is presented in this research article using ANN for IoT network. The detection method is mainly based on categorization of legitimate traffic patterns and attack traffic patterns. The proposed system is simulated and tested in an organized simulated IoT network and the obtained results proved more than 99% detection accuracy. The system successfully identified the attack traffic and performed well in true and false negative accuracy. In the future, the system will be trained with latest threat patterns and will be tested to check its reliability with the modern world technology.

Acknowledgement

This research was funded and conducted at Prince Sattam bin Abdulaziz University, Alkharj, Saudi Arabia during the academic year 2017–2018 under research number 2017/01/8029.

Bibliography

[1] Ahamad, T. (2016). Detection and Defense Against Packet Drop Attack in MANET, *International Journal of Advanced Computer Science and Applications (IJACSA)*, 7(2), 2016.

[2] Ahamad, T.; Aljumah, A. (2015). Detection and defense mechanism against DDoS in MANET, *Indian Journal of Science and Technology*, 8(33), 2015.

[3] Alan, S.; Overill, R.E.; Radzik, T. (2016). Detection of known and unknown DDoS attacks using Artificial Neural Networks, *Neurocomputing*, 172, 385–393, 2016.

[4] Aldaej, A.; Ahamad, T. (2016). AAODV (Aggrandized Ad Hoc on Demand Vector): A Detection and Prevention Technique for Manets, *International Journal of Advanced Computer Science and Applications(IJACSA)*, 7(10), 2016.

[5] Aljumah, A.; Ahamad, T. (2016). Black Hole and Mobile Ad Hoc Network (MANET): A Simple Logical Solution, In: *11th International Conference on Cyber Warfare and Security: ICCWS2016*, 1-9, 2016.

[6] Aljumah, A.; Ahamad, T. (2016). A Novel Approach for Detecting DDoS using Artificial Neural Networks, *International Journal of Computer Science and Network Security*, 16(12), 132-138, 2016.

[7] Ahmed, E.; Yaqoob, I.; Gani, A.; Imran, M.; Guizani, M. (2016). Internet-of-things-based smart environments: state of the art, taxonomy, and open research challenges, *IEEE Wireless Communications*, 23(5), 10–16, 2016.

[8] Alrajeh, N. A.; Khan, S.; Shams, B. (2013). Intrusion detection systems in wireless sensor networks: a review, *International Journal of Distributed Sensor Networks*, 1-7, 2013.

[9] Alshehri, A.; Sandhu, R. (2016). Access Control Models for Cloud-Enabled Internet of Things: A Proposed Architecture and Research Agenda. In: *Collaboration and Internet Computing (CIC), 2016 IEEE 2nd International Conference on*, 530–538, 2016.

[10] Bucerzan, D.; Cayrel, P.-L.; Dragoi, V.; Richmond, T. (2017). Improved Timing Attacks against the Secret Permutation in the McEliece PKC, *International Journal of Computers Communications & Control*, 12(1), 7-25, 2017.

[11] Butun, I.; Morgera, S. D.; Sankar, R. (2014). A survey of intrusion detection systems in wireless sensor networks, *IEEE Communications Surveys & Tutorials*, 16(1), 266–282, 2014.

[12] Creech, G.; Hu, J. (2014). A semantic approach to host-based intrusion detection systems using contiguousand discontiguous system call patterns, *IEEE Transactions on Computers*, 63(4), 807–819, 2014.

[13] Elhag, S.; Ferná ndez, A.; Bawakid, A.; Alshomrani, S.; Herrera, F. (2015). On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on Intrusion Detection Systems, *Expert Systems with Applications*, 42(1), 193–202, 2015.

[14] Elkhodr, M.; Shahrestani, S.; Cheung, H. (2016). The internet of things: new interoperability, management and security challenges. *arXiv preprint arXiv:1604.04824*.

[15] Han, G.; Shu, L.; Chan, S.; Hu, J. (2016). Security and privacy in Internet of things: methods, architectures, and solutions. *Security and Communication Networks*, 9(15), 2641–2642, 2016.

[16] Gong, W. (2016). *The Internet of Things (IoT): what is the potential of the internet of things (IoT) as a marketing tool?*, Bachelor's Thesis, University of Twente, 2016.

[17] Gunasekaran, A.; Subramanian, N.; Tiwari, M.K. (2016). Information technology governance in Internet of Things supply chain networks, *Industrial Management & Data Systems*, 116.7, 2016.

[18] Kim, G.; Lee, S.; Kim, S. (2014). A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. *Expert Systems with Applications*, 41(4), 1690–1700, 2014.

[19] Madakam, S.; Date, H. (2016). Security Mechanisms for Connectivity of Smart Devices in the Internet of Things, In *Connectivity Frameworks for Smart Devices* (pp. 23–41). Springer International Publishing.

[20] McKelvey, B.; Tanriverdi, H.; Yoo, Y. (2016). Complexity and Information Systems Research in the Emerging Digital World. *MIS Quarterly.*

[21] Mitchell, R.; Chen, I. R. (2014). A survey of intrusion detection techniques for cyber-physical systems. *ACM Computing Surveys (CSUR)*, 46(4), 55, 2014.

[22] Moshtaghi, M.; Erfani, S. M.; Leckie, C.; Bezdek, J. C. (2017). Exponentially Weighted Ellipsoidal Model for Anomaly Detection. *International Journal of Intelligent Systems*, 32(9), 881-899, 2017.

[23] Niu, J.; Jin, Y.; Lee, A.J.; Sandhu, R.; Xu, W.; Zhang, X. (2016). Panel Security and Privacy in the Age of Internet of Things: Opportunities and Challenges. In *Proceedings of the 21st ACM on Symposium on Access Control Models and Technologies*, 49–50, 2016.

[24] Samaila, M. G.; Neto, M.; Fernandes, D. A.; Freire, M. M.; Iná cio, P. R. (2017). Security Challenges of the Internet of Things, *Beyond the Internet of Things*, 53–82, 2017.

[25] Singh, M.; Rajan, M. A.; Shivraj, V. L.; Balamuralidhar, P. (2015). Secure mqtt for internet of things (iot). In: *Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on, IEEE*, 746–751, 2015.

[26] Tariq, U.; Aldaej A. (2018). Outlook of Coordinated Transmission Control in 5G Networks for IoTs, *International Journal of Computers Communications & Control*, 13(2), 280-293, 2018.

[27] Tellez, M.; El-Tawab, S.; Heydari, H. M. (2016). Improving the security of wireless sensor networks in an IoT environmental monitoring system. In *Systems and Information Engineering Design Symposium (SIEDS), 2016 IEEE*, 72–77, 2016.

[28] Xu, K.; Qu, Y.; Yang, K. (2016). A tutorial on the internet of things: from a heterogeneous network integration perspective, *IEEE Network*, 30(2), 102–108, 2016.

[29] Zheng, Z.; Xie, S.; Dai, H. N.; Wang, H. (2016). Blockchain Challenges and Opportunities: A Survey, *Int. J. Web and Grid Services*, 14(4), 2018.

[30] Zhao, S.; Cheng, B.; Yu, L.; Hou, S. L.; Zhang, Y.; Chen, J. L. (2016). Internet of Things Service Provisioning Platform for Cross-Application Cooperation, *International Journal of Web Services Research (IJWSR)*, 13(1), 1–22, 2016.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(6), 927-937, December 2018.

Model Predictive Control of Stochastic Linear Systems with Probability Constraints

C.F. Caruntu, C.C. Velandia-Cardenas, X. Liu, A.N. Vargas

Constantin F. Caruntu
Department of Automatic Control and Applied Informatics,
Gheorghe Asachi Technical University of Iasi, Str. Prof. D. Mangeron no. 27, Iasi, Romania.

Cristian C. Velandia-Cardenas
Research Group MEM, Faculty of Electronic Engineering,
Universidad Santo Tomás, Cra. 9 no. 51-11, Bogotá, Colombia.

Xinghua Liu
Department of Electrical Engineering,
School of Automation and Information Engineering,
Xi'an University of Technology, Xian 710048, China.

Alessandro N. Vargas*
Universidade Tecnológica Federal do Paraná, UTFPR,
Av. Alberto Carazzai 1640, 86300-000 Cornelio Procópio-PR, Brazil.
*Corresponding author: avargas@utfpr.edu.br

Abstract: This paper presents a strategy for computing model predictive control of linear Gaussian noise systems with probability constraints. As usual, constraints are taken on the system state and control input. The novelty relies on setting bounds on the underlying cumulative probability distribution, and showing that the model predictive control can be computed in an efficient manner through these novel bounds—an application confirms this assertion. Indeed real-time experiments were carried out to control a direct current (DC) motor. The corresponding data show the effectiveness and usefulness of the approach.

Keywords: probability constraints, stochastic systems, linear systems, control.

1 Introduction

Engineering applications show that constraints are not always met with probability one [4]. It is then reasonable to account constraints, together with performance indicators, under certain probability levels, e.g. [9]—this setup is referred to as *model predictive control* (MPC) with probability constraints, as detailed next.

Probability constraints on linear systems constitute a research topic that has attracted attention in recent years, in particular when it is combined with the MPC method, see for instance [2, 5, 10, 15, 20, 25], just to name a few. In this scenario, the model accounts additive disturbances [5, 15, 20, 25], multiplicative disturbances [2], and systems with mixed features; namely, systems with known:

1. probability distribution of the system output [15, 20];
2. Gaussian distribution and general probability distribution of the disturbance [5];
3. observable probability distribution of the disturbance at each time step [2];
4. covariance and mean of the inputs and outputs [10].

In particular, probabilistic constraints on inputs [5, 10, 11], states [2, 5, 10, 11, 25], and outputs [15, 20] were transformed into deterministic constraints on the process input.

To handle probability constraints is a task that the literature suggests the evaluation of element-wise probability constraints [2, 10, 11, 15, 16, 20, 25]; here, we follow the literature with a minor adaptation, that is, the element-wise evaluation yields nonlinear equations, difficult to compute.

To overcome this difficulty, we propose an upper bound for the underlying nonlinear equations. As a byproduct, we obtain linear matrix inequalities—computing linear matrix inequalities increases efficiency on solving optimization problems [4, Ch. 1].

The main contribution of this paper is to convert nonlinear equations, drawn from probability constraints, into linear matrix inequalities useful for real-time implementation. This assertion is illustrated in practice through the MPC applied in a direct current (DC) motor—experimental data are shown in Section 4. These findings illustrate the practical benefits of our approach.

The paper is organized as follows. Section 2 states the problem to be solved and the derived solution. Section 4 provides a performance evaluation of the proposed MPC with probability constraints based on the experimental results obtained on a direct current motor. Finally, concluding remarks are given in Section 5.

Notation and basic definitions.

$\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{Z}$ and $\mathbb{Z}_+$ are the real, non-negative real, integer and non-negative integer numbers. $\mathbb{Z}_{\geq c_1}$ and $\mathbb{Z}_{(c_1,c_2]}$ denote the sets $\{k \in \mathbb{Z}_+ \mid k \geq c_1\}$ and $\{k \in \mathbb{Z}_+ \mid c_1 < k \leq c_2\}$, respectively, for some $c_1, c_2 \in \mathbb{Z}_+$. $|| \cdot ||_\infty$ denotes the infinity norm; given $x \in \mathbb{R}^n$ and $c \in \mathbb{R}$, by definition of the infinity norm, for $||x||_\infty \leq c$ to be satisfied, it is necessary and sufficient that $\pm[x]_j \leq c$ for all $j \in \mathbb{Z}_{[1,n]}$. I_n is the identity matrix of order n and M' denotes the transpose of M. $P[A]$ denotes the probability of event A and $p(x)$ denotes the probability distribution function of random variable x. $\bar{x}$ denotes the mean of the random variable x and S_x denotes its covariance.

2 Problem statement

For some fixed and filtered probability space $(\Omega, \mathcal{F}, P)$, consider the following discrete-time linear time invariant system

$$x_{k+1} = Ax_k + Bu_k + Hw_k, \quad k \in \mathbb{Z}_+, \tag{1}$$

where x_k denotes an n-dimensional system state, u_k represents an m-dimensional control input, and w_k represents a q-dimensional noise or disturbance driven by an independent and identically distributed (i.i.d.) Gaussian process. It is assumed that the initial state x_0 is a Gaussian random variable uncorrelated with w_k.

Although it has been intensively studied over the last decades, (1) presents so far some interesting challenges. For instance, controlling (1) under probabilistic constraints remains an open problem [9].

The main contribution of this paper lies in showing a condition for controlling the system (1) under probabilistic constraints. We revisit some results from the literature, and show how to convert them into linear matrix inequalities, a useful scheme for numerical evaluations.

To turn our contribution clear, we present some notation borrowed from [3]. Stacking the elements of (1) for N steps in the form

$$\mathbb{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \quad \mathbb{U} = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}, \quad \mathbb{W} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix}, \tag{2}$$

we obtain the Nn-dimensional vector representation

$$\mathbb{X} = G_{xx}x_0 + G_{xu}\mathbb{U} + G_{xw}\mathbb{W}. \tag{3}$$

where the matrices G_{xx}, G_{xu} and G_{xw} are fixed, and their entries depend only on A, B, and H. Here, N is called prediction horizon. Note that $\mathbb{X}$ in (3) is a multivariate normal random vector, a direct consequence of the i.i.d. Gaussian assumption on $\mathbb{W}$.

Set $\bar{\mathbb{X}}$ as the mean value of $\mathbb{X}$ and let $S_{\mathbb{X}}$ be its covariance, both given by [3]

$$\begin{aligned} \bar{\mathbb{X}} &= G_{xx}\bar{x}_0 + G_{xu}\mathbb{U} + G_{xw}\bar{\mathbb{W}} \\ S_{\mathbb{X}} &= G_{xx}S_{x_0}G'_{xx} + G_{xw}S_{\mathbb{W}}G'_{xw}. \end{aligned} \tag{4}$$

The probabilistic constraints to be imposed on (1) are defined as follows. Let F_X be the set made up by the intersection of M linear inequality constraints, i.e.,

$$F_X := \bigcap_{i=1}^{M} \{\mathbb{X} : a'_i\mathbb{X} \le b_i\}, \tag{5}$$

where a_i and b_i, $i = 1, \dots, M$, represent vectors on $\mathbb{R}^{nN}$ and $\mathbb{R}$, respectively. Moreover, consider F_U as a set defined in similar fashion.

Assume for the moment that a given functional $J : F_X \times F_U \mapsto \mathbb{R}_+$, convex with respect to its arguments, is associated with (1). Hereafter, this functional is referred to as cost function.

Problem 1. Given some $\delta \in \mathbb{R}_{(0,1)}$, find a solution for the next optimization problem.

$$\begin{aligned} &\min_{\mathbb{U} \in F_U} \; J(\bar{\mathbb{X}}, \mathbb{U}) \\ &\text{s.t.} \;\; \Pr[\mathbb{X} \notin F_X] \le \delta, \\ &\mathbb{X} = G_{xx}x_0 + G_{xu}\mathbb{U} + G_{xw}\mathbb{W} \end{aligned} \tag{6}$$

As a matter of fact, Problem 1 means that the control input minimizes the cost function while ensuring that the system state may leave the feasible region with a probability less or equal than δ.

To the best of the authors' knowledge, Problem 1 remains unresolved. The difficulty to solve such a problem stems from the fact that the probability in (6) represents a value taken over by multivariate integrals, requiring their evaluation, a difficult task to accomplish [12, 17].

To avoid the evaluation of multivariate integrals, the authors of [3] suggest the element-wise evaluation of the set F_X in probability; namely, find constants ε_i's such that $\sum_{i=1}^{M} \varepsilon_i \le \delta$, and

$$\Pr\left[a'_i\mathbb{X} > b_i\right] \le \varepsilon_i, \quad \forall i \in \mathbb{Z}_{[1,M]}. \tag{7}$$

Note that ε_i's are slack variables, now in association with a univariate probabilistic inequality, which can be computed via the standard cumulative distribution function. Even so, despite the progress researchers have made so far, the problem still remains difficult to solve because checking (7) may be cumbersome from the numerical point of view, as detailed next.

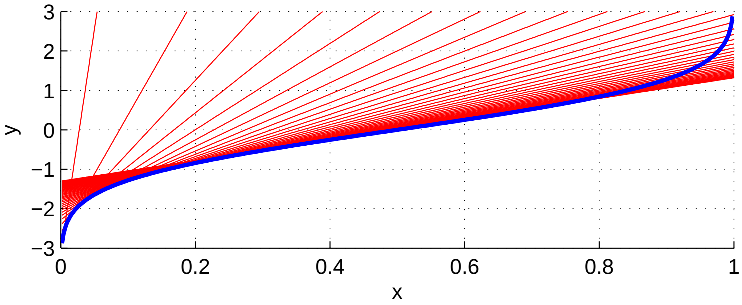


Figure 1: The inverse cumulative distribution function (thick line) and upper linear approximations (straight lines tangent to the icdf function).

Let cdf(·) be the standard Gaussian cumulative distribution function

$$\mathrm{cdf}(\gamma) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\gamma} e^{-\frac{z^2}{2}} dz, \tag{8}$$

and let icdf(·) be the corresponding inverse.

One can show that (7) holds if and only if both inequalities $\sum_{i=1}^{M} \varepsilon_i \leq \delta$ and

$$\frac{b_i - a_i'\bar{\mathbb{X}}}{s_i} \geq \mathrm{icdf}\left(1 - \varepsilon_i\right), \quad \forall i \in \mathbb{Z}_{[1,M]}, \tag{9}$$

hold true, where $s_1, \ldots, s_M$, are fixed, positive constants.

Even though the inequalities in (9) are convex with respect to ε_i's, as shown in [3], finding a solution for (9) may be cumbersome—there is no closed, algebraic expression for computing a feasible solution for (9). Consequently, to evaluate (9), it is necessary to use iterative algorithms, usually slow for computing the point of convergence [1, Ch. 7]. But slow algorithms cannot be applied in some online control schemes, such as model predictive control [6, 7, 14, 19, 24, 26].

In order to speed up the computation of a feasible solution for (9), we convert (9) into an affine function so as to obtain linear matrix inequalities (LMI)—it is known that LMI solvers outperform classical convex optimization algorithms [4, Ch. 1]. The idea behind such conversion is now detailed.

As described in Fig. 1, the icdf function can be bounded from above by an affine function on some interval. This fact allows us to define an affine function $h(\cdot)$ such that

$$h(1 - \varepsilon_i) \geq \mathrm{icdf}\left(1 - \varepsilon_i\right), \quad \forall i \in \mathbb{Z}_{[1,M]}. \tag{10}$$

The novel condition hinges on checking whether the inequalities

$$\frac{b_i - a_i'\bar{\mathbb{X}}}{s_i} \geq h\left(1 - \varepsilon_i\right), \quad \forall i \in \mathbb{Z}_{[1,M]}, \tag{11}$$

hold true. In the positive case (9) is assured; as a byproduct, we obtain $\mathbb{U}$ that suffices to (6) in Problem 1, the main condition pursued in this paper.

The main contribution of this paper is capable of showing (11). Yet derived from a simple idea of imposing upper bounds on the icdf function, the condition in (11) represents linear matrix inequalities, useful for real-time experiments—the paper shows a real-time application for a direct current motor (see Section 4). This application illustrates the practical benefits of our approach.

3 Model predictive control with probability constraints

The problem we solve in this paper is presented in what follows. Define $y_i = a_i' x$, and note that y_i is a univariate Gaussian random variable with mean and variance given by

$$\begin{aligned}\bar{y}_i &= a_i' G_{xx} \bar{x}_0 + a_i' G_{xu} \mathbb{U} + a_i' G_{xw} \bar{\mathbb{W}} \\ S_{y_i} &= a_i' G_{xx} S_{x_0} G_{xx}' a_i + a_i' G_{xw} S_{\mathbb{W}} G_{xw}' a_i.\end{aligned} \tag{12}$$

Using the novel condition in (11), we recast the Problem 1 as the next problem.

Problem 2. Given some $\delta \in \mathbb{R}_{(0,1)}$, find the solution to the optimization problem

$$\begin{aligned}&\min_{\mathbb{U} \in F_U} \ J(\bar{\mathbb{X}}, \mathbb{U}) \\ &\text{s.t. } \sum_{i=1}^{M} \varepsilon_i \le \delta, \quad \frac{b_i - \bar{y}_i}{\sqrt{S_{y_i}}} \ge h\left(1 - \varepsilon_i\right), \\ &\bar{y}_i = a_i' G_{xx} \bar{x}_0 + a_i' G_{xu} \mathbb{U} + a_i' G_{xw} \bar{\mathbb{W}}, \\ &S_{y_i} = a_i' G_{xx} S_{x_0} G_{xx}' a_i + a_i' G_{xw} S_{\mathbb{W}} G_{xw}' a_i, \ \forall i \in \mathbb{Z}_{[1,M]}.\end{aligned} \tag{13}$$

The next result is a direct consequence of the aforementioned facts.

Theorem 3. *A feasible solution to Problem 2 is a feasible solution to Problem 1.*

Proof: Recall that

$$\begin{aligned}\Pr[\mathbb{X} \notin F_X] &= \Pr\left[\mathbb{X} \notin \bigcap_{i=1}^{M} \{\mathbb{X} : a_i' \mathbb{X} \le b_i\}\right] \\ &= \Pr\left[\mathbb{X} \in \bigcup_{i=1}^{M} \{\mathbb{X} : a_i' \mathbb{X} > b_i\}\right].\end{aligned} \tag{14}$$

The rightmost term in (14) is bounded from above by $\sum_{i=1}^{M} \Pr[a_i' \mathbb{X} > b_i]$ due to the Boole's inequality. When Problem 2 has a solution, we have from (10) and (13) that

$$\frac{b_i - \bar{y}_i}{\sqrt{S_{y_i}}} \ge h\left(1 - \varepsilon_i\right) \ge \operatorname{icdf}\left(1 - \varepsilon_i\right), \quad \forall i \in \mathbb{Z}_{[1,M]},$$

which shows (9). Both (9) and $\sum_{i=1}^{M} \varepsilon_i \le \delta$ assure $\Pr[a_i' \mathbb{X} > b_i] \le \varepsilon_i$ for each $i = 1, \ldots, M$ from (7), which in turn assures that $\Pr[\mathbb{X} \notin F_X] \le \delta$. This argument completes the proof. □

Remark 4. Theorem 3 assures that any optimal solution to Problem 2 is a sub-optimal solution to Problem 1—it means that the optimality of Problem 1 is lost; however, the sub-optimal strategy turns the problem tractable from the numerical viewpoint. Theorem 3 then contributes towards finding a solution to the so-far unsolved Problem 1.

Recall that the model predictive control (MPC) works under the receding horizon control principle [14, Ch. 1]; namely, the cost function $J(\cdot)$ in Problem 2 should be minimized at each time instant $k = k_0, \ldots, k_1$. The current input u_k in (1) is obtained by determining the input sequence $\{\hat{u}_k, \ldots, \hat{u}_{k+N-1}\}$ that solves Problem 2, and next by setting $u_k = \hat{u}_k$. The remaining elements of the sequence are discarded, and this iterative procedure is repeated [14, Ch. 1].

The aforementioned iterative procedure, key for the MPC scheme, was implemented in a laboratory testbed as detailed in the next section.

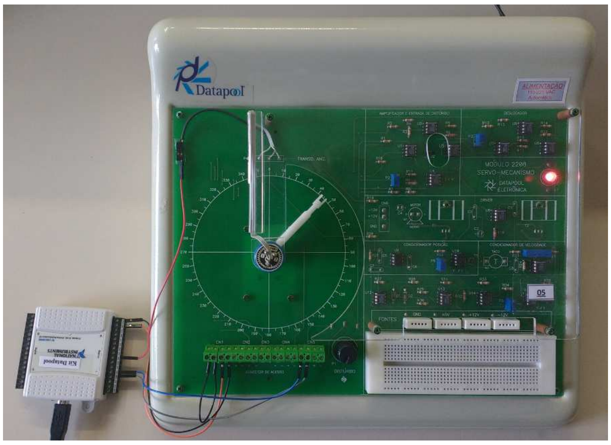


Figure 2: Equipment overview.

4 Application of model predictive control to a direct current motor subject to probability constraints

The goal of this section is to illustrate the proposed model predictive control scheme with probability constraints. To do so, we present data collected from real-time experiments—the experiments were carried out in a laboratory tested designed to control a direct current (DC) motor.

4.1 Identification

The laboratory testbed was assembled with “Modulo 2208 servo-mecanismo”, a DC motor device assembled by Datapool-Brazil [18,21], which worked in association with a data-acquisition board (model USB-6008 DAQ) from National Instruments (see Fig. 2). The board was configured to work with fixed sampling time of 20 ms.

The DC motor was equipped with a tachometer—it generated a voltage in the range $[0, 5]$ V for an angular velocity in the range of $[0, 26.17]$ rad/s, in a linear relationship. Likewise, a current sensor generated the current consumed by the motor in the range $[0, 5]$ A.

Pseudo random binary signals (PRBS) with amplitudes between 0 and 5 V were generated and applied in a sixth-order Butterworth low-pass filter with a cutoff frequency randomly chosen between 0.01Hz and 70Hz. The resulting signal was then applied in the input of the DC motor, and the corresponding data was collected and stored.

Each experiment for sake of identification was performed in chunks of 60s, that is, each chunk contained three thousand points. The data were then split into smaller chunks, each with 1300 points. These chunks were used to obtain a model in the observability canonical form through the subspace identification method [8,13]. The canonical model was then modified to cope with the prediction-error minimization algorithm. Finally, the obtained model was validated through unused, uncorrelated data (see Fig. 3 for illustration of a sample). This procedure produced the model ($\approx 85\%$ fitting):

$$\begin{bmatrix} \omega_{k+1} \\ i_{k+1} \end{bmatrix} = \begin{bmatrix} 0.9814 & 0.4582 \\ -0.0087 & 0.7667 \end{bmatrix} \begin{bmatrix} \omega_k \\ i_k \end{bmatrix} + \begin{bmatrix} 0.0064 \\ 0.0207 \end{bmatrix} (u_k + d_k), \tag{15}$$

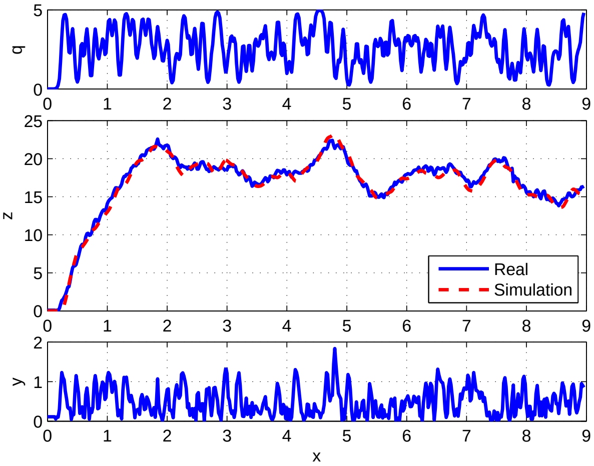


Figure 3: Real and simulated data: experiment suggests that the model attained for the DC motor seem appropriate for representing the real-time equipment.

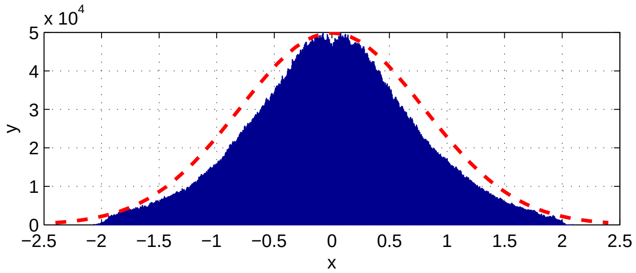


Figure 4: Histogram of disturbances. Dashed curve shows the Gaussian distribution.

where ω, i, u, and d represent the angular velocity of the shaft, the electrical current consumed by the motor, the armature voltage applied in the terminals of the DC motor, and disturbance, respectively.

The disturbance d stands for an i.i.d. stochastic process under Gaussian probability distribution—the corresponding experimental data is summarized in Fig. 4.

4.2 Probability constraints

Armature voltage (control signal) that drove the motor was restricted by lower and upper bounds as

$$u^{\min} \leq u_k \leq u^{\max}, \tag{16}$$

where $u^{\min}$ and $u^{\max}$ are the minimum and maximum voltages. The rotor angular velocity and the armature current were bounded as well, i.e.,

$$\omega^{\min} \leq \omega_k \leq \omega^{\max}, \quad i^{\min} \leq i_k, \quad \text{and} \quad \Pr[i_k \geq i^{\max}] \leq \delta, \tag{17}$$

where $\omega^{\min}$, $\omega^{\max}$, $i^{\min}$, and $i^{\max}$ denote the corresponding limits.

The control objective was to reach a desired value of the rotor angular velocity, say ω^{ss}, while fulfilling the constraints given in (16) and (17).

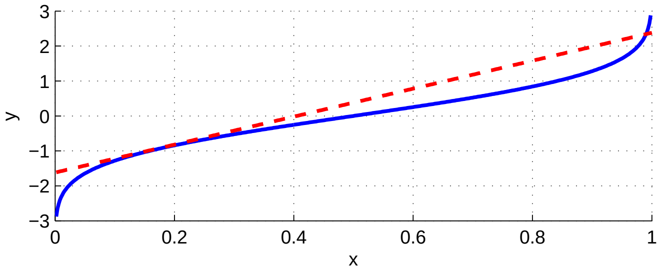


Figure 5: Inverse cumulative distribution function and its upper linear approximation (dashed line).

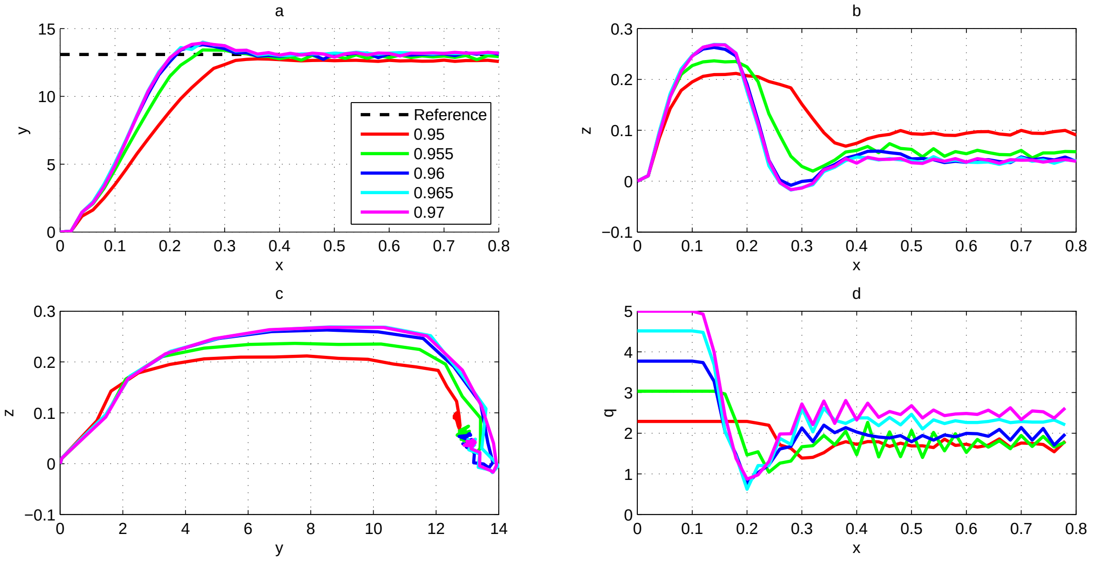


Figure 6: Data from real-time experiments. Problem 2 was evaluated for each $\delta = 0.95, 0.955, 0.96, 0.965, 0.97$, and every curve shows the mean of one-hundred sample paths. The curves show (a) angular velocity of the DC motor; (b) armature current of the DC motor; (c) phase-portrait of the system; and (d) armature voltage of the DC motor.

4.3 Experimental results

Recall that Problem 2 requires a linear approximation for the inverse cumulative distribution function. To fulfill this requirement, the linear approximation $h(1-\varepsilon_i) = 2.38 - 4\varepsilon_i$ was considered, valid for $\varepsilon_i \in [0.01, 1)$ (see Fig. 5).

Associated with Problem 2, we consider the cost

$$J(\bar{\mathbb{X}}, \mathbb{U}) = 1000||\bar{\mathbb{X}}||_\infty + 2||\mathbb{U}||_\infty. \tag{18}$$

Problem 2 was evaluated with the following additional parameters: $N = 2$, $S_{x0} = I_2$ and $S_{\mathbb{W}} = I_2$.

The limits for the constraints in (16) and (17) were set as $u^{\min} = 0$ V, $u^{\max} = 5$ V, $\omega^{\min} = 0$ rad/s, $\omega^{\max} = 26.17$ rad/s, $i^{\min} = 0$ A and $i^{\max} = 0.38$ A. The reference for the angular velocity was fixed at $\omega^{ss} = 13$ rad/s.

Problem 2 was evaluated with δ taking values 0.95, 0.955, 0.96, 0.965, and 0.97, and every corresponding solution was checked in practice. Indeed, one-hundred experiments were conducted for each value of δ—the experimental outcome is subsumed under the curves shown Fig. 6.

The curves in Fig. 6 show the statistical mean of the stored one-hundred samples. Fig. 6(a) shows the angular velocity of the DC motor; it can be observed that as δ decreases, settling time increases. Armature current is represented in Fig. 6(b), which shows that the upper bound on the current is satisfied and the maximum value of the current decreases when the value of δ decreases. Phase portrait of the process is illustrated in Fig. 6(c)—data suggest that the DC model be second moment stable [22]. The curves of the armature voltage (control input) in Fig. 6(d) suggest that as long as δ decreases, the upper bound on the armature voltage also decreases; as a result, δ successfully actuated as a constraint on the armature voltage.

In summary, the experimental data indicate that our approach seems to be useful to handle real-time applications subject to physical constraints.

5 Conclusion

The paper presents an easy-to-check method to handle linear stochastic systems subject to probability constraints. Here, constraints hinge upon Gaussian distributions. However, nonlinear equations arise naturally from that Gaussian-based evaluations, turning the problem difficult to solve. To overcome such a drawback, we have converted those nonlinear equations into simple-to-check linear matrix inequalities (conversion guarantees only sub-optimal solutions, see Remark 4). The derived conversion shows appropriate for real-time applications—specially when fast algorithms are necessary—as illustrated in the paper.

Acknowledgments

This work was partially supported by the Thousand Talents Plan of Shaanxi Province for Young Professionals, and the Brazilian agencies FAPESP Grants 03/06736-7; CNPq Grant 304856/2007-0; and CAPES Grant Program PVE 88881.030423/2013-01.

Bibliography

[1] Bazaraa, M.S.; Sherali, H.D.; Shetty, C.M. (2006). *Nonlinear programming: theory and algorithms*, 3rd edn., Wiley-Interscience, New Jersey, 2006.

[2] Bernardini, D.; Bemporad, A. (2012). Stabilizing model predictive control of stochastic constrained linear systems, *IEEE Trans. Autom. Control*, 57, 1468–1480, 2012.

[3] Blackmore, L.; Ono, M. (2009). Convex chance constrained predictive control without sampling, *In: AIAA Guidance, Navigation and Control Conference*, Chicago, Illinois, USA, 1-17, 2009.

[4] Boyd, S.; El Ghaoui, L.; Feron, E.; Balakrishnan, V. (1994). *Linear matrix inequalities in system and control theory*, SIAM, Philadelphia, 1994.

[5] Cannon, M.; Kouvaritakis, B.; Rakovic, S.; Cheng Q. (2011). Stochastic tubes in model predictive control with probabilistic constraints, *IEEE Trans. Autom. Control*, 56, 194–200, 2011.

[6] Cao, G.; Lai, E.M.-K.; Alam, F. (2017). Gaussian process model predictive control of unknown non-linear systems, *IET Control Theory Appl.*, 11, 703–713, 2017.

[7] Caruntu, C.F.; Balau, A.E.; Lazar, M.; van den Bosch, P.P.J.; Di Cairano, S. (2016). Driveline oscillations damping: A tractable predictive control solution based on a piecewise affine model, *Nonlinear Analysis: Hybrid Systems*, 19, 168–185, 2016.

[8] Costa Junior, A.G.; Riul J.A.; Montenegro, P.H.M. (2016). Application of the subspace identification method using the N4SID technique for a robotic manipulator, *IEEE Latin America Transactions*, 14, 1588–1993, 2016.

[9] Farina, M.; Giulioni, L.; Scattolini, R. (2016). Stochastic linear model predictive control with chance constraints-A review, *Journal of Process Control*, 44, 53–67, 2016.

[10] Farina, M.; Giulioni, L.; Magni, L.; Scattolini, R. (2015). An approach to output-feedback MPC of stochastic linear discrete-time systems, *Automatica*, 55, 140–149, 2015.

[11] Farina, M.; Scattolini, R. (2016). Model predictive control of linear systems with multiplicative unbounded uncertainty and chance constraints, *Automatica*, 70, 258–265, 2016.

[12] Hashorva, E.; Hüsler, J. (2003). On multivariate Gaussian tails, *Annals of the Institute of Statistical Mathematics*, 55, 507–522, 2003.

[13] Katayama, T. (2005). *Subspace methods for system identification, communications and control engineering*, Springer-Verlag, London, 2005.

[14] Kwon, W.H., Han, S.H. (2005). *Receding horizon control: model predictive control for state models*, Springer-Verlag, New York, 2005.

[15] Li, P.; Wendt, M.; Wozny, G. (2002). A probabilistically constrained model predictive controller, *Automatica*, 38, 1171–1176, 2002.

[16] Li, J.W.; Li, D.W.; Xi, Y.G. (2017). H_∞ predictive control with probability constraints for linear stochastic systems, *IET Control Theory Appl.*, 11, 557–566, 2017.

[17] Lu, D.; Li, W.V. (2009). A note on multivariate Gaussian estimates, *Journal of Mathematical Analysis and Applications*,354, 704–707, 2009.

[18] Oliveira, R.C.L.; Vargas, A.N.; do Val, J.B.R.; Peres, P.L.D. (2014). Mode-independent H_2-control of a DC motor modeled as a Markov jump linear system, *IEEE Transactions on Control Systems Technology*, 22, 1915–1919, 2014.

[19] Rubagotti, M.; Patrinos, P.; Guiggiani, A.; Bemporad, A. (2016). Real-time model predictive control based on dual gradient projection: theory and fixed-point FPGA implementation, *Int. J. Robust Nonlinear Control*, 26, 3292–3310, 2016.

[20] Schwarm, A.T.; Nikolaou, M. (1999). Chance-constrained model predictive control, *AIChE Journal*, 45, 1743–1752, 1999.

[21] Vargas, A.N.; Costa, E.F.; do Val, J.B.R. (2013). On the control of Markov jump linear systems with no mode observation: application to a DC motor device, *Int. J. Robust Nonlinear Control*, 23, 1136–1950, 2013.

[22] Vargas, A. N.; do Val, J.B.R. (2010). Average cost and stability of time-varying linear systems, *IEEE Trans. Autom. Control*, 55, 714–720, 2010.

[23] Wang, S.; Yu, M.; Sun, X. (2015). Robust H_∞ control for time-delay networked control systems with probability constraints, *IET Control Theory Appl.*, 9, 482–2489, 2015.

[24] Yang, H.; Guo, M.C.; Xia, Y.; Cheng, L. (2018). Trajectory tracking for wheeled mobile robots via model predictive control with softening constraints, *IET Control Theory Appl.*, 12, 206–214, 2018.

[25] Yan, J.; Bitmead, R.R. (2005). Incorporating state estimation into model predictive control and its application to network traffic control, *Automatica*, 41, 595–604, 2005.

[26] Zeilinger, M.N.; Raimondo, D.M.; Domahidi, A.; Morari, M.; Jones, C.N. (2014). Flocking of multi-agents with a virtual leader, *Automatica*, 50, 683–694, 2014.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(6), 938-955, December 2018.

ANN based Short-Term Load Curve Forecasting

V. Chis, C.Barbulescu, S. Kilyeni, S. Dzitac

Violeta Chis
Mathematics and Computer Science Department
Aurel Vlaicu University of Arad
Arad, Romania
violeta.chis@uav.ro

Constantin Barbulescu*, Stefan Kilyeni
Power Systems Department
Politehnica University Timisoara Romania
Timisoara, Romania
*Corresponding author: constantin.barbulescu@upt.ro
stefan.kilyeni@upt.ro

Simona Dzitac
Power Engineering Department
University of Oradea
Oradea, Romania
simona@dzitac.ro

Abstract: A software tool developed in Matlab for short-term load forecasting (STLF) is presented. Different forecasting methods such as artificial neural networks, multiple linear regression, curve fitting have been integrated into a stand-alone application with a graphical user interface. Real power consumption data have been used. They have been provided by the branches of the distribution system operator from the Southern-Western part of the Romanian Power System. This paper is an extended variant of [4][a].

Keywords: artificial neural networks; short-term load forecasting; artificial intelligence; load curve.

[a] Partially reprinted and extended, with permission based on Licence Number 4453460839057 © IEEE, from "2018 7th International Conference on Computers Communications and Control (ICCCC)."

1 Introduction

The current paper is an extended version of [4]. It deals with real power systems data applied for short-term daily load forecasting. In [4] one single distribution branch has been discussed. Currently, the work has been extended for all of the distribution branches belonging to the distribution system operator (DSO) involved. A new set of data, corresponding to the entire network of the DSO has been added. Finally, a comparative analysis has been provided for all the distribution branches involved and the entire network assembly.

Load forecasting has a great impact on future decisions, predicting the energy demand for the operation and planning of power systems plays an important role in the control, power security, market operation, and scheduling of reasonable dispatching plans for smart grids.

Load forecasting is an important component of power system to establish economical and reliable operations for power stations and their generating units. An accurate load forecasting approach used to predict load demand is essential part of any energy management system.

The load forecasting methods are generally classified in statistical-based methods like exponential smoothing, regression, Kalman filter, state space model and artificial intelligence based methods such as expert system techniques [7], neural networks (NN), fuzzy expert system [20], fuzzy time series, fuzzy neural networks, genetic algorithms [8], support vector machine, pattern recognition and hybrid methods.

Load forecasting is carried out over a wide range of intervals ranging from a few seconds to decades and is mainly based on retrospective load variation data. According to the time span, load forecasting can be divided into very short (a few minutes), short term load forecasting (STLF), medium term load forecasting (MTLF) and long term load forecasting (LTLF).

LTLF covers a period of several years and is applicable for system and long term network planning. The MTLF covers a period of a few weeks and is applicable for scheduling of the fuel supply, the planning of maintenance, etc. The STLF refer to hourly prediction of the load for a lead time ranging from one hour to several days out and it's used to predict load demands so that the day-to-day operation of a power system can be efficiently planned [9].

The accuracy of load forecasting depends significantly on the availability of historical consumption data as well as on the knowledge about the main influence parameters on the energy consumption [16].

The quality of the forecast methods is influenced by several factors including:

- weather data- temperature, humidity, wind speed, fog, precipitation, solar radiation;
- seasonal (time) factors - seasons' sequence, day of the week and hour of the day (with low periods during the night hours and with peaks at different day hours);
- causative factors (holidays, strikes and some specific public events, etc.);
- economic factors (industrial activity, demographic change, economic growth).

The influence of these factors is different from one type of forecast to another.

Short-term load forecasting (STLF) performs load forecasting of the system from few hours to weeks, using generally historical load data and weather data as inputs. The forecasted data are used to estimate load flows, for transmission line loading, for transient stability studies. Many decisions, like generating capacity, planning for energy transactions or system security assessment are based on STLF.

Researchers have considered different approaches to STLF, like time series method, regression analysis [1], intelligent techniques, such as NN [2], [17], fuzzy logic [10], [14], neuro-fuzzy [6] or data mining [18], [21].

Time series have been used for decades in areas such as the economy, digital signal processing and the load forecasting. ARMA (AutoRegressive Moving Average) [3], ARIMA (Auto Regressive Integrated Moving Average) [11] and ARIMAX (AutoRegressive Moving Average with Exogeneous Variables) [5] are the most commonly used time series methods.

Since 1990, the focus has been on using different Artificial Intelligence (IA) techniques. Thus, the authors of the paper [15] were among the first research groups that chose to use NN for the STLF. In [12], a model based on decision trees is used for load forecasting.

In [19] the neural networks are also used for load forecasting problem, due to the non-linear character of the load. But, the authors are going further, combining the ANNs with optimization techniques. They are proving that the forecast error is reduced by applying this method, in order to estimate the parameters of the network.

Another hybrid model is applied in [22]. The authors are focusing on empirical mode decomposition (IEMD), autoregressive integrated moving average (ARIMA) and wavelet neural network (WNN). All these algorithms are optimized by fruit fly optimization algorithm (FOA). This kind of approach is recommended in case of load forecasting affected by natural and social factors.

Paper [13] is one of the papers that are dealing with dynamic neural networks, in order to predict the daily power consumption. These networks are able to adaptively learn de patterns from historical data. They are capable to tackle the high non-linear degree between input and output data.

The introduction is presented within the 1st section of the paper. The 2nd one focuses on describing the applied models. The software-tool is briefly presented within the 3rd one. The case study and the results are largely discussed within the 4th section. Finally, the 5th one synthesizes the conclusions.

2 Models used

Several models have been implemented within the developed software tool.

2.1 Multiple linear regression model

Regression methods are used to model the relationship between load consumption and other factors such as weather, day type, stochastic influences such as average loads and customer class [23].

The following data have been used within the current model: previous day, type of the day (working day or weekend), previous day same hour load and previous 24 hour average load.

The following stages have been tackled:

- generate predictors (previousDayHour, pre24HourAverLoad, day, dayweek);
- forming validation input data (vpreviousdayHour, vpre24HourAverLoad, vday, dayweek);
- create regression coefficients *(b, bint, r)*;
- validate the regression coefficients (evaluating the input data) - calculate the mean absolute percent error (MAPE) and plot actual load vs. predicted load;
- use the regression coefficients to forecast one day ahead - calculate MAPE and plot actual load vs. forecasted load.
- use the regression coefficients to forecast one day ahead - calculate MAPE and plot actual load vs. forecasted load

The time factors include the time of the day of the week, and the day hour. There are important differences in load between weekdays and weekends. The load on different weekdays also can behave differently.

2.2 Curve fitting model

From the Fourier library models, fourier8 has been used. Multiple regression has been used to obtain an average estimate. Figure 1 presents synthetically the algorithm of this method for STLF.

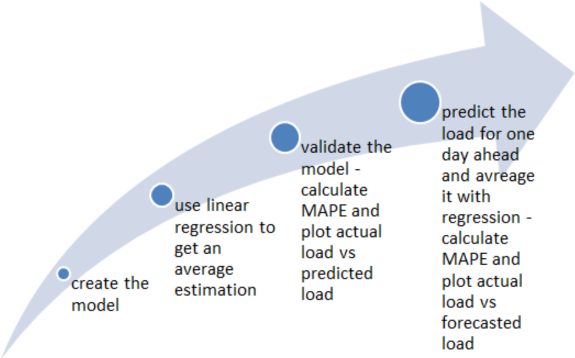

create the model
use linear regression to get an average estimation
validate the model - calculate MAPE and plot actual load vs predicted load
predict the load for one day ahead and avreage it with regression - calculate MAPE and plot actual load vs forecasted load

Figure 1: Curve fitting algorithm

2.3 The neural network model

Several papers propose the use of NN for complex problems like STLF, which are highly non-linear. The main advantages of NN are that it can learn to gain on any nonlinear function from a large number of data.

To identify the assumed relation between the future load and the earlier load data a multilayer perceptron (MLP) network with a single hidden layer is used. MLP is one of the most well-known NN architectures for prediction algorithms, and is popular because of its flexibility in assuming shapes of complex patterns. The introduction of hidden layer(s) makes it possible for the network to exhibit non-linear behavior. For calculation of hidden layer neurons does not exist any specific formula. Researchers have observed that too few hidden layer neurons can cause the network to not learn the convergence and too many hidden layers can cause the network to memorize the scheme rather than forecasting [2], [17].

The NN is trained using historical data and use back propagation (BP) method. BP is widely applied for STLF because of its ability to study and remember the relation between inputs and outputs as well as to approach any types of function. For training, the network uses the default BP training algorithm, Levenberg-Marquardt. This is the fastest method in the toolbox for training moderate-sized feed-forward neural networks. The overall design of NN is presented in Figure 2.

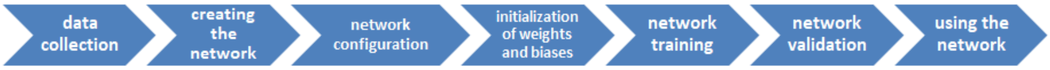


Figure 2: The overall design of NNs

For the current paper the NN architecture has the following configuration: one output (load), six inputs: load on previous day at same hour, load on previous week at same hour, month, day, day of the week (there are important differences in load between weekdays and weekends).

3 Software tool

A software tool has been developed in Matlab environment for STLF. Of course, there are commercial software packages. Their main disadvantage is that they are a black box, offering no modeling transparency and they are more difficult to modify.

Data are retrieved from Microsoft Excel files. For different loads, the influence of different factors is not the same: temperature could have a major influence for residential loads, but may have little impact on some industrial loads. In this case the temperature has not been taken into account. Future power demand is estimated based on the historical load data.

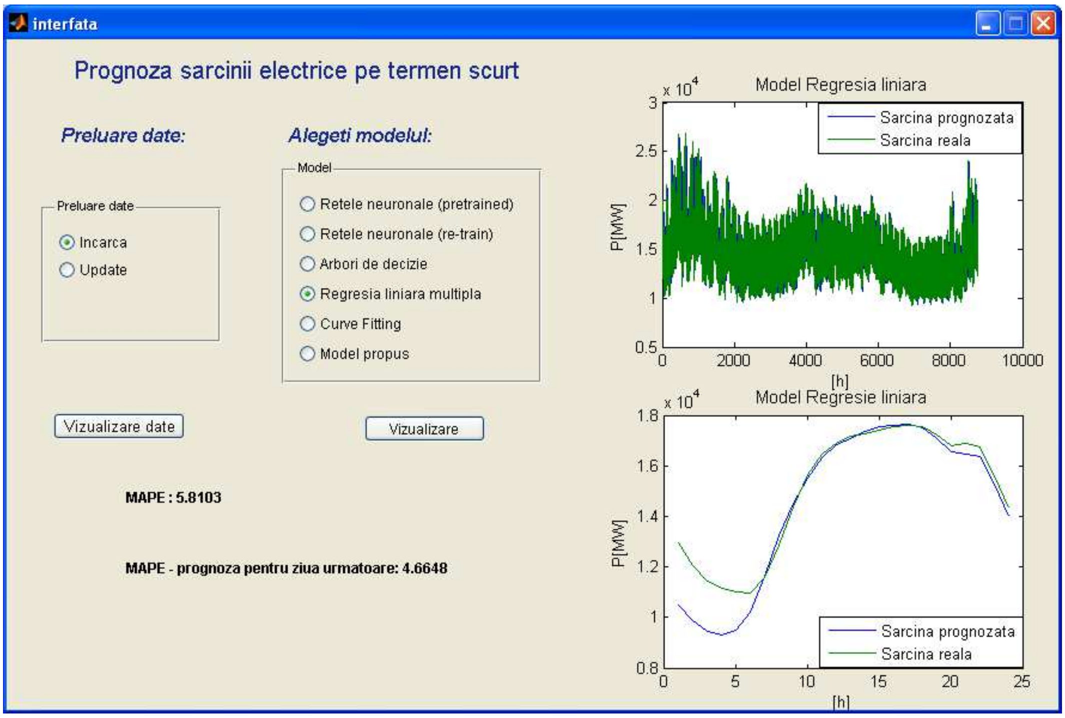
The developed software allows a quick graphical comparative analysis: *Actual Load* versus *Load Forecasting*. The MAPE is used as a performance criterion.

$$MAPE = \frac{1}{N}(\sum_{i=1}^{n} \frac{|y_i - p_i|}{y_i}) * 100 \tag{1}$$

where y_i, p_i- actual and forecasted load of i hour; $N-$ the forecasting horizon

This application was developed using the Matlab GUI Tool-box and implements different models. The models developed have been integrated into a stand-alone application with a graphical user interface.

Starting from the idea that the interface design activity should be centered on the user, it was intended to meet the recommended requirements for such interfaces: friendly, intuitive, easy-to-use, extensible but also consistent. Radio buttons have been used to select the model (Figure 3). Depending on data, one may be better than another. The developed software allows comparing different models automatically.



interfata

Prognoza sarcinii electrice pe termen scurt

Preluare date:

Preluare date
● Incarca
○ Update

Alegeti modelul:

Model
○ Retele neuronale (pretrained)
○ Retele neuronale (re-train)
○ Arbori de decizie
● Regresia liniara multipla
○ Curve Fitting
○ Model propus

MAPE : 5.8103

MAPE - prognoza pentru ziua urmatoare: 4.6648

Figure 3: The software tool main window

4 Results and discussions

Real data have been used, that are belonging to the Distribution System Operator managing the distribution network from the Western side of Romania. The load curves forecasting is performed for the most significant summer day (21^{st} June). 10 years known period is considered (2004-2013). Results validation refers to the range of 2014-2016 years. The forecasts have been performed using artificial neural networks (ANN), multiple linear regression (MLR) and curve fitting (CF).

Table 1 and Figure 4 contain the known load curves for the 2004-2013 period, for the most significant summer day.

Table 1: Load curves for known period 2004-2013 [MW]

Year/Hour	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
1	67.70	72.40	68.70	69.90	65.00	55.30	61.20	63.30	64.80	61.30
2	64.30	68.90	65.50	67.40	61.20	52.30	58.00	59.30	59.90	56.40
3	62.10	66.10	62.80	66.20	58.00	50.10	55.80	56.90	57.60	54.10
4	61.80	64.90	61.90	65.10	57.30	49.00	54.10	55.70	55.80	54.30
5	63.90	68.40	64.50	66.30	59.00	50.90	56.00	58.00	58.10	58.50
6	68.70	74.90	68.90	69.30	63.60	54.60	59.80	61.70	60.30	58.60
7	75.60	81.40	73.60	72.30	65.80	56.40	64.20	63.40	63.60	61.80
8	89.90	97.10	86.40	84.90	80.70	70.60	82.90	79.90	81.30	78.30
9	92.60	101.6	92.60	89.90	86.00	77.60	86.50	85.00	84.80	83.00
10	90.30	99.30	89.30	88.10	88.10	78.60	84.50	85.00	85.40	81.60
11	85.40	95.20	81.60	84.10	84.00	75.30	81.30	80.80	81.10	78.50
12	85.50	95.10	81.40	83.60	85.00	75.90	80.50	80.60	82.30	80.30
13	85.10	94.00	82.80	84.60	84.60	77.80	81.60	83.20	84.90	81.50
14	85.00	93.10	83.40	84.40	84.60	78.20	81.90	84.80	88.00	81.70
15	82.70	91.50	82.40	83.10	79.70	77.20	82.20	82.20	87.30	80.00
16	75.60	84.20	75.70	78.20	73.80	70.80	76.00	77.60	81.90	74.50
17	72.70	81.50	73.30	75.30	72.00	68.60	73.40	74.20	78.10	71.50
18	72.40	80.60	73.20	73.90	70.50	66.50	71.50	70.60	76.60	69.70
19	70.30	80.40	72.20	72.30	67.80	64.80	69.60	69.50	72.10	66.80
20	71.80	81.90	72.20	71.60	67.20	65.00	67.70	68.30	68.50	66.00
21	88.40	95.20	81.90	80.20	73.60	69.10	70.00	69.80	72.80	68.10
22	95.50	101.9	94.20	90.50	89.00	83.60	80.10	79.50	74.60	76.40
23	86.20	91.20	86.70	81.50	84.80	81.40	81.10	81.30	75.40	82.40
24	75.50	79.20	76.40	71.70	73.60	69.50	70.80	70.80	69.00	72.30

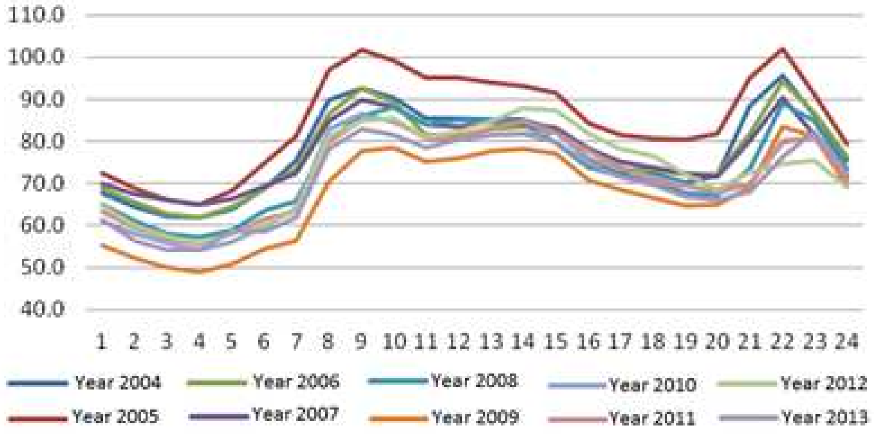


Figure 4: Load curves' variation for the 2004-2013 period [MW]

Table 2 and Figure 5 are presenting the load curves for the 2014-2016 period. They are going to be used in order to validate the forecast based on previously presented load curves (Table 1).

Table 2: Load curves for 2014-2016 period [MW] - validation period

Year/Hour	**2014**	**2015**	**2016**
1	62.60	65.00	71.00
2	57.60	59.90	65.20
3	56.30	58.50	63.00
4	55.40	57.40	61.20
5	55.20	58.40	62.20
6	58.30	58.20	64.10
7	60.60	60.70	67.50
8	78.80	78.70	82.20
9	84.60	83.00	87.60
10	85.50	84.30	87.50
11	81.90	80.80	85.00
12	83.70	80.90	86.30
13	84.50	82.30	87.20
14	86.80	83.80	87.00
15	84.00	84.10	86.60
16	78.60	79.90	80.00
17	75.90	79.00	79.00
18	75.00	74.50	77.40
19	71.80	71.70	76.30
20	71.40	71.20	74.30
21	74.00	72.40	74.40
22	87.00	88.10	89.30
23	81.00	83.30	84.20
24	69.50	75.40	76.10

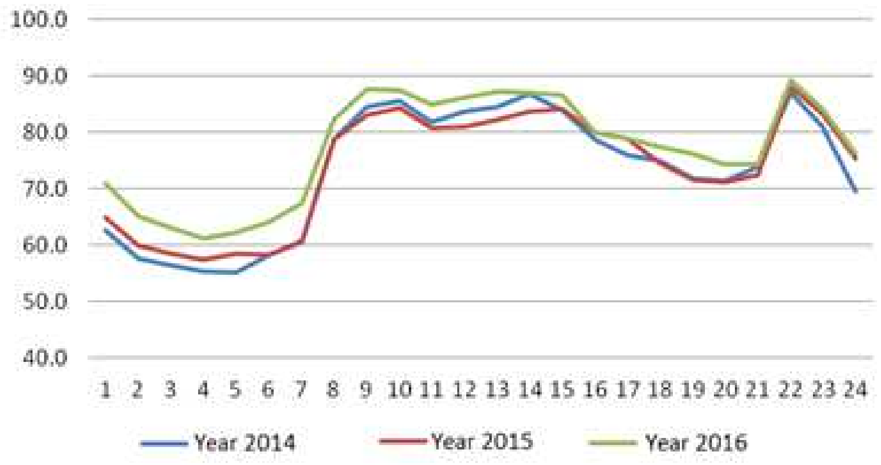


Figure 5: Load curves' variation for the 2014-2016 period [MW]

The following comments are suitable for the case of Tables 1 and 2 (respectively Figure 4 and Figure 5):

- a rigorous consumed power ascending or descending trend is not highlighted, in generally;

- the load curves are crossing each other (thus, their shape is different), meaning that they are slightly correlated (on horizontally);
- an ascending consumed power trend is highlighted for the 2011-2013 period (with few deviations);
- load curves' correlation degree is reduced (time evolution and daily shape).

The results applying different methods are subjected to:

- forecasted values;
- differences from the known values (relative deviation in %);
- relative square deviation;
- performance index for each forecasted year (2014-2016). It is computed as the sum of relative square deviations for the 24 hourly values.

These results are presented in Tables 3 - 5 (FV - Forecasted value, RD - Relative deviation, RSD - Relative square deviation) for the 2014-2016 years and graphically in Figures 6-8.

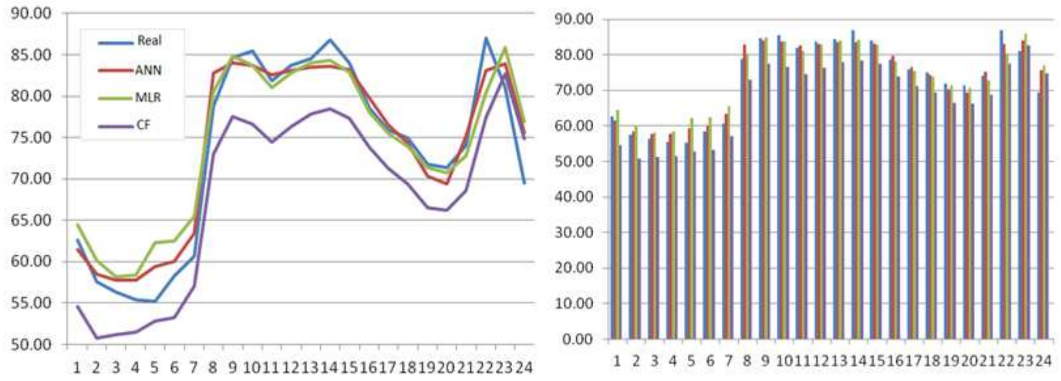


Figure 6: 2014 Forecasted value - comparative analysis [MW]

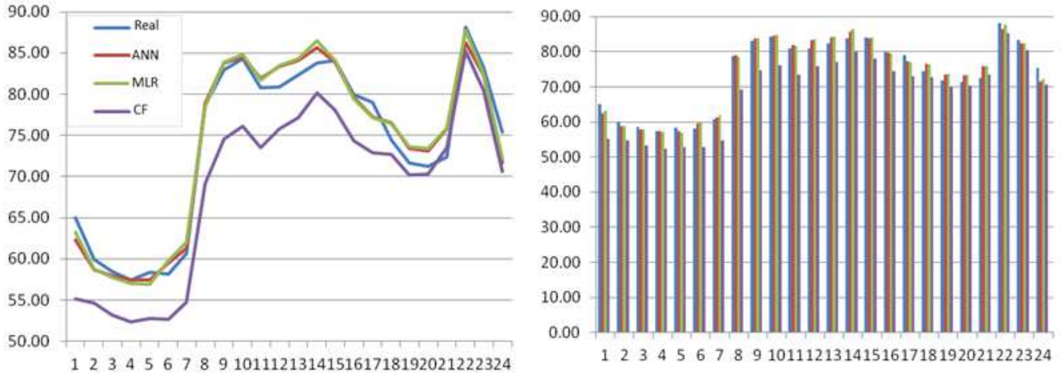


Figure 7: 2015 Forecasted value - comparative analysis [MW]

The performance indices have been gathered in Table 4. The forecast errors are presented in Table 7.

Comparative analysis of the results leads to the following conclusions:

Table 3: 2014 Year load curves forecasted [MW]

Hour	Know value	ANN			CF			MLR		
		FV	RD [%]	RSQ	FV	RD [%]	RSQ	FV	RD [%]	RSQ
1	62.6	61.51	-1.75	3.05	54.57	-12.83	164.5	64.49	3.03	9.16
2	57.6	58.49	1.54	2.37	50.79	-11.82	139.7	60.16	4.44	19.72
3	56.3	57.72	2.51	6.32	51.18	-9.10	82.83	58.16	3.31	10.96
4	55.4	57.73	4.21	17.73	51.49	-7.06	49.82	58.43	5.46	29.87
5	55.2	59.36	7.54	56.78	52.78	-4.39	19.25	62.31	12.88	165.8
6	58.3	59.99	2.90	8.39	53.19	-8.76	76.75	62.49	7.18	51.59
7	60.6	63.40	4.62	21.32	57.05	-5.87	34.41	65.47	8.04	64.59
8	78.8	82.76	5.03	25.28	73.04	-7.31	53.46	80.48	2.14	4.56
9	84.6	84.05	-0.66	0.43	77.52	-8.37	70.09	84.85	0.30	0.09
10	85.5	83.76	-2.03	4.13	76.62	-10.38	107.8	83.71	-2.09	4.37
11	81.9	82.53	0.77	0.60	74.48	-9.06	82.07	81.03	-1.06	1.12
12	83.7	83.13	-0.68	0.46	76.32	-8.82	77.78	82.78	-1.10	1.20
13	84.5	83.52	-1.16	1.36	77.82	-7.91	62.57	83.99	-0.60	0.37
14	86.8	83.60	-3.69	13.59	78.44	-9.64	92.83	84.29	-2.89	8.33
15	84.0	83.18	-0.98	0.96	77.37	-7.90	62.35	82.88	-1.33	1.77
16	78.6	79.76	1.48	2.19	73.81	-6.09	37.1	78.03	-0.72	0.52
17	75.9	76.47	0.75	0.56	71.22	-6.17	38.03	75.43	-0.62	0.38
18	75.0	74.39	-0.81	0.66	69.41	-7.45	55.54	73.91	-1.45	2.10
19	71.8	70.36	-2.01	4.03	66.54	-7.32	53.61	71.4	-0.56	0.31
20	71.4	69.39	-2.81	7.92	66.17	-7.32	53.64	70.78	-0.87	0.76
21	74.0	75.17	1.59	2.52	68.62	-7.27	52.82	72.78	-1.65	2.73
22	87.0	83.09	-4.50	20.21	77.46	-10.96	120.2	80.38	-7.60	57.82
23	81.0	83.95	3.65	13.29	82.64	2.03	4.12	85.93	6.08	36.99
24	69.5	75.60	8.78	77.11	74.84	7.69	59.13	76.92	10.68	113.9
		PI_{2014}		291.3	PI_{2014}		1650	PI_{2014}		589.1

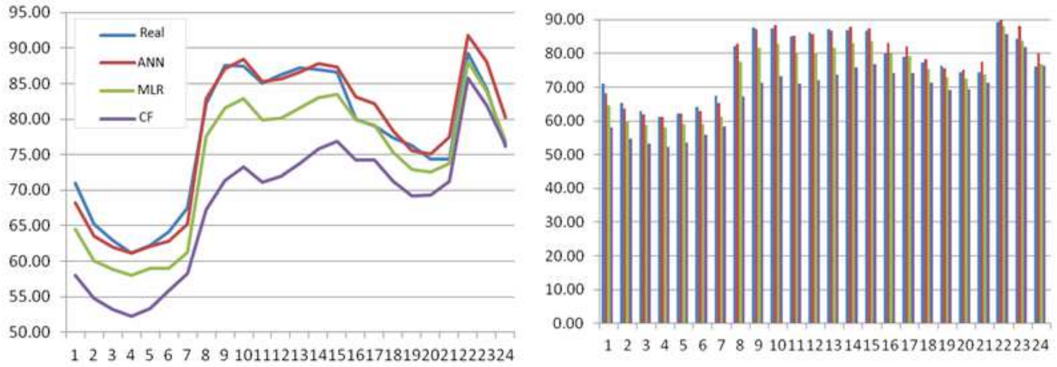


Figure 8: 2016 Forecasted value - comparative analysis [MW]

- classical forecasting methods (CF, MLR) are inadequate. The global performance index (Table 6) sustains this fact;

- ANN method: the best case is recorded for the 2015 year (Table 6: performance index

Table 4: 2015 Year load curves forecasted [MW]

Hour	Know value	ANN			CF			MLR		
		FV	RD [%]	RSQ	FV	RD [%]	RSQ	FV	RD [%]	RSQ
1	65.0	62.36	-4.05	16.44	55.14	-15.17	230.1	63.23	-2.72	7.42
2	59.9	58.70	-2.01	4.03	54.67	-8.73	76.23	58.83	-1.79	3.20
3	58.5	57.94	-0.95	0.90	53.22	-9.03	81.46	57.75	-1.28	1.63
4	57.4	57.47	0.13	0.02	52.40	-8.71	75.88	57.03	-0.64	0.40
5	58.4	57.42	-1.67	2.79	52.80	-9.59	91.95	56.95	-2.49	6.20
6	58.2	59.50	2.23	4.97	52.70	-9.45	89.31	59.83	2.80	7.84
7	60.7	61.31	1.00	1.00	54.70	-9.88	97.71	62.00	2.14	4.57
8	78.7	78.94	0.30	0.09	69.18	-12.10	146.5	78.49	-0.26	0.07
9	83.0	83.78	0.94	0.88	74.59	-10.13	102.6	83.85	1.02	1.05
10	84.3	84.49	0.23	0.05	76.14	-9.68	93.74	84.80	0.59	0.35
11	80.8	81.92	1.39	1.93	73.56	-8.96	80.22	81.70	1.11	1.23
12	80.9	83.40	3.09	9.56	75.76	-6.36	40.40	83.45	3.16	9.97
13	82.3	84.06	2.14	4.57	77.16	-6.25	39.02	84.31	2.45	5.98
14	83.8	85.64	2.20	4.84	80.13	-4.38	19.19	86.52	3.25	10.57
15	84.1	83.88	-0.26	0.07	78.05	-7.19	51.75	84.15	0.05	0.00
16	79.9	79.64	-0.33	0.11	74.34	-6.96	48.51	79.42	-0.60	0.36
17	79.0	77.24	-2.22	4.94	72.87	-7.76	60.21	77.12	-2.38	5.68
18	74.5	76.49	2.67	7.13	72.72	-2.40	5.74	76.43	2.59	6.72
19	71.7	73.39	2.36	5.58	70.15	-2.16	4.65	73.67	2.75	7.56
20	71.2	73.11	2.68	7.18	70.29	-1.28	1.63	73.43	3.13	9.81
21	72.4	75.83	4.74	22.5	73.56	1.60	2.56	75.89	4.82	23.24
22	88.1	86.31	-2.03	4.12	85.27	-3.21	10.32	87.72	-0.43	0.18
23	83.3	82.35	-1.14	1.30	80.44	-3.43	11.77	82.46	-1.01	1.01
24	75.4	71.64	-4.98	24.83	70.66	-6.28	39.46	72.24	-4.20	17.61
		PI_{2015}		129.8	PI_{2015}		1500	PI_{2015}		132.7

129.8); the worst case is recorded for the 2014 year (Table 6: performance index 291.3);

- the forecast errors are ranging between 4.7 % and 8.8 % in case of ANN based method. Values of 18 %, respectively 13 % have been obtained in case of CF, respectively MLR.

The load curves for a period of 10 years (2004-2013) are presented in Table 8, for the most significant summer day. These ones are corresponding to the entire network assembly of the considered distribution network operator. These data are going to be used in order to perform the forecast for 2014-2016 period.

The same information is graphically presented in Figure 9.

The load curves for 3 years period are presented in Table 9 and Figure 10. These ones are used in order to validate the performed forecast.

The trend is unclear for the 2004-2013 period. There are several increasing periods, alternating with decreasing ones, during a relatively short set of values. The load curves are not crossing each other. This means that they are characterized by a high correlation degree.

The 2014-2016 period is also characterized by an unclear evolution. The load curves correlation degree during a day is relatively good, taking into consideration their shape. Thus, it is envisaged that the ANN based forecasting methods could lead to better results.

Table 5: 2016 Year load curves forecasted [MW]

Hour	**Know value**	**ANN**			**CF**			**MLR**		
		FV	**RD [%]**	**RSQ**	**FV**	**RD [%]**	**RSQ**	**FV**	**RD [%]**	**RSQ**
1	71.0	68.29	-3.82	14.57	58.06	-18.23	332.2	64.50	-9.15	83.75
2	65.2	63.51	-2.60	6.74	54.80	-15.95	254.4	60.05	-7.91	62.49
3	63.0	62.00	-1.58	2.51	53.22	-15.52	240.9	58.90	-6.51	42.37
4	61.2	61.12	-0.12	0.02	52.20	-14.71	216.3	58.02	-5.20	27.01
5	62.2	62.17	-0.05	0.00	53.40	-14.15	200.2	59.02	-5.11	26.11
6	64.1	62.85	-1.95	3.79	55.82	-12.92	166.9	58.95	-8.04	64.58
7	67.5	65.21	-3.39	11.52	58.25	-13.70	187.8	61.30	-9.19	84.40
8	82.2	82.93	0.88	0.78	67.25	-18.19	330.9	77.56	-5.64	31.86
9	87.6	87.15	-0.52	0.27	71.34	-18.56	344.4	81.56	-6.89	47.51
10	87.5	88.45	1.08	1.18	73.25	-16.29	265.3	82.88	-5.28	27.87
11	85.0	85.3	0.36	0.13	71.12	-16.33	266.8	79.89	-6.01	36.08
12	86.3	85.62	-0.79	0.63	71.94	-16.64	276.9	80.13	-7.15	51.08
13	87.2	86.65	-0.63	0.40	73.79	-15.38	236.5	81.54	-6.50	42.19
14	87.0	87.79	0.90	0.81	75.82	-12.86	165.3	83.03	-4.56	20.79
15	86.6	87.29	0.80	0.64	76.85	-11.25	126.7	83.46	-3.63	13.19
16	80.0	83.18	3.98	15.80	74.21	-7.24	52.36	79.84	-0.20	0.04
17	79.0	82.21	4.06	16.52	74.26	-6.00	36.01	79.18	0.23	0.05
18	77.4	78.34	1.22	1.48	71.28	-7.90	62.45	75.29	-2.73	7.45
19	76.3	75.57	-0.96	0.91	69.18	-9.33	87.04	72.91	-4.44	19.73
20	74.3	75.12	1.11	1.22	69.31	-6.71	45.06	72.59	-2.30	5.28
21	74.4	77.49	4.16	17.30	71.25	-4.24	17.97	73.80	-0.81	0.65
22	89.3	91.78	2.78	7.71	85.83	-3.89	15.13	88.02	-1.43	2.06
23	84.2	88.10	4.64	21.5	81.88	-2.75	7.57	83.87	-0.39	0.15
24	76.1	80.20	5.39	29.02	76.35	0.33	0.11	76.94	1.10	1.21
		PI_{2016}		155.5	PI_{2016}		3935	PI_{2016}		697.9

Table 6: Performance indices - comparative analysis

	ANN	**CF**	**MLR**
PI_{2014}	291.3	1650	589.1
PI_{2015}	129.8	1500	132.7
PI_{2016}	155.5	3935	697.9
PI_{total}	576.6	7085	1420

Table 7: Maximum forecast errors

	ANN	**CF**	**MLR**
2014	8.78	12.83	12.88
2015	4.74	15.17	4.82
2016	5.39	18.23	8.19
Maximum	8.78	18.23	12.88

The same forecasting methods have been applied (as the ones presented in Tables 2-5). The

Table 8: Load curves for 2004-2013 period [MW]

Year/Hour	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
1	455.5	460.2	472.1	436.3	418.7	398.4	411.6	401.7	401.7	388.6
2	427.8	452.8	457.7	427.3	415.0	382.6	390.8	381.4	380.6	372.2
3	417.9	441.0	436.1	421.0	410.6	378.5	394.7	386.6	389.0	387.8
4	413.9	454.8	447.6	418.4	393.8	365.7	382.7	374.7	385.8	384.1
5	434.4	456.8	459.3	424.5	404.1	380.7	403.8	391.4	389.8	376.5
6	459.4	465.2	469.0	441.7	411.6	388.1	403.5	404.8	382.1	358.6
7	477.9	503.0	489.1	455.6	422.1	396.2	395.5	404.5	393.6	377.7
8	534.9	570.2	517.9	525.6	473.9	449.2	464.4	446.3	437.0	432.3
9	554.1	607.2	529.7	574.0	525.4	479.5	481.7	489.0	483.4	486.0
10	536.9	604.8	530.5	564.6	530.8	492.4	490.5	506.5	494.3	479.4
11	519.1	583.5	499.0	549.3	529.6	483.5	472.7	492.0	490.5	487.8
12	512.3	557.3	527.7	531.5	514.1	477.6	465.7	469.2	472.6	474.8
13	496.2	551.3	539.4	525.2	500.5	468.6	445.1	467.3	464.9	460.9
14	502.0	567.3	547.0	518.9	495.3	473.4	452.3	489.4	472.0	456.7
15	491.1	546.9	529.1	518.6	488.2	462.2	438.2	463.9	467.6	458.5
16	452.2	505.2	485.6	487.7	451.3	450.6	447.0	451.2	451.7	428.9
17	462.9	490.6	492.0	479.3	439.1	442.8	440.0	449.3	443.6	426.8
18	448.5	502.7	490.6	476.5	424.6	443.0	445.4	445.5	440.9	414.9
19	440.4	495.6	488.9	464.3	405.2	432.4	419.4	453.3	443.5	413.5
20	436.8	476.2	445.0	442.2	393.6	418.9	392.2	418.8	427.2	409.2
21	510.1	534.7	496.2	479.5	426.3	421.5	386.5	405.2	432.0	424.6
22	545.3	561.3	534.7	528.4	494.5	478.9	432.7	429.4	445.7	453.3
23	562.8	580.5	573.5	540.3	524.9	518.9	498.9	475.3	463.7	480.3
24	536.7	538.0	524.1	501.8	488.1	482.1	443.3	444.7	437.5	438.7

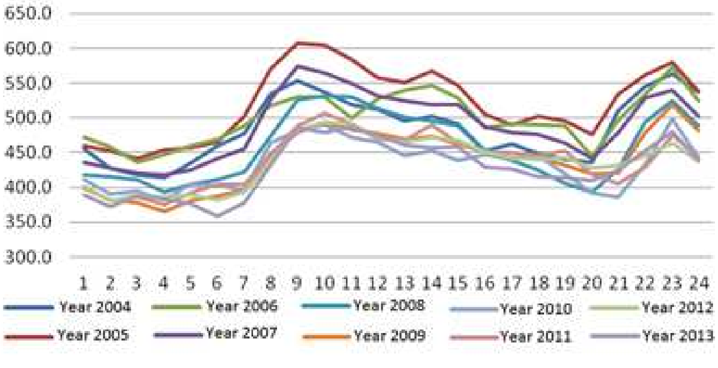


Figure 9: Known load curves for 2004-2013 period

extended results are not presented anymore. Only a synthesis of the forecasted values is presented in Table 10. The same results are graphically presented in Figures 11-13. The performance indices are synthesized in Table 11.

The conventional forecasting methods (CF and MLR) are totally inadequate (Table 11, global performance indices values). Among them, the MLR provides more accurate results.

Comparing the yearly performance indices for the ANN based results it is highlighting that the 2016 year the best situation is recorded (around 59). The worst one is recorded for the 2014

Table 9: Load curves for validation period 2014-2016 [MW]

Year/Hour	**2014**	**2015**	**2016**
1	389.4	384.8	384.2
2	374.7	370.3	364.6
3	377.4	380.9	373.9
4	370.8	364.9	363.3
5	370.6	367.4	372.5
6	368.2	358.1	364.3
7	382.0	381.3	380.1
8	430.4	421.4	422.7
9	464.1	450.9	449.9
10	482.8	472.9	471.2
11	490.3	486.2	481.9
12	460.6	453.9	452.7
13	449.0	442.3	450.7
14	459.8	454.9	450.5
15	453.0	450.5	441.4
16	433.3	431.1	430.0
17	437.5	440.2	433.8
18	427.2	418.0	423.0
19	419.9	409.6	415.9
20	408.3	402.2	405.6
21	410.9	411.6	406.0
22	450.0	439.8	436.4
23	472.6	474.6	467.7
24	438.6	439.4	434.6

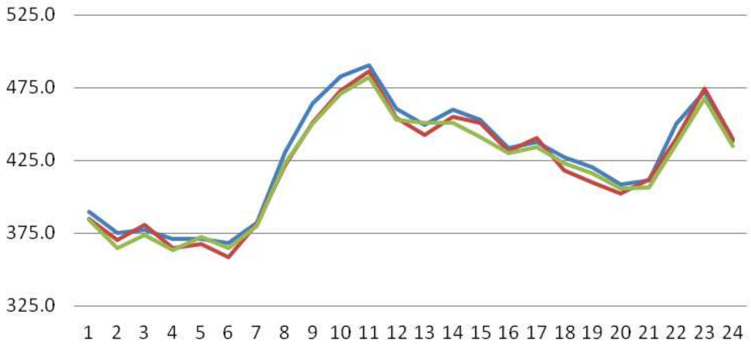


Figure 10: Load curves corresponding to the forecasted period

year (around 152).

The maximum forecast errors are presented in Table 12.

Based on the results presented in Table 12, acceptable values have been obtained for ANN based forecasting method (bellow 5 % errors). In case of CF and MLR forecasting based methods, they are ranging towards 17 % respectively 8 %.

The considered distribution system operator is divided into a number of 4 distribution branches. A comparison between the obtained values is performed in the following. Only the results corresponding to the ANN and MLR based forecasting methods have been considered.

Table 10: Performance indices - comparative analysis

2014			2015			2016		
ANN	**CF**	**MLR**	**ANN**	**CF**	**MLR**	**ANN**	**CF**	**MLR**
387.7	358.8	365.2	365.8	349.6	357.7	379.9	343.5	353.8
382.5	355.6	353.1	361.0	342.8	347.2	366.1	341.0	343.3
386.1	354.2	368.3	367.3	343.7	351.4	377.0	343.2	354.1
385.7	353.5	367.1	367.5	342.2	347.7	362.0	339.7	342.2
384.1	351.7	362.5	369.8	344.2	349.4	364.7	341.2	346.1
380.6	348.4	349.1	368.5	341.2	349.2	356.2	338.6	339.9
386.2	348.7	367.1	378.1	343.2	362.8	379.0	339.8	361.5
421.7	375.9	415.6	411.1	353.6	405.9	418.0	349.6	397.6
477.3	438.8	463.5	446.8	390.8	436.6	442.5	383.3	424.9
472.0	437.3	460.3	474.7	423.5	454.9	457.4	399.4	445.8
476.4	441.2	469.8	486.6	426.5	463.7	464.9	418.6	459.5
468.5	438.5	461.2	457.1	398.5	441.0	445.8	398.4	434.2
456.9	437.4	451.7	446.4	394.8	433.5	437.5	392.4	426.4
454.4	432.2	450.4	458.5	407.2	444.9	447.3	407.3	439.3
454.4	413.3	454.2	451.7	407.2	441.5	444.6	408.5	437.7
419.7	390.9	431.2	429.8	393.8	427.1	430.0	392.5	423.3
419.2	393.0	431.5	431.8	402.3	432.9	437.9	406.5	433.1
409.9	384.0	423.5	422.0	396.1	426.3	419.6	388.3	416.2
409.2	386.9	424.3	415.3	392.5	422.3	412.4	384.8	411.0
402.5	387.6	422.7	406.8	386.1	414.6	405.8	381.6	406.7
420.3	407.8	437.9	410.0	392.3	418.9	414.8	394.2	416.7
454.9	440.6	464.5	443.8	435.0	454.1	438.6	425.4	442.7
485.9	461.3	489.7	464.1	452.8	475.6	461.6	453.6	474.6
439.7	436.9	456.6	432.2	433.9	449.1	439.1	433.5	446.8

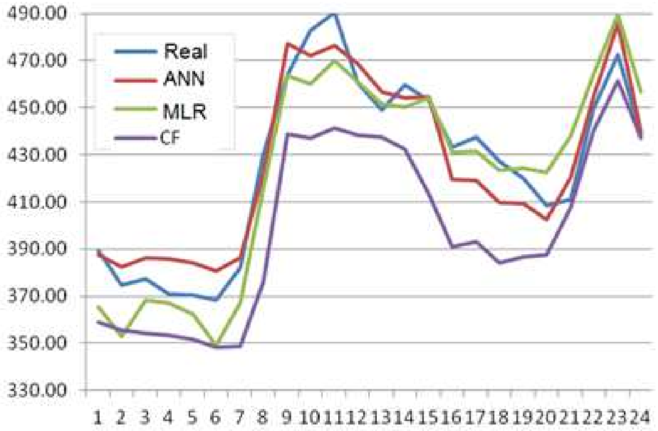


Figure 11: 2014 year forecasted load curve

The specific performance indices have been synthesized in Table 13. These indices have been obtained by dividing the performance index with 24 (3 years x 24 hours = 72).

The distribution system operator entire network assembly is ranking on the 1st place. It is a logic result: the errors are attenuating through summation, due to the opposite signs. It closely

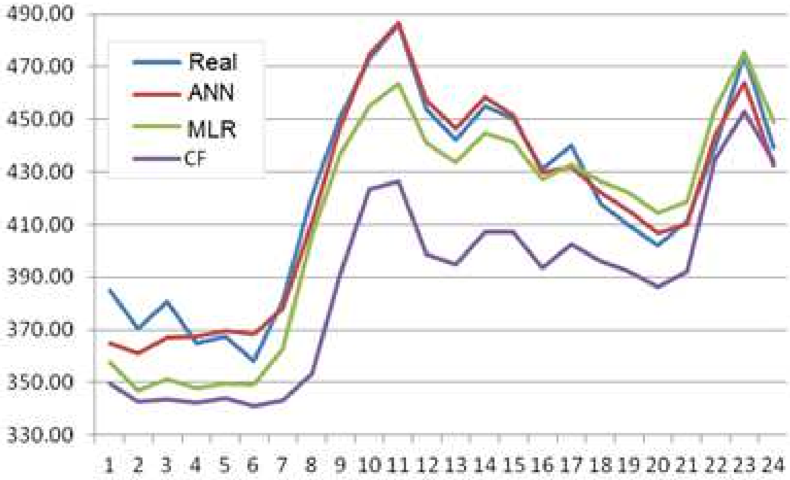


Figure 12: 2015 year forecasted load curve

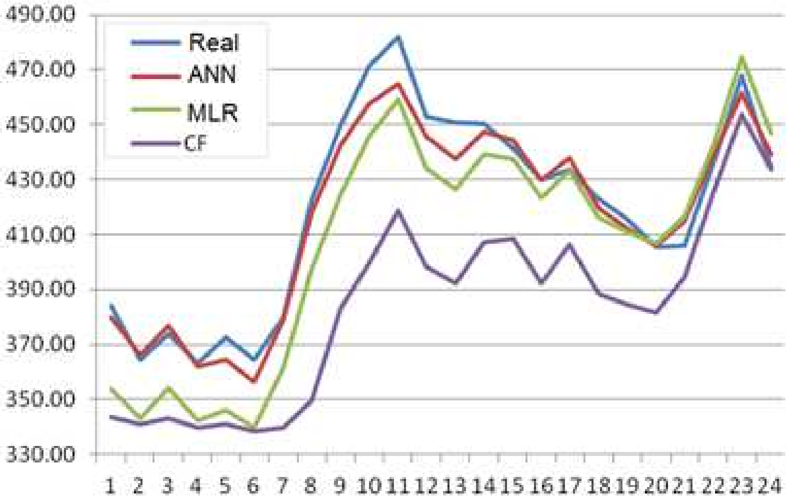


Figure 13: 2016 year forecasted load curve

Table 11: Performance indices - comparative analysis

	ANN	**CF**	**MLR**
PI_{2014}	151.5	1200	281.0
PI_{2015}	78.89	1855	350.9
PI_{2016}	58.72	2176	473.1
PI_{total}	289.1	5231	1105

followed up by Distribution Branch TM and, at a considered distance, by Distribution Branch AR and Distribution Branch HD. Distribution Branch CS is ranking on the last place.

The Distribution Branch TM is ranking on the 1st place, if the MLR based results are discussed. The entire network assembly is following up. The Distribution Branches AR and HD are the next ones. The Distribution Network CS is also ranking on the last place. The 1st two places have been changed comparing with the ANN based hierarchy. The ANN specific indices

Table 12: Maximum forecast errors

	ANN	**CF**	**MLR**
2014	4.17	12.67	6.56
2015	4.94	16.09	7.75
2016	3.52	17.30	7.90
Maximum	4.94	17.30	7.90

Table 13: Specific performance indices

No	**Distribution branch**	**ANN**		**MLR**		**Specific indices ratio**
		Total	**Specific**	**Total**	**Specific**	
1	Entire network assembly	298.1	4.02	1105	15.35	3.82
2	Distribution branch TM	354.1	4.92	958.5	13.31	2.71
3	Distribution branch AR	576.6	8.01	1420	19.72	2.46
4	Distribution branch HD	821.5	11.41	2690	37.36	3.27
5	Distribution branch CS	2278	31.64	10530	146.2	4.63

are covering a large domain, from 4.02 (entire network assembly), to 31.64 (Distribution Branch CS). Their values are influenced by the input data (2004-2013), the load curves' correlation degree and the real consumption evolution (2014-2016).

High values for the specific performance index highlight the presence of possible wrong load consumption data (discrepancy is recorded). An acceptable situation could be obtained if they are eliminated (corrected).

The maximum forecast errors are presented in Table 14.

Table 14: Maximum forecast errors

Nr. crt	**Distribution branch**	**ANN**	**CF**	**MLR**
1	Entire network assembly	4.94	17.30	7.90
2	Distribution branch TM	5.11	21.50	8.01
3	Distribution branch AR	8.78	18.23	12.88
4	Distribution branch HD	8.11	15.63	12.28
5	Distribution branch CS	14.91	45.26	33.33

The provided conclusions, regarding the quality of the results are sustained. The ones provided by the ANNs are the most acceptable ones. Among the conventional methods, the MLR based ones are the most suitable.

5 Conclusion

Accurate load forecasts are critical for distribution planning, for utilities. The quality of the forecast methods depends on the available historical data as well as on the knowledge about the main influence parameters on the energy consumption.

Different forecasting methodologies have been integrated into a stand-alone application with a graphical user interface. The authors have applied these methodologies to obtain hourly load forecasts (for next 24 hours). A good performance and reasonable prediction accuracy was achieved for NN model.

The historical data preprocessing could be improved in order to obtain better results.

The developed software tool deals with real data; it leads to accurate consumed power forecasts. The conclusion is sustained through comparison performed with real monitored data for the same period.

The results are practically confirming the performed comments corresponding to the 2004-2013, respectively 2014-2016 periods.

Classical forecasting methods are not recommended to be applied. But, in case of MLR method a slight advantage is highlighted. Smallest forecasting errors have been obtained in case of ANN based method.

Bibliography

[1] Charytoniuk, W.; Chen, M.S.; Van Olinda, P. (1998). Nonparametric Regression Based Short-Team Load Forecasting, *IEEE Transaction on Power Systems*, 13(3), 735-730, 1998.

[2] Chen, H.; Canizares, A.C.; Ajit, S. (2011). ANN based Short-Term Load Forecasting in Electricity markets, *Proceedings of the IEEE Power Engineering Society Winter Meeting*, 2, 411-415, 2011.

[3] Chen, J.F.; Wang, W.M.; Huang, C.M.(2005). Analysis of an adaptive time-series autoregressive moving-average (ARMA) model for short-term load forecasting, *Electric Power Systems Research*, 34, 187-196, 2005.

[4] Chis, V.; Barbulescu, C., Kilyeni, S.; Dzitac S. (2018). Short-Term Load Forecasting Software Tool, *Proceedings of the 7th International Conference on Computers Communications and Control (ICCCC)*, 111–118, 2018.

[5] Cho, M.Y.; Hwang, J.C.; Chen, C.S. (1995). Customer short-term load forecasting by using ARIMA transfer function model, *Proceedings of the International Conference on Energy Management and Power Delivery*, 317-322, 1995.

[6] Danladi, A.; Yohanna, M.; Puwu, M.I.; Garkida, B.M. (2016). Long-term load forecast modelling using a fuzzy logic approach, *Pacific Science Review A: Natural Science and Engineering*, 18(2), 123-127, 2016.

[7] Ho, K.l.; Hsu, Y.I.; Chen, C.F.; Lee, T.E.; Liang, C.C.; Lai, T.S.; Chen, K.K. (1990). Short Term Load Forecasting of Taiwan Power System Using a Knowledge Based Expert System, *IEEE Transactions on Power Systems*, 5(4), 1214-1221, 1990.

[8] Hong, W.C.; Dong, Y.; Chen, L.Y.; Wei, S.Y. (2012). Seasonal Support vector Regression with Chaotic Genetic Algorithm in Electric Load, *ICGEC 6th International Conference on Genetic and Evolutionary Computing*, 124-127, 2012.

[9] Hyndman, R.J.; Koehler, A.B. (2016). Another look at measuring forecast accuracy, *International Journal of Forecasting*, 22(2), 679-688, 2016.

[10] Ismail, Z.; Efendy, R. (2011). Enrollment forecasting based on modified weight fuzzy time series, *Journal of Artificial Intelligence*, 4(1), 110-118, 2011.

[11] Jin, X.; Dong, Y.; Wu, J.; Wang, J. (2010). An Improved Combined Forecasting Method for Electric Power Load Based on Autoregressive Integrated Moving Average Model, *International Conference of Information Science and Management Engineering (ISME)*, 2, 476-480, 2010.

[12] Karapidakis, S. (2007). Machine learning for frequency estimation of power systems, *Applied Soft Computing*, 7(1), 105-114, 2007.

[13] Mordjaoui, M.; Haddad, S.; Medoued, A.; Laouafi, A. (2017). Electric load forecasting by using dynamic neural network, *Journal hydrogen Energy*, 42, 17655-17663, 2017.

[14] Pandian, S.C.; Duraiswamy, K.; Rajan, C.C.A. (2006). Fuzzy approach for short term load forcasting, *Electric Power Systems Research*, 76, 541-548, 2006.

[15] Park, D.C.; El-Sharkawi, M.A.; Marks, R.J.; Atlas, L.E.; Damborg, M.J. (1991). Electric load forecasting using an artificial neural network, *IEEE Transactions on Power Systems*, 6(2), 442-449, 1991.

[16] Schellong, W. (2011). *Energy Demand Analysis and Forecast, Energy Management Systems P. Giridhar Kini*, IntechOpen, DOI: 10.5772/21022, 2011.

[17] Sheikh, S.K.; Unde, M.G. (2012). Short-Term Load Forecasting Using ANN Technique, *International Journal of Engineering Sciences & Emerging Technologies*, 1(2), 97-107, 2012.

[18] Shelke, M.; Thakare, P.D. (2014). Short Term Load Forecasting by Using Data Mining Techniques, *International Journal of Science and Research (IJSR)*, 3(9), 1363-1367, 2014.

[19] Singh, P.; Dwivedi, P. (2018). Integration of new evolutionary approach with artificial neural network for solving short term load forecast problem, *Journal of Applied Energy*, 217, 537-549, 2018.

[20] Srinivasan, D.S.; Tan, S.S.; Cheng, C.S.; Chan, E.K. (1999). Parallel neural network-fuzzy expert system strategy for short-term load forecasting: system implementation and performance evaluation, *IEEE Transactions on Power Systems*, 14(3), 1100-1106, 1999.

[21] Yang, H.P.; Yan, F.F.; Wang, H.; Zhang, L.(2016). Short-term load forecasting based on data mining, *IEEE 20th International Conference on Computer Supported Cooperative Work in Design*, 170-173, 2016.

[22] Zhang, J.,; Yi-Ming, W.; Dezhi, L.; Zhongfu, T.; Jianhua, Z. (2018). Short term electricity load forecasting using a hybrid model, *Journal Energy*, 158(C), 774-781, 2018.

[23] [Online]. Available: www.mathworks.com Matlab Users guide, Accesed on 12 February 2018.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(6), 956-971, December 2018.

Multi-Objective Tabu Search to Balance Multihoming Loads in Heterogeneous Wireless Networks

J. A. Huertas, Y. Donoso

Jorge A. Huertas
Center for Optimization and Applied Probability (COPA)
Industrial Engineering Department
Universidad de los Andes, Bogotá, Colombia
huertas.ja@uniandes.edu.co

Yezid Donoso*
Computing and Systems Engineering Department
Universidad de los Andes, Bogotá, Colombia
*Corresponding author: ydonoso@uniandes.edu.co

Abstract: The advantages of the increasing usage of mobile devices that operate under the multihoming scheme are changing the communications world drastically. Therefore, next generation networks operators have the challenging task to distribute connections of mobile devices efficiently over their access networks, creating a big heterogeneous wireless network for telecommunications. We present a mixed integer-linear programming (MILP) model to balance the load of multiple services over wireless networks taking into account three key indicators: connection loads of access networks, connection costs, and battery consumption of connections. To solve the multi-objective problem, we propose a multi-objective Tabu Search procedure that is capable to find non-supported solutions in the online efficient set. To test the performance of our multi-objective Tabu Search, we tested it over four instances of the literature. In the first instance, a small instance, our procedure finds the true efficient set of solutions. For the other three instances, large instances with over a thousand mobile devices, our procedure finds good online efficient sets of solutions in less than 30 seconds. Finally, using appropriate multi-objective metrics, we compare the results of our multi-objective Tabu Search against the results of a *state of the art* multi-objective genetic algorithm in the literature for the same problem, outperforming the genetic algorithm in every instance tested.

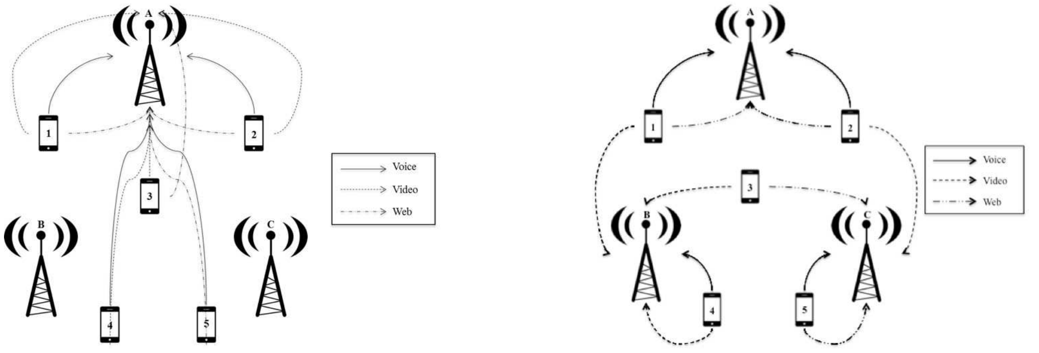
Keywords: Heterogeneous networks, Multihoming, Vertical handover, Optimization, Multi-objective, Tabu Search.

1 Introduction

Multihoming refers to the ability mobile devices have to connect to different access networks at a time through multiple network interfaces [12]. On the other hand, vertical handover (VHO) is the ability to change the access network that provides any service, while maintaining its quality [10]. Multihoming facilitates VHO, making it more seamless to the user [11]; and enhances the flow of data of multiple services across various network interfaces [2], allowing prioritized data to flow through next generation networks (e.g., 4G), and less important data to flow through legacy networks. This behaviour converges towards heterogeneous wireless networks (HWNs).

Thanks to the exponential growth of mobile devices that operate under a multihomming scheme, authorities operating access networks must rely on tools capable to perform VHO processes, while ensuring users' quality of service (QoS). Standards such as IEEE 802.21 MH provide only the framework for performing VHO, yet the decision-making algorithms to do so are an open discussion topic among the research world [15].

The methods in charge to perform VHO in a WHN must recover useful information from mobile devices to decide how to efficiently distribute the connections across different available access points. To illustrate this issue, assume that an operator has three access networks (i.e., A, B, and C), and there are five mobile devices using at least one of three services (i.e., Web, Video, and Voice). A possible bad “distribution" of the connections is presented in Figure 1(a), where all the services are provided by the same access network, which may not have enough bandwidth to do so. Assuming that all services in our example demand the same bandwidth, have the same connection cost, and generate the same power consumption, Figure 1(b) presents a balanced distribution of the connections, which is a better scenario for QoS.



(a) (b)

Figure 1: Connections example: (a) Possible distribution of connections; (b) Balanced solution

In this paper we present an optimization model to achieve a balanced distribution of connections to multiple access networks, taking into account three key indicators: load of access networks, cost of connections, and battery consumption due to connections. To solve the model we propose a multi-objective Tabu Search heuristic. We test our procedure under different small and large-scale instances and compare our results against a genetic algorithm in the literature.

The remainder of this paper is organized as follows: Section 2 discusses the relevant related work in the literature; Section 3 formally introduces the problem description; Section 4 presents the proposed multi-objective Tabu Search; Section 5 presents the computational experiments, as well as its results; finally, Section 6 concludes the paper and outlines future work.

2 Literature review

Multihoming and Load Balancing strategies have been topics of several research projects. For example, [3] presents a model that minimizes the load of networks by re-allocating services from the more loaded network. [1] presents a queuing model that combines multihoming and network coding to improve the allocation of network resources and the communication performance in heterogeneous technologies. [4] proposes an optimization model to perform fairness balance in WHS. Additionally, [9] extended the previous work to a multi-objective model to balance WHNs considering service loads, connection cost, and battery consumption. The two former models only balance the services in the intersection of the access networks’ coverage zone. We attempt to extend these models by balancing services in the union of all the coverage zones of the access networks.

Multiple meta-heuristics have proven to be an excellent way to solve combinatorial problems. Simulated Annealing algorithms mimic the cooling processes of metals to find good solutions. Particle Swarm algorithms reproduces the movements of swarms across the globe to solve problems. Genetic algorithms combine the way we evolve to find good solutions across generations. All of these heuristics and meta-heuristics reproduce the nature's processes to find solutions to specific problems. [3] presents multiple heuristics to solve problems related to computer networks. For the load balancing across heterogeneous networks problem [9] uses a multi-objective genetic algorithm to obtain an efficient set of solutions across generations. Unlike them, we specifically focus on Tabu Search to solve the problem.

Introduced by [7], Tabu Search (TS) allows to search the solution space and enables early stopping of algorithms due to local optima. It has been applied to solve a large variety of problems, including set covering problems.

Regarding wireless sensor netowrks (WSNs) problems, numerous work has been accomplished using Tabu Search meta-heuristic. [14] provides a TS-based routing protocol (TSRP) for data routing in WSNs. [6] proposes a centralized clustering method for a data collection mechanism in WSNs that seeks the optimization of the consumption of energy while collecting data. They achieve their mechanism through a Tabu Search meta-heuristic that returns better solutions in terms of cost and execution than other meta-heuristics such as simulated annealing or distributed methods.

Regarding the multi-objective nature of problem we address, [8] provides a framework for the usage of Tabu Search in WSNs problems with multiple objectives. It also addresses multiple WSNs related problems, such as the wireless LAN planning problem (WLP problem), and the evaluation of wireless sensor networks. Using multi-objective Tabu Search meta-heuristics, the authors achieve high-quality solutions regarding the Paretto Optimal frontier, proving that this type of heuristic is suitable to handle problems with numerous combinatorial and continuous variables, such as the ones related to WSNs.

As showed in the previous paragraphs, although many WSN-related problems have been solved using a Tabu Search approach, having extremely good results, many HWN-related problems have been tackled using other types of heuristics. We attempt to fill this gap by balancing multiple service loads in a WHN using a multi-objective Tabu Search procedure.

3 Problem description

We proposed a formulation based on the model presented in [4] and extended to consider multiple objectives in [9]. While both models balance connections from services in the intersection of the access networks' coverage zones, we do so in the union of the available access networks' coverage space.

Let $\mathcal{N}$, $\mathcal{K}$, and $\mathcal{I}$ be the set of the available access networks, mobile devices, and all services in use, respectively. Subset $\mathcal{S}_k$ references the services being used in mobile device $k \in \mathcal{K}$. let c_j and w_j be the cost of connection to the access network $j \in \mathcal{N}$ and the theoretical bandwidth of network $j \in \mathcal{N}$, respectively. Let f_k be the maximum cost the user of mobile device $k \in \mathcal{K}$ is willing to pay to connect to any access network. Let d_i be the demanded bandwidth by service $i \in \mathcal{I}$.

Let m_s be the minimum signal strength required by any mobile device to connect to any network. Let s_{kj} be the signal strength perceived by mobile device $k \in \mathcal{K}$ from network $j \in \mathcal{N}$. To model the power consumption generated in a mobile device when connected to a network we use a power consumption indicator that relates the signal strength perceived by the device. Let o_{kj} be the indicator of power consumption generated in mobile device $k \in \mathcal{K}$ if connected to network $j \in \mathcal{N}$. The power consumption indicator is inversely proportional to the signal strength perceived; it is low (i.e., 1) if the perceived signal is above a signal upper limit u_s; high (i.e., 3) if the perceived signal is below a signal lower limit l_s; and medium (i.e., 2) if it lays between l_s and u_s. Eq. 1 describes how the power consumption is defined.

$$o_{kj} = \begin{cases} 1, & \text{if} \quad u_s < s_{kj}; \\ 2, & \text{if} \quad l_s \leq s_{kj} \leq u_s; \\ 3, & \text{if} \quad l_s > s_{kj}; \end{cases} \qquad \forall\, k \in \mathcal{K}, j \in \mathcal{N}. \tag{1}$$

Let p_k be the percentage of battery remaining in mobile device $k \in \mathcal{K}$. Let b_k be an indicator of the battery percentage remaining in mobile device $k \in \mathcal{K}$; it takes a high value (i.e., 3) if the battery percentage p_k is above a battery-percentage upper limit u_b; it takes a low value (i.e., 1) if p_k is bellow a battery-percentage lower limit l_b; finally, it takes a medium value (i.e., 2), if p_k is between l_b and u_b. Eq. 2 describes how the battery percentage indicator is defined.

$$b_k = \begin{cases} 1, & \text{if} \quad l_b > p_k; \\ 2, & \text{if} \quad l_b \leq p_k \leq u_b; \\ 3, & \text{if} \quad u_b < p_k; \end{cases} \qquad \forall\, k \in \mathcal{K}. \tag{2}$$

This model considers x^i_{kj}, a binary decision variable that takes the value of 1, if the mobile device $k \in \mathcal{K}$ connects to network $j \in \mathcal{N}$ to satisfy service $i \in \mathcal{S}_k$; 0, otherwise. Also, it considers y_{kj}, another binary decision variable that takes the value of 1, if mobile device $k \in \mathcal{K}$ connects to network $j \in \mathcal{N}$ to satisfy any of its services $i \in \mathcal{S}_k$; 0, otherwise. Finally, α, β, and γ are non-negative decision variables representing the maximum load of the networks, the maximum connection cost to the networks, and the maximum consumption indicator from connections to the networks, respectively. Thus the formulation of our MILP is as follows:

$$\text{minimize} \quad \alpha \tag{3}$$

$$\text{minimize} \quad \beta \tag{4}$$

$$\text{minimize} \quad \gamma \tag{5}$$

Subject to,

$$
\begin{aligned}
y_{kj} &\leq \sum_{i\in\mathcal{S}_k} x^i_{kj}, && \forall\ k\in\mathcal{K}, j\in\mathcal{N}; && (6)\\
x^i_{kj} &\leq y_{kj}, && \forall\ k\in\mathcal{K}, j\in\mathcal{N}, i\in\mathcal{S}_k; && (7)\\
\sum_{j\in\mathcal{N}} x^i_{kj} &= 1, && \forall\ k\in\mathcal{K}, i\in\mathcal{S}_k; && (8)\\
d_i\cdot x^i_{kj} &\leq w_j, && \forall\ k\in\mathcal{K}, j\in\mathcal{N}, i\in\mathcal{S}_k; && (9)\\
c_j\cdot y_{kj} &\leq f_k, && \forall\ k\in\mathcal{K}, j\in\mathcal{N}; && (10)\\
o_{kj}\cdot y_{kj} &\leq b_k, && \forall\ k\in\mathcal{K}, j\in\mathcal{N}; && (11)\\
m_s\cdot y_{kj} &\leq s_{kj}, && \forall\ k\in\mathcal{K}, j\in\mathcal{N}; && (12)\\
\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{S}_k}\frac{d_i}{w_j}\cdot x^i_{kj} &\leq \alpha, && \forall\ j\in\mathcal{N}; && (13)\\
\sum_{k\in\mathcal{K}} c_j\cdot y_{kj} &\leq \beta, && \forall\ j\in\mathcal{N}; && (14)\\
\sum_{k\in\mathcal{K}} o_{kj}\cdot y_{kj} &\leq \gamma, && \forall\ j\in\mathcal{N}; && (15)\\
x^i_{kj} &\in \{0,1\}, && \forall\ k\in\mathcal{K}, j\in\mathcal{N}, i\in\mathcal{S}_k; && (16)\\
y_{kj} &\in \{0,1\}, && \forall\ k\in\mathcal{K}, j\in\mathcal{N}; && (17)\\
\alpha,\beta,\gamma &\geq 0. && && (18)
\end{aligned}
$$

Objective function (3) minimizes the maximum load of the networks, objective function (4) minimizes the maximum connection cost to a network, and objective function (5) minimizes the maximum battery consumption from connections to the networks. Constraints (6) and (7) relate decision variables y_{kj} and x^i_{kj} by determining if a mobile device connects to a network to satisfy the demand of one of its services, or not. Nevertheless, to ensure that every service in the mobile devices is served, constraints (8) guarantee the accessibility of the services in every mobile device to a network. Constraints (9) guarantee that the services being used can only be provided by networks with enough bandwidth to meet its demand, not overloading any network. Constraints (10) guarantee that the connection cost to a network does not exceed the maximum connection cost established by the user of the mobile device. Constraints (11) ensure that the consumption indicator generated by the connection to a network does not exceed the battery level of a mobile device. One of the big contributions of this paper is the ability to guarantee accessibility to the services being used in all the mobile devices located in the overall coverage area of the access networks. Constraints (12) guarantee the connection of a mobile device to a reachable network. Constraints (13) capture the maximum load of the available networks. Constraints (14) capture the maximum connection cost to the networks. Constraints (15) capture the maximum power consumption indicator due to connection to networks. Finally, constraints (16)-(18) define the nature of the decision variables.

4 Multi-objective Tabu Search approach

Tabu Search (TS) heuristic explores the solution space of a problem in an iterative way, prohibiting previous solutions to be visited during a number of iterations; and storing some of these previous solutions (which is a good aspect to extend to the multi-objective procedure). We propose a multi-objective TS (MOTS) procedure for solving the model described in Section 3.

The remainder of this section is organized as follows: Sub-section 4.1 introduces the encoding of our MOTS. Sub-section 4.2 shows the move operator. Sub-section 4.3 presents the memory management used. Sub-section 4.4 defines the dominance criteria for our MOTS. Sub-Section 4.5 discusses some of the diversification and intensification strategies used. Sub-section 4.6 presents the pseudo-codes of the MOTS. Finally, Sub-section 4.7 presents appropriate measures to devise the quality of the MOTS.

4.1 Encoding

In the problem defined in Section 3, the decision variable that defines the structure of the problem is x^i_{kj}; it tells how the connections and the multihoming are managed. The other decision variables depend on this binary decision variable. The encoding of the solution is guided by this variable; therefore, we encapsulate the values of the solution in a three-dimensional array $\mathbf{x}$ as the one presented in Fig. 2.

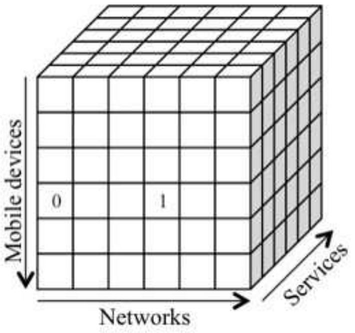


Figure 2: Encoding of the solution

The structure of $\mathbf{x}$ is faithful to the variable notation: the first index (i.e., the rows) references the mobile devices; second index (i.e., columns), the networks; third index (i.e., depth), the services. Each cell of the array can take the values defined by the variable x^i_{kj}, and the value of the remaining decision variables can be obtained by iterating $\mathbf{x}$. For example, in Fig. 2, the fourth mobile device is connecting to the fourth access network to provide the first service (i.e. $x^1_{4,4} = 1$); no other access network can provide this service for this mobile device (i.e., $x^1_{4,j} = 0, \forall\, j \neq 4$).

4.2 Move operator

Constraints (8) guarantee that every service in use in every mobile device gets connected to a network. To guarantee such constraint, we defined a *swap* move. More formally, any possible swap move has the form:

$$swap(x^i_{kj}, x^i_{kj\prime}), \qquad \forall\, k \in \mathcal{K}, i \in \mathcal{S}_k, j, j\prime \in \mathcal{N} \mid x^i_{kj} = 1 \wedge x^i_{kj\prime} = 0. \tag{19}$$

Move operator (19) defines the neighbourhood of the current solution $\mathbf{x}$. With this definition, the neighbourhood of $\mathbf{x}$ includes both feasible and infeasible reachable solutions. In our MOTS we only move to a feasible solution, due to the convexity of the model defined in Section 3.

4.3 Memory management

Managing short-term and long-term memory in TS is crucial for a good performance of the procedure [7]. For short-term memory we defined a *temporary* list of tabued moves $\mathbf{T}$ given the chronological realizations of the swap move. At any iteration of the algorithm, a new cell in $\mathbf{x}$ takes the value of 1. We prohibit the removal of the new cell (i.e., the cell takes the value of 0) during certain number of iterations. More formally, the tabu move (k, j, i, l) tells a cell x^i_{kj} cannot be zero (i.e., $x^i_{kj} \neq 0$) during l iterations. Constant insertion and deletion of tabu moves to the temporary list allows the exploration of the solution space over solutions that have not been reachd yet; this is a key process in any meta-heuristic [3].

On the other hand, some services won't be used in some devices, because of subset $\mathcal{S}_k, \forall\, k \in \mathcal{K}$. Additionally, a device $k \in \mathcal{K}$ cannot connect to a given network $j \in \mathcal{N}$ if the perceived signal s_{kj} is below m_s, or the connection cost c_j is above f_k. These cases are checked through a *constant* tabu list that allows predefined infeasibility to be taken into account. Some of the impossible connections are preprocessed for the procedure, and taken into account with a dynamic 3D array to preserve the dimensionality of the WSNs related problems.

4.4 Dominance

Let $\alpha(\mathbf{x})$, $\beta(\mathbf{x})$ and $\gamma(\mathbf{x})$ be the associated values of the maximum load of a network, the maximum connection cost to a network, and the maximum consumption of battery level for connecting to a network in solution $\mathbf{x}$, respectively. Let $\mathbf{x}$ and $\mathbf{x\prime}$ be two different feasible solutions to the problem. Solution $\mathbf{x}$ dominates $\mathbf{x\prime}$ if

$$\begin{array}{l}(\alpha(\mathbf{x}) < \alpha(\mathbf{x\prime}) \textit{ and } \beta(\mathbf{x}) \leq \beta(\mathbf{x\prime}) \textit{ and } \gamma(\mathbf{x}) \leq \gamma(\mathbf{x\prime})) \textit{ or}\\ (\alpha(\mathbf{x}) \leq \alpha(\mathbf{x\prime}) \textit{ and } \beta(\mathbf{x}) < \beta(\mathbf{x\prime}) \textit{ and } \gamma(\mathbf{x}) \leq \gamma(\mathbf{x\prime})) \textit{ or}\\ (\alpha(\mathbf{x}) \leq \alpha(\mathbf{x\prime}) \textit{ and } \beta(\mathbf{x}) \leq \beta(\mathbf{x\prime}) \textit{ and } \gamma(\mathbf{x}) < \gamma(\mathbf{x\prime})).\end{array} \tag{20}$$

Additionally, solution $\mathbf{x}$ alternates with $\mathbf{x\prime}$ if

$$\alpha(\mathbf{x}) = \alpha(\mathbf{x\prime}) \textit{ and } \beta(\mathbf{x}) = \beta(\mathbf{x\prime}) \textit{ and } \gamma(\mathbf{x}) = \gamma(\mathbf{x\prime}) \tag{21}$$

A solution $\mathbf{x}$ is efficient when there is no other solution $\mathbf{x\prime}$ that dominates it.

On the other hand, an efficient solution can be either *supported* or *non-supported.* Supported solutions are found through the linear convex combination of the objective functions (i.e., weighted sum of objectives). Non-supported efficient solutions cannot be found through the weighted sum of objective functions. For further information on multiobjective concepts, the interested reader is referred to [3, 5].

We defined a set of current solutions $\bar{\mathbf{S}} = \{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^v\}$ that changes iteratively depending on the *swap* operator. At the end of any iteration, we obtain the efficient solutions $\mathbf{S}$ in $\bar{\mathbf{S}}$ (i.e., $\mathbf{S} \subseteq \bar{\mathbf{S}}$). After n iterations, the algorithm stops, finding the *online* efficient set $\mathbf{S}$, which is the set containing all the efficient solutions found after n iterations of the procedure.

4.5 Search strategies

Diversification strategies in TS are conceived to explore the solution space that has not been visited until a given iteration. Intensification strategies are made to guide the search through solutions found better historically [7]. To generate both diversification and intensification strategies in our MOTS, at any iteration we move to a random non-dominated solution of the current neighbourhood. With this decision we explore the remaining solution space, because the new solution is non-dominated, while searching good solutions, for the same reason.

4.6 Pseudo-codes

Algorithms 1, 2 and 3 show the pseudo-codes of our MOTS. In Algorthm 1, we define a set of current solutions (i.e., Line 3.) that is iteratively mutated (i.e., Line 6.) to explore the whole search space. Additionally, iteratively we compare the set of current solutions with online efficient set (i.e., Line 7.) to replace it. In Algorithm 2, the random component of the procedure is taken into account in Line 3. The mutation of the current solution is performed in Lines 4. to 8.; also, the addition of the new tabu move is executed in these lines as well (i.e., Line 7.). Finally, during the dominance check in Algorithm 3 we replace the online efficient set in Line 23.

Algorithm 1 Multi-objective Tabu Search

```
Input: $n, v$
Output: $\mathbf{S}$
 1: procedure MOTS($n, v$)
 2:     $\mathbf{S}, \mathbf{T} \leftarrow \emptyset$
 3:     $\bar{\mathbf{S}} \leftarrow \textsc{generateInitialSolutions}(v)$
 4:     $\mathbf{S} \leftarrow \textsc{checkDominance}(\mathbf{S}, \bar{\mathbf{S}})$
 5:     for $it < n$ do
 6:         $\bar{\mathbf{S}} \leftarrow \textsc{move}(\bar{\mathbf{S}}, \mathbf{T})$
 7:         $\mathbf{S} \leftarrow \textsc{checkDominance}(\mathbf{S}, \bar{\mathbf{S}})$
 8:         $\mathbf{T} \leftarrow \textsc{iterateTabuList}(\mathbf{T})$
 9:         $it \leftarrow it + 1$
10:     end for
11:     return $\mathbf{S}$
12: end procedure
```

Algorithm 2 Move

```
Input: $\bar{\mathbf{S}}, \mathbf{T}$
Output: $\bar{\mathbf{S}}$
 1: procedure $\textsc{move}(\bar{\mathbf{S}}, \mathbf{T})$
 2:     for $index < |\bar{\mathbf{S}}|$ do
 3:         if $random(0, 1) \geq 0.5$ then
 4:             $\mathbf{x} \leftarrow \bar{\mathbf{S}}(index)$
 5:             $\mathbf{x}' \leftarrow$ non–dominated solution of neighborhood($\mathbf{x}$)
 6:             $(k, j, i) \leftarrow$ swap done to achieve $\mathbf{x}'$
 7:             $\mathbf{T} \leftarrow \mathbf{T} + \{(k, j, i, l)\}$
 8:             $\bar{\mathbf{S}}(index) \leftarrow \mathbf{x}'$
 9:         end if
10:         $index \leftarrow index + 1$
11:     end for
12:     return $\bar{\mathbf{S}}$
13: end procedure
```

Algorithm 3 Check Dominance

```
Input: $\mathbf{S}, \bar{\mathbf{S}}$
Output: $\mathbf{S}$
procedure checkDominance($\mathbf{S}, \bar{\mathbf{S}}$)
    for $index < |\bar{\mathbf{S}}|$ do
        $\mathbf{x} \leftarrow \bar{\mathbf{S}}(index)$
        if $|\mathbf{S}| = 0$ then
            $\mathbf{S} \leftarrow \{\mathbf{x}\}$
        else
            $dominates$, $alternates$, $isDominated$, $enters \leftarrow$ False
            $\mathbf{D} \leftarrow \emptyset$
            for $index_2 < |\mathbf{S}|$ do
                $\mathbf{x\prime} \leftarrow \mathbf{S}(index_2)$
                $dominates \leftarrow \mathbf{x}$ dominates $\mathbf{x\prime}$          ▷ see Eq. 20.
                $isDominated \leftarrow \mathbf{x\prime}$ dominates $\mathbf{x}$        ▷ see Eq. 20.
                $alternates \leftarrow \mathbf{x}$ alternates $\mathbf{x\prime}$        ▷ see Eq. 21.
                $enters \leftarrow$ True
                if $dominates =$ True then
                    $\mathbf{D} \leftarrow \mathbf{D} + \{\mathbf{x\prime}\}$
                end if
            end for
            if $dominates =$ True then
                $\mathbf{S} \leftarrow \mathbf{S} - \mathbf{D}$
            end if
            if $enters =$ True $\wedge$ $isDominated =$ False $\wedge$ $alternates =$ False then
                $\mathbf{S} \leftarrow \mathbf{S} + \{\mathbf{x}\}$
            end if
        end if
        $index \leftarrow index + 1$
    end for
    return $\mathbf{S}$
end procedure
```

4.7 Quality of MOTS

To measure the quality of our MOTS we compute the *Spacing* and the *Spread* of the online efficient set, as [13] did to measure the performance of a multi-Objective Particle Swarm Optimization procedure. These are suitable metrics to measure multi-objective procedures performance. The Spacing of an efficient set measures how distant are the efficient solutions in an efficient set; as this metric takes into account the standard deviation of the distances between the efficient solutions, a low value of spacing tells that the solutions are uniformly spaced, thus, a low variable is desirable. The Spread measures how close the efficient solutions are to the optimal values of the single-objective problems.

Let q be the number of solutions in the efficient set. Let e_i be the distance to the nearest neighbour within the set of solution i. Let $\bar{e}$ be the average distance of e_i. Finally, let e^*_δ be the distance of the optimal solution of objective $\delta \in \{\alpha, \beta, \gamma\}$ to its nearest neighbour within the efficient set. Eq. (22) and (23) present the Spacing and Spread equations, respectively.

$$Spacing(\mathbf{S}) = \sqrt{\frac{1}{q}\sum_{e=1}^{q}(\bar{e} - e_i)^2} \tag{22}$$

$$Spread(\mathbf{S}) = \frac{\sum_{\delta\in\{\alpha,\beta,\gamma\}} e^*_\delta + \sum_{e=1}^{q} |\bar{e} - e_i|}{\sum_{\delta\in\{\alpha,\beta,\gamma\}} e^*_\delta + q \cdot \bar{e}} \tag{23}$$

5 Computational experiments

To test the performance of our MOTS we used four instances instances based on the literature: a small instance and three large instances.

5.1 Small instance

We used the same instance presented in the first case study in [9]. Tables 1 - 4 present the parameters used in the instance.

Table 1: Mobile devices information.

Mobile devices	**Voice**	**Video**	**Web**	f_k **(\$ units)**	p_k **(% charge)**
K1	1	1	0	62	85
K2	1	0	1	44	34
K3	1	1	1	98	74
K4	1	0	1	73	28
K5	1	1	1	82	50

Table 2: Networks information.

Network technology	w_j **(Mbps)**	c_j **(\$ units)**
LTE	70	80
wifi g	54	0
HSPA+	15	40

Table 3: Perceived signal of networks by mobile devices.

Mobile devices	**LTE**	**wifi g**	**HSPA+**
K1	87	78	64
K2	55	68	28
K3	92	47	88
K4	0	85	25
K5	91	95	93

Table 4: Services' demanded bandwidth.

Service	**Voice**	**Video**	**Web**
d_i **(Mbps)**	0.1	3	0.5

We ran our experiments using a Macbook Pro with 8GB RAM with intel i7 processor running at 3.2GHz with turbo boost. We used Gurobi Optimizer on Java to obtain the results of the single objective problem described in Section 3. Table 5 summarizes the results obtained. Column one show the objective optimized, column 2 the id of the solution obtained, columns 3,4, and 5 present the value of the decision variables.

Table 5: MIP results.

Objective	**Solution**	α	β	γ
minimize α	1	0.086	160	6
minimize β	2	0.213	0	9
minimize γ	3	0.207	160	4
minimize γ	4	0.447	80	4

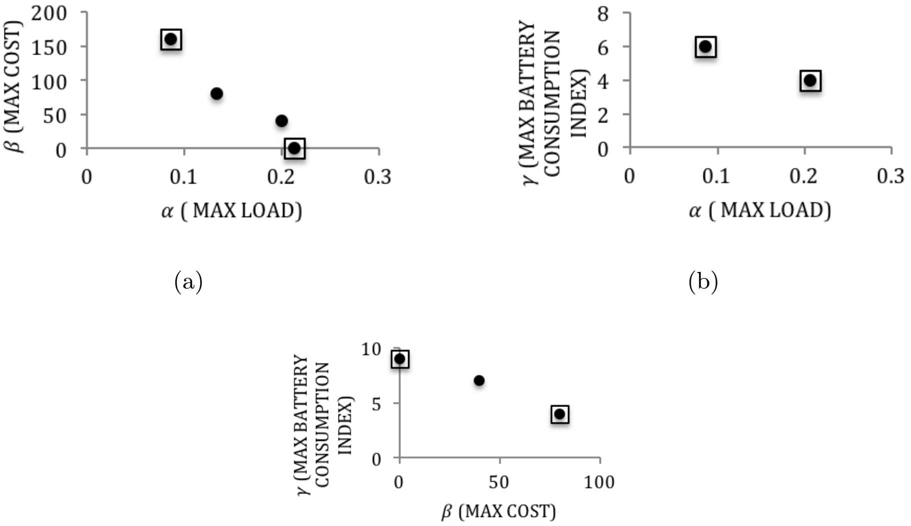
When minimizing the load (i.e., solution 1), the value of α is minimum regarding the ones obtained with the other solutions. As well, when minimizing the cost of connections, the value of β reduces to 0, this is because all services are provided by the wifi g network, which has no cost. Finally, minimizing γ we found two alternated solutions; solutions 3 and 4 have the same value for γ, yet the value of the other variables are different.

To test our MOTS, we generated objective function pairs in order to graph the solutions of the meta-heuristic. After fine-tuning of the parameters, we set $v = 10$, $n = 2,000$, and $l = 1,000$. Fig. 3 presents the efficient set for the three pairs of objective functions (i.e., α vs. β, α vs. γ, and β vs. γ), and the contrasted solution of the single objective problem. Black dots represent the efficient set; white squares represent the points associated to the optimal solutions of the single-objective problem. Note that solutions 1 and 2 (minimizing load and cost, respectively) are the extreme points of the efficient set in Fig. 3 (a); Solutions 1 and 3 are the efficient set of Fig. 3 (b); finally, Solutions 3 and 4 are the extreme points of the efficient set in Fig. 3 (c). Thus, our mots finds optimal efficient set for small instances of the problem.

Our meta-heuristic is able to calculate not only efficient supported solutions, but also the non-supported ones. Fig. 4 illustrates how the efficient set in Fig. 3(a) distributes over supported and non-supported solutions. Black-dotted solutions can be found through the convex linear combination of objective functions α and β; white-dotted solutions are non-supported ones. Traditional optimization of the MILP presented in Section 3 would have never find non-supported efficient solutions. This proves our MOTS is a good method to solve this type of problems.

Finally, we ran our MOTS taking into account all three objectives. As a result, we obtained the efficient set presented in Table 6 in a computational time of 0.96 seconds. Note that all optimal solutions of table 5 are present in the efficient set found.

For the optimal efficient set shown in Table 6, we obtained a Spacing equal to 13.54 and a Spread equal to 1.157. [9] reported a Spacing of 21.505 for this instance.



(a)

(b)

(c)

Figure 3: Efficient set when balancing pair of objectives: (a) α vs. β ;(b) α vs. β ; (c) β vs. γ

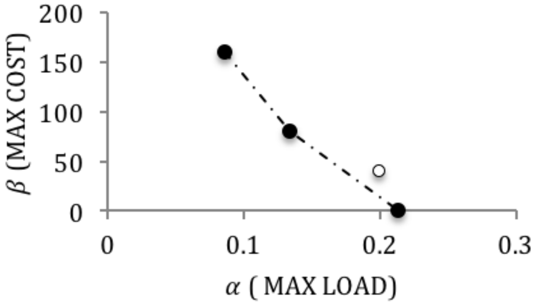


Figure 4: Supported and non-supported solutions for $\alpha - \beta$.

Table 6: Multi-objective efficient set.

Solution	α	β	γ
5	0.207	40	7
6	0.200	40	9
7	0.207	80	5
8	0.447	80	4
9	0.213	0	9
10	0.207	160	4
11	0.086	160	6
12	0.133	80	7

5.2 Large instances

To test the scalability of our MOTS we reproduced the experiments in [9]. We generated three random instances changing the amount of mobile devices: Instance 2 with 20 mobile devices, Instance 3 with 500 devices, and Instance 4 with 1,000 mobile devices. We maintained, at most, five services per device that can connect to seven access networks, having the information presented in [9]. Table 7 and Table 8 present the information of the networks and services used, respectively.

Table 7: Large-scale instances' networks information.

Network technology	w_j **(Mbps)**	c_j **($ units)**
LTE	70	80
wifi n	300	0
wifi g	54	0
wiMAX	15	60
HSPA+	15	40
HSDPA	2	20
UMTS	0.3	10

Table 8: Large-scale instances' networks information.

Service	d_i **(Mbps)**
Voice	0.1
Video	3.0
Web	0.5
Game	2.0
Chat	0.2

We ran our MOTS over the three large instances 30 times. Since the number of possible solutions is higher over the instances, after fine-tuning of the parameters we set $v = 10$, $n = 5,000$, and $l = 2,500$. The results obtained are different from the ones in [9] because we assumed some of the parameters that [9] did not specified in the case studies. Table 9 presents our results. Column 1 references the instance. Columns 2, 3 and 4 reference the average minimum values of α, β, and γ, respectively. Column 5 references the average number of solutions in the efficient set. Column 6 shows the average time running the algorithm. Finally, columns 7 and 8 present the average Spacing and Spread of the efficient set, respectively. Table 10 presents the standard deviation of the values shown in Table 9.

Table 9: Large-scale instances' average results.

Instance	α	β	γ	**Solutions**	**Time (s)**	**Spacing**	**Spread**
Instance 1	0.086	000.000	4.000	8.000	0.960	13.539	1.156
Instance 2	0.427	156.000	10.067	26.867	7.308	8.357	0.958
Instance 3	2.186	548.667	27.333	18.300	14.557	16.590	1.013
Instance 4	4.447	1,134.667	50.400	21.433	20.618	19.499	1.001

As the mobile devices grow, the minimum values of the objective functions are greater too. Nevertheless, the average time to solve the problem is quite reasonable, taking into account the

Table 10: Large-scale instances' standard deviations results.

Instance	α	β	γ	**Solutions**	**Time (s)**	**Spacing**	**Spread**
Instance 1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Instance 2	0.0500	24.7295	0.2494	156.0224	0.3881	4.9527	0.2154
Instance 3	0.4250	46.7470	0.4618	126.3764	0.3095	6.8807	0.2512
Instance 4	0.5419	76.4933	0.5384	140.1392	0.8100	7.1564	0.1929

average number of solutions in the efficient set. Due to the units of the objective functions, the average Spacing of the solutions obtained are obviously large, yet reasonable. Finally, the Spread of the solutions obtained oscillates around one, which is good. Once we take into account the values presented in Table 10, it is possible to see that the procedure is quite stable regarding CPU time. Also, the standard deviation of the Spreads are really low, ensuring that our meta-heuristic is always near to the real optimal values of the single-objective solutions. Taking into account the units of the instances, the variances of the Spacing are reasonable, telling that the uniformity of the efficient sets found is quite stable.

We are only presenting the minimum values found for each objective, and the decision of using one solution against another one depends on the decision-taker. Nevertheless, considering the metrics measured, the solutions are well distributed along the search space. The values of the standard deviation of β are larger than the other ones, because of the monetary units of this statistic, making reasonable to oscillate around the 50 monetary units.

We compared the results of our average Spacing metric per instance against the Spacing obtained with the genetic algorithm in [9]. Table 11 shows the Spacing obtained in each of the instances by [9] with the genetic algorithm and the average Spacing obtained with our proposed MOTS. As they do not provide any spacing metric for the fourth instance (i.e., 1,000 mobile devices), we only compare Spacings of the second and third instances.

Table 11: Spacing results.

	[9] spacing	**MOTS average spacing**
Instance 1	21.505	13.54
Instance 2	35.542	8.357
Instance 3	30.168	16.590
Instance 4	–	19.499

As the Spacing metric takes into account the standard deviation between solutions in the efficient set, a low value is always better. It is possible to notice that, although optimality was only proved in the first instance, our meta-heuristic produces better online efficient sets than the ones obtained with the genetic algorithm. In all the comparable instances our average Spacing outperforms the ones obtained by [9].

6 Conclusions

We developed a MILP model that takes into account the multihoming scheme; additionally, it only takes into account the destination network during the VHO process. This is a versatile consideration, because in real life, operators know *a priori* the initial networks that the mobile devices are connected to. Our model balance service loads in the union of the coverage areas of the access networks, representing a major contribution to the literature.

We tested our MILP using commercial optimizer Gurobi in a small instance to obtain optimal single-objective solutions. Its result were compared to the efficient set obtained with our MOTS, and matched with the extreme points of the non-dominated frontier. We found more than one non-supported solution, proving that the meta-heuristic was quite successful compared to the convex linear combination of the objective functions; but furthermore, our MOTS reaches the real efficient set for small instances of the problem.

We additionally tested the MOTS over large instances, and achieved good solutions within a reasonable computational time. In addition, we obtained some metrics on the online efficient set proving the efficiency of our meta-heuristic, yet optimality could not be proved. We compared the metrics of the online efficient sets obtained with our MOTS against a multi-objective genetic algorithm in the literature under the same instances. Our results are always better than those we compared to.

With the intuition behind our MOTS we were able to calculate good solutions, yet optimality was not proved. Future work includes merging efficient exact algorithms over networks with our meta-heuristic as a procedure to find good initial solutions; therefore, it could be possible to achieve optimality while solving multi-objective large scale problems.

Author contributions

Jorge Huertas formulated the new MILP model. He also designed and programmed the multi-objective Tabu Search, as well as the computational experiments. Yezid Donoso was the investigation project director. He contributed guiding and reviewing the investigation. All authors contributed equal to the paper.

Conflict of interest

The authors express no conflict of interests.

Bibliography

[1] Capela, N.; Sargento, S. (2012). Optimizing network performance with multihoming and network coding, In *2012 IEEE Globecom Workshops*, IEEE, Anaheim, CA, USA, 2012.

[2] Chu, H.D.; Kim, H.; Seok, S.J. (2013). Flow based 3G/WLAN vertical handover scheme using MIH model, In *The International Conference on Information Networking 2013 (ICOIN) IEEE*, 658–663, 2013.

[3] Donoso, Y.; Fabregat, R. (2007). *Multi-Objective Optimization in Computer Networks Using Metaheuristics*, Auerbach Publications, Boston, MA, USA, 2007.

[4] Donoso, Y.; Lozano-Garzon, C.; Camelo, M.; Vila, P. (2014). A Fairness Load Balancing Algorithm in HWN Using a Multihoming Strategy, *International Journal of Computers Communications & Control*, 9(5), 1841-9844, 2014.

[5] Duque, D.; Lozano, L.; Medaglia, A.L. (2014). An exact method for the biobjective shortest path problem for large-scale road networks, *European Journal of Operational Research*, 242, 788–797, 2014.

[6] El Rhazi, A.; Pierre, S. (2009). A Tabu Search Algorithm for Cluster Building in Wireless Sensor Networks, *IEEE Transactions on Mobile Computing* 8(4), 433–444, 2009.

[7] Glover, F.; Laguna, M. (1997). Tabu Search, Springer, US Boston, MA, 1997.

[8] Jaffrès-Runser, K.; Gorce, J.M.; Comaniciu, C. (2008). A Multiobjective Tabu Framework for the Optimization and Evaluation of Wireless Systems, In *Tabu Search*, I-Tech Education and Publishing, 2008.

[9] Lozano-Garzon, C.; Molina, M.; Donoso, Y. (2016). A Multi-Objective Approach for a Multihoming Load Balancing Scheme in WHN,*International Journal of Computers Communications* & *Control*, 11(2), 1841-9836, 2016.

[10] Marquez-Barja, J.; Calafate, C.T.; Cano, J.C.; Manzoni, P. (2011). An overview of vertical handover techniques: Algorithms, protocols and tools, *Computer Communications* 34, 985–997, 2011.

[11] Patriarca, F.; Salsano, S.; Bonola, M.; Cerqua, P. (2012). UPMT per-application mobility management solution, In *Proceedings of the 10th ACM international symposium on Mobility Management and wireless access - MobiWac '12*, New York, New York, USA (2012); ACM Press, ISBN 9781450316231, 2012.

[12] Peer Azmat Shah; Yousaf, M.; Qayyum, A.; Hasbullah, H.B. (2012). Effectiveness of multihoming and parallel transmission during and after the vertical handover, In *2012 International Conference on Computer & Information Science (ICCIS)*, 625–629 IEEE, 2012.

[13] Salazar-Lechuga, M.; Rowe, J. (2005). Particle Swarm Optimization and Fitness Sharing to solve Multi-Objective Optimization Problems, In *IEEE Congress on Evolutionary Computation*, 2, 1204–1211, 2005.

[14] Semchedine, F.; Bouallouche-Medjkoune, L.; Bennacer, L.; Aber, N.; Aïssani, D. (2012). Routing Protocol Based on Tabu Search for Wireless Sensor Networks, *Wireless Personal Communications* 67(2), 105–112,2012.

[15] Yan, X.; Ahmet, Y.; Şekercioğlu, S.; Narayanan, S. (2010). A survey of vertical handover decision algorithms in Fourth Generation heterogeneous wireless networks, *Computer Networks* 54, 1848–1863, 2010.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(6), 972-987, December 2018.

Scheme for Statistical Analysis of Some Parametric Normalization Classes

A. Krylovas, N. Kosareva, E.K. Zavadskas

Aleksandras Krylovas
Department of Mathematical Modelling, Vilnius Gediminas Technical University,
Sauletekio av. 11, Vilnius, Lithuania
aleksandras.krylovas@vgtu.lt

Natalja Kosareva
Department of Mathematical Modelling, Vilnius Gediminas Technical University,
Sauletekio av. 11, Vilnius, Lithuania
natalja.kosareva@vgtu.lt

Edmundas Kazimieras Zavadskas*
Laboratory of Operational Research, Institute of Sustainable Construction,
Vilnius Gediminas Technical University,
Sauletekio av. 11, Vilnius, Lithuania
*Corresponding author: edmundas.zavadskas@vgtu.lt

Abstract: In this research 7 parametric classes of normalization functions depending on 1 or 2 parameters proposed for MCDM problem solution. Monte Carlo experiments carried out to perform comparative statistical analysis and find optimal parameter values for the case of Gaussian distribution of decision making matrix elements. Optimal parameter values were ascertained for each normalization method. Normalization formulas were compared with each other in the sense of their efficiency. Logarithmic and Max normalization formulas demonstrated highest values of the best alternative identification. The proposed methodology of determining optimal parameter values of normalization formulas could be applied by approximation of real data with appropriate probability distributions.

Keywords: normalization methods, multi-criteria optimization, Monte Carlo method, comparative statistical analysis, SAW.

1 Introduction

Multiple criteria decision making (MCDM) methods deal with ranking of alternatives (projects) by measurements or evaluations of the projects according to finite number of attributes (criteria). Ranking results depend on many components of this process, the main of them that influence finite results are

- data normalization formula,
- weight determination method,
- data aggregation method.

Data normalization is an important part of a decision-making problem, but it is still not given enough attention in scientific literature. Nevertheless, it was shown in the number of scientific papers that data normalization method selection often significantly affects the accuracy of MCDM problem solution. The main topic of existing articles is investigation of various normalization formulas together with TOPSIS as one of the most popular MCDM method for ranking alternatives. Jahan and Edwards [4] proposed the comprehensive systematized review of existing normalization methods. Some of them are traditionally used with the certain MCDM methods, for example, the well known tandem of vector normalization (Van Delft and Nijkamp [17]) and The Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) method (Hwang and Yoon [3]). The study of Celen [1] revealed that TOPSIS with vector normalization generated the most consistent results among the most popular four normalization procedures. Among the linear normalization procedures, max-min and max methods appeared as the possible alternatives to the vector normalization procedure. A simulation comparison of normalization procedures for TOPSIS in terms of their ranking consistency and weight sensitivity was carried out by Chakraborty and Yeh [2]. The study results also justify the use of the vector normalization procedure for TOPSIS. Influence of normalization tools on COPRAS-G method applied for material selection task proposed by Yazdani *et al.* [19] In the study of Podviezko and Podvezko [13] it is shown that different types of transformation and normalization of data applied to popular MCDA methods, such as SAW or TOPSIS may produce considerable differences in evaluation. Authors stated that attention has to be paid to making a choice of the type of normalization, which reflects preferences of decision-maker.

Kosareva *et al.* [7] accomplished comparative statistical analysis of 5 widely used normalization methods with SAW method for ranking the alternatives and ternary estimates matrix. It is notable that results strongly differ for benefit and cost type attributes. Minmax method in most cases is significantly better than other. In the study of Peldschus [12] the impact of linear, concave and convex function profiles for mapping on a dimensionless interval (normalization) was investigated. Review of the normalization methods used in construction engineering and management, and their applications there are presented by Kaplinski and Tamošaitienė [6]. Milani *et al.* [11] examined how different families of norms affect the result of solving engineering decision problem by entropy and TOPSIS methods. It was verified that the linear optimization norms cannot affect the rank of alternatives significantly. In contrast, nonlinear norms may yield some deviations, mainly for alternatives that are inherently close.

Recently, some new normalization formulas were proposed in the literature. Research of Zavadskas and Turskis [20] is focused on introducing a new logarithmic method for decision making matrix normalization. Based on Weitendorf [18] and Juttler [5] formulas, Stanujkic *et al.* [14] proposed a new normalization procedure. The idea is to use the distance from the preferred ratings, which respects the decision-maker's preferences. This procedure, adapted for negotiations, was integrated with Step-Wise Weight Assessment Ratio Analysis (SWARA) and Additive Ratio Assessment (ARAS) methods in the research of Stanujkic *et al.* [15] Data normalization, as well as the measurement scales, inconsistency issues, missing judgement estimation methods, etc. have been extensively studied in the pairwise comparison matrix (PCM). A comprehensive literature review on PCM provided in Kou *et al.* [8] A group decision-making (GDM) method for integrating heterogeneous information proposed by Li *et al.* [9]

The purpose of this study is to propose a new methodology of constructing parametric normalization methods and to carry out their statistical comparative analysis. 7 classes of parametric data normalization procedures are presented. Some of the proposed normalization methods with particular parameter values are well known and widespread, other methods aren't so popular. Nevertheless, all these methods were not being applied using the wide range of parameter values so far.

The article is organized as follows. In the Section 2 seven classes of parametric normalization functions are introduced, their properties and dependency on parameter values are discussed. Experiment design and detailed description of initial data matrices generation procedure is given in the Section 3. Monte Carlo experiment results and comparative statistical analysis of normalization methods depending on the parameter values presented in the Section 4. Conclusions and future research are discussed in the Section 5.

2 Parametric data normalization methods

In this article Simple Additive Weighting (SAW) method (see Ref. 19) with equal weights is applied for MCDM problem solution. Let us suppose that the initial data are the results of some measurements, expert evaluations, etc., and are written in $m \times n$-dimension matrix $X = (x_{ij})_{m\times n}$. The element x_{ij} of decision making matrix is evaluation of alternative $i(i = 1, 2, \ldots, m)$ by the criterion $j(j = 1, 2, \ldots, n)$. Decision making matrix after normalization procedure is noted as follows: $\tilde{X} = (\tilde{x}_{ij})_{m\times n}, 0 \le \tilde{x}_{ij} \le 1$. Let $w_j, j = 1, 2, \ldots, n$ be criteria weights satisfying conditions $\sum_{j=1}^{n} w_j, 0 \le w_j \le 1$. Then, SAW criteria aggregated value is calculated for each alternative:

$$Q_i = \sum_{j=1}^{n} w_j \tilde{x}_{ij}, i = 1, 2, \ldots, m.$$

If $Q_1 \ge Q_2 \ge \ldots \ge Q_m$, then the alternatives are ranked as follows:

$$altern_1 \succ altern_2 \succ \ldots \succ altern_m.$$

The essence of any normalization procedure is mapping of the real values $x_{ij} \in [m_j, M_j] \subset (-\infty, +\infty)$ having certain meaning in the interval $[0, 1]$, which in general case represents these numbers unevenly. For example, functions $x^\alpha : [0, 1] \to [0, 1]$ depending on the parameter α represent half of the maximum values to the 13% of the maximum normalized values, when $\alpha = 0.2$, to the 30%, when $\alpha = 0.5$, 75%, when $\alpha = 2$ and even 97%, when $\alpha = 5$ (see Figure 1).

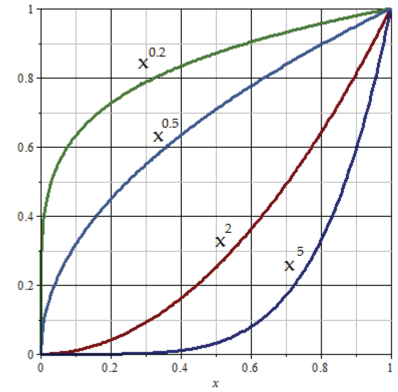


Figure 1: Normalization functions $x^\alpha : [0, 1] \to [0, 1]$.

It means that the aggregated values Q_i can not only depend strongly and not so much on the absolute values of x_{ij} as on the relationships $x_{i'j} < x_{i''j}$. In this research we'll limit ourselves with direct optimization, when higher values of criteria are better, omitting inverse optimization, when lower values are treated as better. So, the only essential requirement for normalization functions is – they must be monotonously increasing. The other restriction of our research is dealing with random generation of initial decision making matrix elements that have the Gaussian (normal) probability distribution.

Practically applied normalization methods can be more complex compared to functions depicted in Figure 1 and can therefore have a great deal of influence in alternative comparisons and decision-making results. Therefore, the literature deals with a large number of normalization methods, the new methods are developed, their researches and comparisons are carried out.

Denote $m_j = \min\limits_{1\le i\le m} x_{ij}, M_j = \max\limits_{1\le i\le m} x_{ij}, j = 1, 2, \ldots, n$. In the Table 1 we propose 7 classes of normalization functions $[m_j, M_j] \to [0, 1]$ depending on 2 parameters $\alpha \ge 0$ and $k \ge 0$.

These classes describe 7 different normalization approaches. Note, that (1), (6) and (7) methods depend exclusively on the parameter α, meanwhile the rest (2)–(5) methods are depending on two parameters $\alpha \geq 0$ and $k \geq 0$. When $\alpha = 1$, methods (1), (6) and (7) are known from the literature and are entitled as: (1) – Minmax normalization, (6) – Logaritmic normalization, (7) – Max normalization. Plots of functions (1)–(7), when $\alpha = 1, k = 1$, are given in Figure 2, plots of functions (1), (6) and (7), when $\alpha = 0.5, 2$ – in Figure 3). Graphs of functions (2)–(5) with corresponding parameter values $\alpha = 0.5, 2; k = 2, 3, 5$ are depicted in Figure 4. In the Table 2 the example of data matrix normalization by methods (1)–(7) is presented when $\alpha = 1$, $k = 1$ and initial data matrix is

$$(x_{ij})_{(4\times 4)} = \begin{pmatrix} 120 & 1.2 & 1 & 4 \\ 250 & 2.5 & 2 & 5 \\ 3600 & 36 & 5 & 8 \\ 6400 & 64 & 10 & 13 \end{pmatrix}.$$

The second column elements of matrix (x_{ij}) are obtained dividing the first column elements by 100, while the fourth column elements are calculated by adding constant 3 to the third column elements. Let us see that data normalization formulas (1)–(5) are invariant with respect to linear transformation $\alpha x + \beta$, formula (7) is invariant with respect to multiplication/division, but does not preserve addition, while formula (6) does not preserve nor multiplication, neither addition. All the first 5 methods map the lowest value to 0, and the highest value to 1, method (7) maps the highest value to 1.

Several questions naturally arise considering this issue. Are the mutual differences between the results of these methods application for MCDM problem solution significant? What are the "best" values of parameters α and k? How do the results vary when varying parameter α and k values? How do the results vary when varying m and n values?

Table 1: Formulas for 7 parametric normalization methods in the case of direct normalization.

Formula number	Normalization method	Direct normalization formula
(1)	Minmax normalization (Weitendorf, 1976)	$\tilde{x}_{ij} = \left(\dfrac{x_{ij} - m_j}{M_j - m_j}\right)^{\alpha}$
(2)	Exponential normalization	$\tilde{x}_{ij} = e^{-k\left(\dfrac{M_j - x_{ij}}{x_{ij} - m_j}\right)^{\alpha}}$
(3)	Logaritmic maxmin normalization	$\tilde{x}_{ij} = \dfrac{1}{\left(1 + k\ln\left(\dfrac{M_j - m_j}{x_{ij} - m_j}\right)\right)^{\alpha}}$
(4)	Arctangent normalization	$\tilde{x}_{ij} = \left(\dfrac{2}{\pi}\arctan\left(k\dfrac{x_{ij} - m_j}{M_j - x_{ij}}\right)\right)^{\alpha}$
(5)	Double exponential normalization	$\tilde{x}_{ij} = \left(\dfrac{1 - e^{-k\left(\dfrac{x_{ij} - m_j}{M_j - x_{ij}}\right)}}{1 + e^{-k\left(\dfrac{x_{ij} - m_j}{M_j - x_{ij}}\right)}}\right)^{\alpha}$
(6)	Logarithmic normalization (Zavadskas and Turskis, 2008)	$\tilde{x}_{ij} = \left(\dfrac{\ln(x_{ij})}{\ln(\prod_{i=1}^{m} x_{ij})}\right)^{\alpha}, x_{ij} \geq 1$
(7)	Max normalization (Stopp [16], 1975)	$\tilde{x}_{ij} = \left(\dfrac{x_{ij}}{M_j}\right)^{\alpha}$

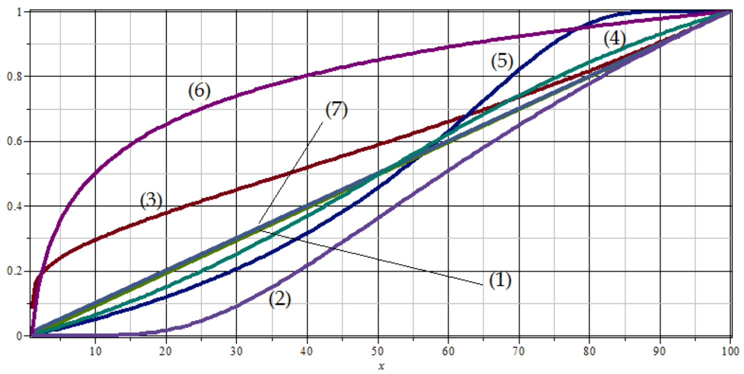


Figure 2: Functions (1)–(7), when $\alpha = 1, k = 1$.

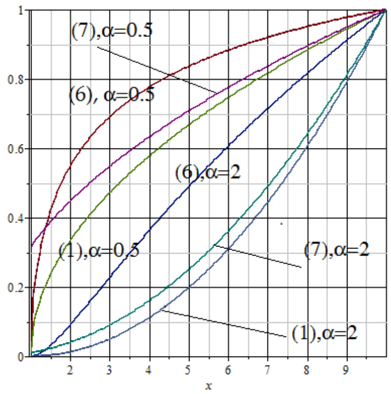


Figure 3: Functions (1), (6) and (7), when $\alpha = 0.5$ and $\alpha = 2$.

3 Experiment design

The case under investigation is when the decision making matrix elements are some measurements done with sufficient precision and having Gaussian probability distribution $x_{ij} \sim N(\mu_i, \sigma)$. We'll suppose, that the first row elements of the matrix have at the average higher values than elements of other rows $\mu_1 > \mu_2 = \ldots = \mu_m$. Only such matrices were analysed, for which the first alternative (first row) has no domination property in comparison with any other alternative, i.e. when the following conditions are not fulfilled:

$$(x_{11}, x_{12}, \ldots, x_{1n}) \succ (x_{i1}, x_{i2}, \ldots, x_{in}), \text{ if } x_{11} \geq x_{i1}, x_{12} \geq x_{i2}, \ldots x_{1n} \geq x_{in}. \tag{8}$$

If condition (8) is valid, weighted averages $Q_i = \sum_{j=1}^{n} w_j \tilde{x}_{ij}$ will satisfy inequalities $Q_1 \geq Q_i$ with any values of weights w_j. Therefore, the result of all normalization methods will be the same – the first alternative is better than i-th alternative. So, when random matrices are being generated, such matrices whose first and any (at least one) i-th row satisfy domination property (8) are being rejected. As a result, the number of cases of a fair decision has decreased. For example, the first and the second rows elements of decision making matrix

$$X^{(1)} = \begin{pmatrix} 52.34 & 66.31 & 63.38 & 67.01 \\ 48.90 & 62.05 & 56.54 & 53.93 \\ 72.05 & 56.24 & 61.58 & 48.05 \\ 52.10 & 65.00 & 71.95 & 56.82 \\ 77.69 & 70.34 & 65.00 & 55.65 \end{pmatrix}$$

Table 2: Normalization methods application example when $\alpha = 1$ and $k = 1$.

Normalization method	Normalized matrix $\tilde{x}_{ij}$	Formula
(1)	$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0.021 & 0.021 & 0.11 & 0.11 \\ 0.55 & 0.55 & 0.44 & 0.44 \\ 1 & 1 & 1 & 1 \end{pmatrix}$	$\tilde{x}_{ij} = \dfrac{x_{ij} - m_j}{M_j - m_j}$
(2)	$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 2.8 \cdot 10^{-21} & 2.8 \cdot 10^{-21} & 0.0003 & 0.0003 \\ 0.45 & 0.45 & 0.29 & 0.29 \\ 1 & 1 & 1 & 1 \end{pmatrix}$	$\tilde{x}_{ij} = e^{-\left(\dfrac{M_j - x_{ij}}{x_{ij} - m_j}\right)}$
(3)	$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0.21 & 0.21 & 0.31 & 0.31 \\ 0.63 & 0.63 & 0.55 & 0.55 \\ 1 & 1 & 1 & 1 \end{pmatrix}$	$\tilde{x}_{ij} = \dfrac{1}{1 + \ln\left(\dfrac{M_j - m_j}{x_{ij} - m_j}\right)}$
(4)	$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0.01 & 0.01 & 0.08 & 0.08 \\ 0.57 & 0.57 & 0.43 & 0.43 \\ 1 & 1 & 1 & 1 \end{pmatrix}$	$\tilde{x}_{ij} = \dfrac{2}{\pi} \arctan\left(\dfrac{x_{ij} - m_j}{M_j - x_{ij}}\right)$
(5)	$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0.01 & 0.01 & 0.06 & 0.06 \\ 0.55 & 0.55 & 0.38 & 0.38 \\ 1 & 1 & 1 & 1 \end{pmatrix}$	$\tilde{x}_{ij} = \dfrac{1 - e^{-\left(\dfrac{x_{ij} - m_j}{M_j - x_{ij}}\right)}}{1 + e^{-\left(\dfrac{x_{ij} - m_j}{M_j - x_{ij}}\right)}}$
(6)	$\begin{pmatrix} 0.18 & 0.02 & 0 & 0.18 \\ 0.20 & 0.10 & 0.15 & 0.21 \\ 0.30 & 0.41 & 0.35 & 0.27 \\ 0.32 & 0.47 & 0.5 & 0.34 \end{pmatrix}$	$\tilde{x}_{ij} = \dfrac{\ln(x_{ij})}{\ln(\prod_{i=1}^{m} x_{ij})}$
(7)	$\begin{pmatrix} 0.02 & 0.02 & 0.1 & 0.31 \\ 0.04 & 0.04 & 0.2 & 0.38 \\ 0.56 & 0.56 & 0.5 & 0.62 \\ 1 & 1 & 1 & 1 \end{pmatrix}$	$\tilde{x}_{ij} = \dfrac{x_{ij}}{M_j}$

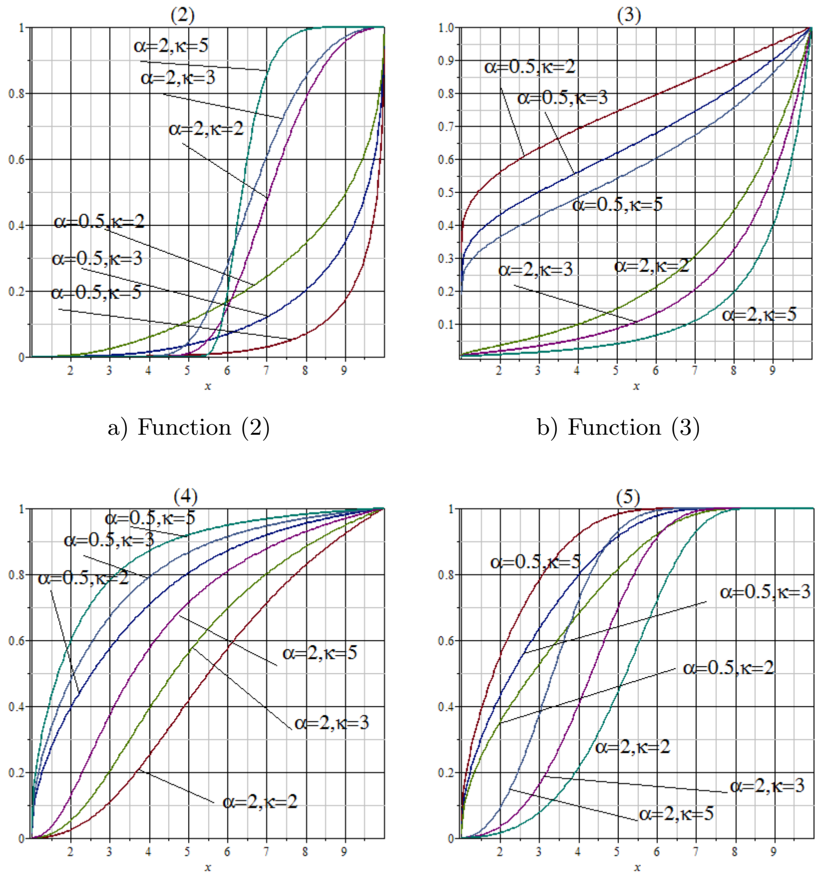
satisfy inequalities $52.34 \geq 48.90, 66.31 \geq 62.05, 63.38 \geq 56.54, 67.01 \geq 53.93$,

therefore $(52.34, 66.31, 63.38, 67.01) \succ (48.90, 62.05, 56.54, 53.93)$ and matrix $X^{(1)}$ is being rejected. Matrix

$$X^{(2)} = \begin{pmatrix} 57.56 & 66.42 & 58.31 & 55.51 \\ 58.51 & 63.08 & 67.93 & 46.31 \\ 55.36 & 57.58 & 59.16 & 72.82 \\ 38.95 & 58.40 & 40.93 & 57.93 \\ 53.68 & 45.57 & 70.41 & 59.98 \end{pmatrix}$$

is appropriate. The results of 7 normalization procedures when $\alpha = 1, k = 1$ are as follows:

$$\tilde{X}^{(2)}(1) = \begin{pmatrix} 0.95 & 1.0 & 0.59 & 0.35 \\ 1.0 & 0.84 & 0.92 & 0.0 \\ 0.84 & 0.58 & 0.62 & 1.0 \\ 0.0 & 0.62 & 0.0 & 0.44 \\ 0.75 & 0.0 & 1.0 & 0.52 \end{pmatrix}, \quad \tilde{X}^{(2)}(2) = \begin{pmatrix} 0.95 & 1.0 & 0.50 & 0.15 \\ 1.0 & 0.83 & 0.91 & 0.0 \\ 0.83 & 0.48 & 0.54 & 1.0 \\ 0.0 & 0.54 & 0.0 & 0.28 \\ 0.72 & 0.0 & 1.0 & 0.39 \end{pmatrix},$$



a) Function (2)

b) Function (3)

c) Function (4)

d) Function (5)

Figure 4: Graphs of functions (2)–(5) with different parameter α and k values.

$$\tilde{X}^{(2)}(3) = \begin{pmatrix} 0.95 & 1.0 & 0.65 & 0.49 \\ 1.0 & 0.85 & 0.91 & 0.0 \\ 0.85 & 0.65 & 0.68 & 1.0 \\ 0.0 & 0.67 & 0.0 & 0.55 \\ 0.78 & 0.0 & 1.0 & 0.60 \end{pmatrix}, \quad \tilde{X}^{(2)}(4) = \begin{pmatrix} 0.97 & 1.0 & 0.61 & 0.31 \\ 1.0 & 0.88 & 0.94 & 0.0 \\ 0.88 & 0.60 & 0.65 & 1.0 \\ 0.0 & 0.64 & 0.0 & 0.42 \\ 0.80 & 0.0 & 1.0 & 0.52 \end{pmatrix},$$

$$\tilde{X}^{(2)}(5) = \begin{pmatrix} 1.0 & 1.0 & 0.62 & 0.26 \\ 1.0 & 0.99 & 1.0 & 0.0 \\ 0.99 & 0.59 & 0.67 & 1.0 \\ 0.0 & 0.66 & 0.0 & 0.37 \\ 0.91 & 0.0 & 1.0 & 0.49 \end{pmatrix}, \quad \tilde{X}^{(2)}(6) = \begin{pmatrix} 0.21 & 0.21 & 0.20 & 0.20 \\ 0.21 & 0.20 & 0.21 & 0.19 \\ 0.20 & 0.20 & 0.20 & 0.21 \\ 0.19 & 0.20 & 0.18 & 0.20 \\ 0.20 & 0.19 & 0.21 & 0.20 \end{pmatrix},$$

$$\tilde{X}^{(2)}(7) = \begin{pmatrix} 0.98 & 1.0 & 0.83 & 0.76 \\ 1.0 & 0.95 & 0.96 & 0.64 \\ 0.95 & 0.87 & 0.84 & 1.0 \\ 0.67 & 0.88 & 0.58 & 0.80 \\ 0.92 & 0.69 & 1.0 & 0.82 \end{pmatrix}.$$

We can see that matrices $\tilde{X}^{(2)}(1) - \tilde{X}^{(2)}(5)$ mutually differ slightly, meanwhile there are essential differences between the first 5 matrices and $\tilde{X}^{(2)}(6) - \tilde{X}^{(2)}(7)$. When proper matrix X is generated and some kind of normalization (1)–(7) is done, SAW criteria aggregated values with equal weights are calculated for each row:

$$Q_i = \sum_{j=1}^{n} \frac{1}{n} \cdot \tilde{x}_{ij},\ i = 1, 2, \ldots, m. \tag{9}$$

The first alternative is considered to be the best one when $Q_1 > Q_2,\ \ Q_1 > Q_3, \ldots,\ Q_1 > Q_m$. 10 series by 100 experiments were conducted. After 100 realizations of experiments denote ID - the number of cases, when the first alternative identified as the best one (the number of correct MCDM problem solutions from 100). Denote also $W_{ID} \ \ = \frac{ID}{100}$, i.e. the percent of the correct MCDM problem solutions.

Table 3: Numbers of cases with the best first alternative (ID) for normalization methods (1)–(7), when $\alpha = 1$ and $k = 1$.

(1)	62	54	62	56	58	52	49	58	52	45
(2)	69	60	66	57	56	56	50	57	54	46
(3)	55	50	59	55	51	45	45	57	55	44
(4)	64	56	60	58	56	51	50	57	52	46
(5)	62	56	59	59	53	49	47	53	51	47
(6)	69	66	66	73	63	64	58	64	63	56
(7)	69	65	66	72	63	62	61	65	63	54

Table 3 shows experiment results – how many times after application of methods (1)–(7) with $\alpha = 1$ and $k = 1$ it was found that the best option is the first alternative. Each row contains results of 10 experiment series for each normalization method.

Next, Monte Carlo experiments were conducted by changing parameters α and k values as follows. First row elements of matrix X were generated according to the Gaussian distribution with the average value $\mu_1 = 67$ and standard deviation $\sigma = 15$, elements of other rows – with the average values $\mu_i = 57,\ i = 2, 3, \ldots, m$ and standard deviation $\sigma = 15$. Calculations were performed using the C++ program.

4 Experiment results

4.1 Dependence of the best alternative detection accuracy on parameter α

At the first stage of the experiments we fixed k value at $k = 1$ and varied α choosing the individual range for each normalization method. 100 series of 100 experiments were repeated for each parameter α value and empirical averages $\overline{W}_{ID} = \frac{1}{100}\sum_{i=1}^{100} W_{ID_i}$ calculated. The purpose is to detect α value which maximize*s* $\overline{W}_{ID}$.

In the Table 4 the dynamics of $\overline{W}_{ID}$ values depending on α is depicted. 95% confidence intervals $[w_{1\_0.95}; w_{2\_0.95}]$ for mean values EW_{ID} are given in the last column of the table. Maximum values of correct MCDM problem solution empirical averages $\overline{W}_{ID}$ were achieved for such α values: $\alpha = 2$ for methods (1), (3), (4) and (7), $\alpha = 0.75$ for method (2), $\alpha = 4$ for method (5), $\alpha = 20$ for method (6). The dependence of empirical mean values on α as well as upper and lower bounds of confidence intervals of the mean values are presented graphically in Figs. 5–6. For improvement of graphic images, smoothing spline fitting of Table 4 data was produced with MATLAB.

The question arises: are the detected differences between normalization methods with different α values statistically significant? Student's t-tests were applied for testing the hypotheses H_0 of equal average values EW_{ID} in the cases of $\alpha = 1$ and corresponding optimal parameter α values. Table 5 shows p-values of the respective t-tests. Significant differences in the best alternative

Table 4: $\overline{W}_{ID}$ values and confidence intervals for mean values EW_{ID} for 7 normalization methods at $k = 1$.

Normalization method	α	$\overline{W}_{ID}$	$[w_{1\_0.95}; w_{2\_0.95}]$
(1)	1	0.4784	[0.4678; 0.4890]
	1.25	0.4899	[0.4790; 0.5008]
	1.5	0.4998	[0.4885; 0.5111]
	2.	**0.5015**	[0.4911; 0.5119]
	3	0.4958	[0.4860; 05055]
	4	0,4905	[0.4804; 0.5006]
	5	0.4859	[0.4758; 0.4960]
(2)	0.5	0.4952	[0.4854; 0.5050]
	0.75	**0.4987**	[0.4876; 0.5098]
	1	0.4936	[0.4836; 0.5036]
	1.25	0.4897	[0.4796; 0.4998]
	1.5	0.4874	[0.4774; 0.4974]
	2	0.4807	[0.4707; 0.4907]
(3)	0.5	0.3438	[0.3342; 0.3534]
	1	0.4485	[0.4373; 0.3534]
	1.5	0.4859	[0.4758; 0.4597]
	2	**0.4979**	[0.4875; 0.4960]
	3	0.4959	[0.4862; 0.5083]
	4	0.4919	[0.4821; 0.5017]
(4)	1	0.4751	[0.4643; 0.4859]
	1.5	0.4947	[0.4842; 0.5052]
	2	**0.5009**	[0.4904; 0.5114]
	3	0.498	[0.4880; 0.5080]
	4	0.4929	[0.4827; 0.5031]
	5	0.4906	[0.4803; 0.5009]
(5)	1	0.4684	[0.4582; 0.4786]
	2	0.4914	[0.4812; 0.5016]
	3	0.4951	[0.4845; 0.5057]
	4	**0.4952**	[0.4844; 0.5060]
	5	0.494	[0.4834; 0.5046]
	6	0.4927	[0.4823; 0.5031]
	7	0.4923	[0.4822; 0.5024]
(6)	1	0.5370	[0.5258; 0.5482]
	2	0.5396	[0.5287; 0.5505]
	3	0.5444	[0.5336; 0.5552]
	4	0.5486	[0.5377; 0.5595]
	5	0.5508	[0.5401; 0.5616]
	10	0.5596	[0.5496; 0.5696]
	20	**0.5609**	[0.5512; 0.5706]
	30	0.5523	[0.5420; 0.5626]
(7)	0.25	0.5394	[0.5284; 0.5504]
	0.5	0.5400	[0.5290; 0.5510]
	0.75	0.5396	[0.5286; 0.5506]
	1	0.5403	[0.5293; 0.5513]
	2	**0.5419**	[0.5310; 0.5528]
	3	0.5384	[0.5276; 0.5492]
	4	0.5340	[0.5232; 0.5448]

detection accuracy on significance level 0.05 were observed for normalization methods (1), (3)–(6). Methods (2) and (7) didn't show significant differences. Consequently, it makes sense using formulas (1), (3)–(6) with the appropriate optimal parameter α values that differ from $\alpha = 1$ (see optimal α values in the Table 5), while it makes sense using $\alpha = 1$ for methods (2) and (7).

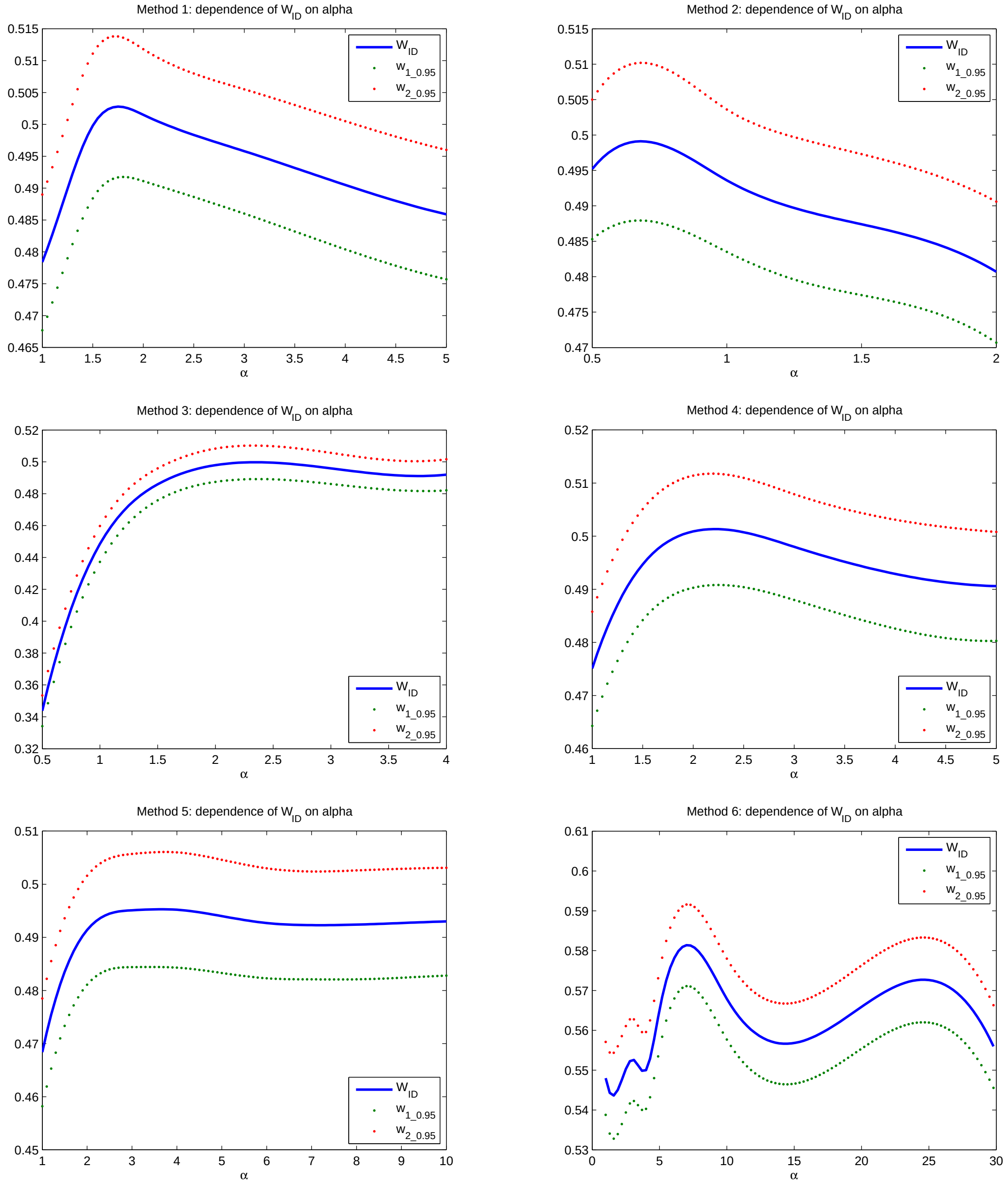


Figure 5: Dependence of empirical average accuracy $\overline{W}_{ID}$ on the parameter α values for (1)–(6) normalization methods and 95% confidence intervals bounds.

Next, ANOVA procedure was applied to check the hypothesis of equal averages EW_{ID} for 7 normalization methods with optimal α values. Procedure results show significant differences existing between methods. Bonferroni test was chosen for Post Hoc multiple comparisons. It revealed that there are nor significant differences between averages EW_{ID} for (1)–(5) methods, neither between (6) and (7) methods. However, when comparing any method from the first group

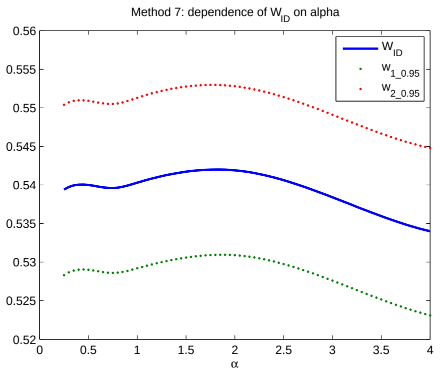


Figure 6: Dependence of empirical average accuracy $\overline{W}_{ID}$ on the parameter α values for (7) method and 95% confidence intervals bounds.

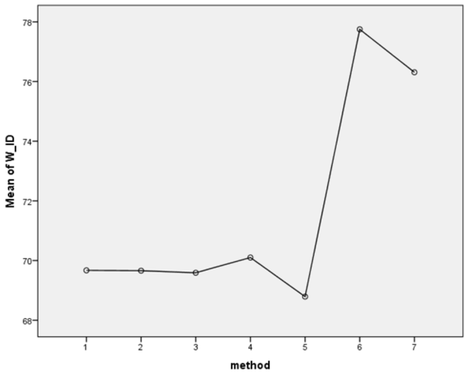


Figure 7: Mean plot from ANOVA procedure comparing EW_{ID} for optimal α values corresponding to 7 normalization methods.

((1)–(5) methods) with any of the methods of the second group ((6)–(7) methods), statistically significant differences were found. The conclusion can be made that (6) and (7) methods at the average are more precise than (1)–(5) methods, since the average accuracy EW_{ID} obtained by (6)–(7) methods is significantly higher than the accuracy of (1)–(5) normalization formulas. In Figure 7 the mean plot from ANOVA procedure output is represented. EW_{ID} for optimal α values corresponding to 7 normalization methods are depicted.

Table 5: Hypotheses of equal average values EW_{ID} testing results for $\alpha = 1$ and optimal α value for (1)–(7) normalization methods for the first experiment.

Normalization method	Optimal α value	H_0	p-value
(1)	2	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=2}$	0.002
(2)	0.75	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=0.75}$	0.499
(3)	2	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=2}$	0.000
(4)	2	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=2}$	0.001
(5)	4	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=4}$	0.000
(6)	20	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=20}$	0.002
(7)	2	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=2}$	0.838

Table 6: $\overline{W}_{ID}$ values, corresponding optimal α values and t-test results for (1)–(7) normalization methods for the second experiment.

Normalization method	Optimal α value	$\overline{W}_{ID}$	H_0	p-value
(1)	2	0.6967	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=2}$	0.000
(2)	0.75	0.6966	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=0.75}$	0.594
(3)	2	0.6959	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=2}$	0.000
(4)	2	0.701	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=2}$	0.000
(5)	3	0.6879	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=3}$	0.002
(6)	10	0.7775	$EW_{ID\ \alpha=1} = EW_{ID\ \alpha=20}$	0.003
(7)	1	0.7631	-	-

Table 7: Recommended normalization formulas (1)–(7) with optimal α values and $k = 1$.

$\tilde{x}_{ij} = \left(\frac{x_{ij}-m_j}{M_j-m_j}\right)^2$	(1)
$\tilde{x}_{ij} = e^{-\left(\frac{M_j-x_{ij}}{x_{ij}-m_j}\right)}$	(2)
$\tilde{x}_{ij} = \frac{1}{\left(1+\ \ln\left(\frac{M_j-m_j}{x_{ij}-m_j}\right)\ \right)^2}$	(3)
$\tilde{x}_{ij} = \left(\frac{2}{\pi}\ \arctan\left(\frac{x_{ij}-m_j}{M_j-x_{ij}}\right)\ \right)^2$	(4)
$\tilde{x}_{ij} = \left(\frac{1-e^{-\left(\frac{x_{ij}-m_j}{M_j-x_{ij}}\right)}}{1+e^{-\left(\frac{x_{ij}-m_j}{M_j-x_{ij}}\right)}}\right)^4$	(5)
$\tilde{x}_{ij} = \left(\frac{\ln(x_{ij})}{\ln(\prod_{i=1}^{m} x_{ij})}\right)^{20}$, $x_{ij} \geq 1$	(6)
$\tilde{x}_{ij} = \frac{x_{ij}}{M_j}$	(7)

Due to randomness of the experiments we observe some variability in the obtained results: optimal α value for (5)-th normalization method changed from 4 to 3, for (6)-th method – from 20 to 10, for (7)-th method – from 2 to 1. However, t-test didn't detect significant differences between EW_{ID} at the mentioned α levels. ANOVA results and conclusions also are the same. So, the conclusions remained unchained – it is recommended to use optimal α values, which differ from 1 for normalization methods (1), (3)–(6) and apply $\alpha = 1$ for methods (2) and (7). Moreover, (6) and (7) normalization formulas lead to significantly higher average accuracy of best alternative detection. The results of the experiments revealed that it is reasonable to use normalization formulas with α values specified in the Table 7.

To evaluate the stability of obtained results, Monte Carlo experiments were repeated with other values of decision making matrices. The elements of matrix X were generated as follows: $x_{1j} \sim N(67, 15)$, $x_{ij} \sim N(52, 15), i = 2, 3, \ldots, m$. As the differences $\mu_1 - \mu_i$ increased, it is natural to expect that the best alternative detection accuracy will be higher than in the previous experiment. The results of the second experiment are shown in the Table 6. They essentially confirmed results of the first experiment.

4.2 Dependence of the best alternative detection accuracy on parameter k

Data normalization formulas (2)–(5) are depending on both parameters α and k. So, it is interesting to reveal whether EW_{ID} change significantly while changing k. Calculations were carried out with few values of parameters α, and for each fixed α for few values of k. The elements of matrix X were generated as follows: $x_{1j} \sim N(67, 15)$, $x_{ij} \sim N(57, 15), i = 2, 3, \ldots, m$. 100 Monte Carlo experiments were repeated for each parameter values combination and empirical mean values $\overline{W}_{ID}$ calculated. Then maximum value of $\overline{W}_{ID}$ was detected for some optimal parameter k value. Calculation results for normalization formulas (4) and (5) are presented in the Table 8. Maximum $\overline{W}_{ID}$ values and corresponding parameter α and k values are as follows: normalization method (2) – $\overline{W}_{ID} = 0.511, \alpha = 0.75, k = 1$; (3) – $\overline{W}_{ID} = 0.505, \alpha = 2, k = 1$; (4) – $\overline{W}_{ID} = 0.516, \alpha = 2, k = 1$; (5) – $\overline{W}_{ID} = 0.509, \alpha = 1, k = 0.5$. In Figure 8 the dependence of average accuracy $\overline{W}_{ID}$ on $k = 1$ is depicted. For normalization formulas (2)–(4) optimal parameter k value is 1, only for method (5) optimal value differs from 1 ($k = 0.5$).

Next, t-test was applied for testing the hypotheses H_0 of equal average values EW_{ID} for $k = 1$ and corresponding optimal parameter k values. The difference between EW_{ID} with parameters $\alpha = 1, k = 0.5$ and EW_{ID} with parameters $\alpha = 1, k = 1$ is significant at significance level 0.05.

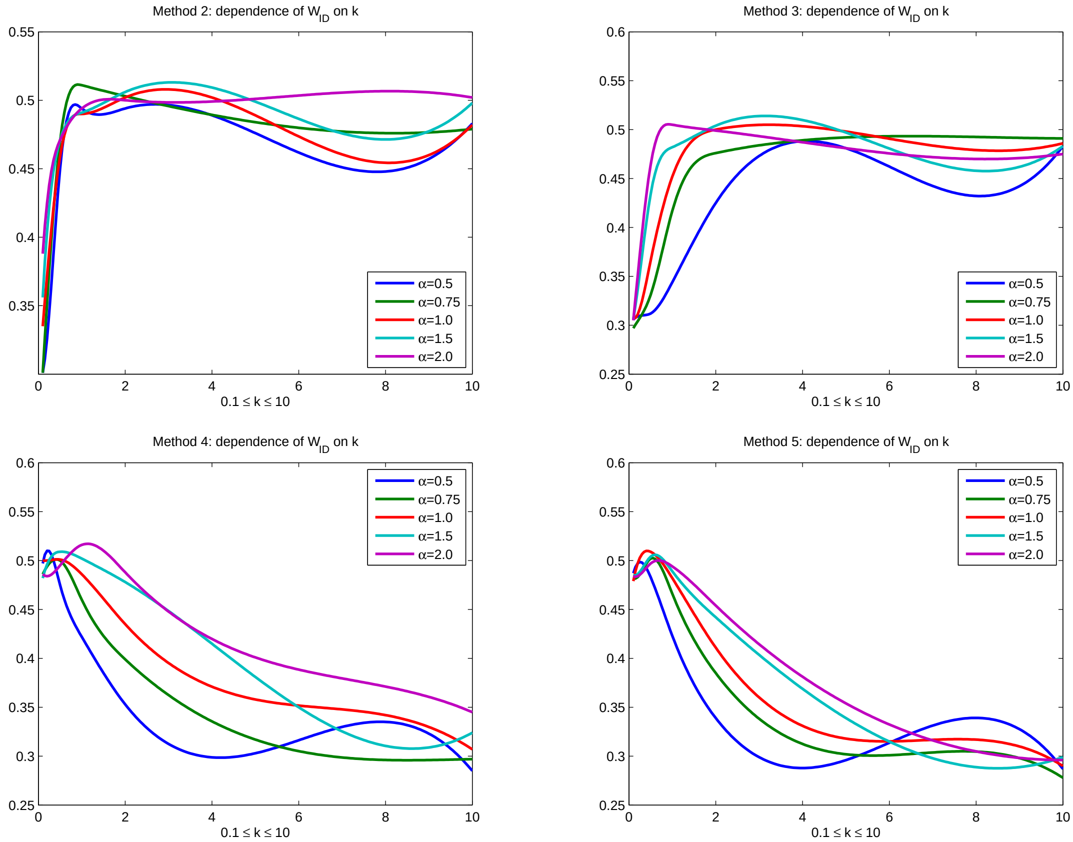


Figure 8: Dependence of empirical average accuracy $\overline{W}_{ID}$ on parameter k values for (2)–(5) normalization methods.

Table 8: $\overline{W}_{ID}$ calculated by varying α and k values for normalization methods (4) and (5).

			Method (4)			
α	0.5	0.75	1	1.5	**2**	3
k						
0.1	0.497	0.486	0.500	0.482	0.486	0.479
0.2	0.510	0.494	0.500	0.496	0.484	0.479
0.5	0.475	0.500	0.501	0.509	0.495	0.497
1	0.422	0.460	0.486	0.502	**0.516**	0.506
2	0.353	0.399	0.435	0.478	0.488	0.504
5	0.303	0.317	0.358	0.381	0.401	0.449
10	0.285	0.297	0.307	0.324	0.345	0.378
			Method (5)			
α	0.5	0.75	**1**	1.5	2	3
k						
0.1	0.487	0.482	0.479	0.485	0.484	0.485
0.2	0.497	0.483	0.479	0.486	0.484	0.483
0.5	0.483	0.502	**0.509**	0.504	0.497	0.498
1	0.423	0.466	0.482	0.489	0.494	0.494
2	0.339	0.385	0.411	0.442	0.454	0.473
5	0.296	0.302	0.318	0.339	0.354	0.381
10	0.287	0.278	0.290	0.299	0.296	0.323

So, it is recommended to use $\alpha = 1, k = 0.5$ for normalization method (5):

$$\tilde{x}_{ij} = \frac{1 - e^{-\frac{1}{2}\left(\frac{x_{ij}-m_j}{M_j - x_{ij}}\right)}}{1 + e^{-\frac{1}{2}\left(\frac{x_{ij}-m_j}{M_j - x_{ij}}\right)}},$$

whereas for methods (2)–(4) it is reasonable to use $k = 1$ with respective optimal α value (see relevant formulas in Table 7).

5 Conclusions and future research

The purpose of this article is to analyze some parametric normalization formulas and establish how various data normalization methods and parameter values affect the accuracy of MCDM problem solution. 7 data normalization methods were investigated (see formulas (1)–(7)). Some of them with certain parameter values are generalizations of the well known normalization methods. Data matrices were randomly generated according to Gaussian probability distribution. In all conducted Monte Carlo experiments decision making matrices were generated with the first alternative as the best one. Then the alternatives were ranked by the SAW method overall aggregated value with equal weights. The measure of identification accuracy is the percentage of correct identifications of the best alternative. The results of experiments are as follows.

1. Identification accuracy obtained by methods (6)–(7) is significantly higher than the accuracy for normalization methods (1)–(5).
2. Variation of parameter k revealed that for normalization method (5) it is reasonable to use combination of parameters $\alpha = 1, k = 0.5$, whereas for methods (2)–(4) it is reasonable to use $k = 1$ with respective optimal α value (see relevant formulas in Table 7).
3. Optimal α value have some "degrees of freedom", for example, it is possible to choose another optimal value $\alpha = 10$ instead of $\alpha = 20$ for normalization formula (6), since there isn't significant difference between identification accuracy at these values.
4. This research is accomplished in the special case when elements of decision matrix have Gaussian distribution. If we possess more information, we can apply the same methodology by approximating real data with appropriate probability distributions.
5. Parameters of initial decision making matrix were chosen so that identification of the best alternative would be higher than 50%. Experiments revealed that identification accuracy is higher with the bigger number of criteria n, and is lower with fewer alternatives m.

Bibliography

[1] Celen, A. (2014). Comparative Analysis of Normalization Procedures in TOPSIS Method: With an Application to Turkish Deposit Banking Market, *Informatica*, 25(2), 185–208, 2004.

[2] Chakraborty, S.; Yeh, C.-H. (2009). A Simulation Comparison of Normalization Procedures for TOPSIS, *International Conference on Computers and Industrial Engineering (CIE39): Troyes, France, JUL 06-09*, 1-3, 1815–1820, 2009.

[3] Hwang, C.L.; Yoon, K. (1981). *Multiple attribute decision making - methods and applications*, 2nd eds., Springer-Verlag, Berlin, 1981.

[4] Jahan, A.; Edwards, K.; Kevin, L. (2015). A state-of-the-art survey on the influence of normalization techniques in ranking: Improving the materials selection process in engineering design, *Materials & Design*, 65, 335–342, 2005.

[5] Juttler, H. (1966). *Untersuchungen zur Fragen der Operationsforschung und ihrer Anwendungs-moglichkeiten auf okonomische Problemstellungen unter besonderer Berucksichtigung der Spieltheorie* [Investigations on the question of operational research and its application to economic problems with special consideration of the game theory]: Doctoral dissertation, Fakultat der Humboldt-Universitat, Berlin, 1966.

[6] Kaplinski, O.; Tamošaitienė, J. (2015). Analysis of Normalization Methods Influencing Results: A Review to Honour Professor Friedel Peldschus on the Occasion of his 75th Birthday, *Procedia Engineering*, 122, 2–10, 2015.

[7] Kosareva, N.; Krylovas, A.; Zavadskas, E.-K. Statistical analysis of MCDM data normalization methods using Monte Carlo approach. The case of ternary estimates matrix, *Economic Computation and Economic Cybernetics Studies and Research. (In Press).*

[8] Kou, G.; Ergu, D.; Lin, C.; Chen, Y. (2016). Pairwise comparison matrix in multiple criteria decision making, *Technological and Economic Development of Economy*, 22(5), 738–765, 2016.

[9] Li, G.; Kou, G.; Peng, Y. (2018). A Group Decision Making Model for Integrating Heterogeneous Information, *IEEE Transactions on Systems, Man, and Cybernetics-System*, 48(6), 982–992, 2018.

[10] MacCrimmon, K.R. (1968). Decision making among multiple-attribute alternatives: a survey and consolidated approach, *No. RM-4823-ARPA, Santa Monica: RAND Corporation*, 1968.

[11] Milani, A.S.; Shanian, A.; Madoliat, R.; Nemes, J.A. (2005). The effect of normalization norms in multiple attribute decision making models: a case study in gear material selection, *Struct Multidiscip Optimization*, 29(4), 312–318, 2005.

[12] Peldschus, F. (2018). Recent Findings from Numerical Analysis in Multi-Criteria Decision Making, *Technological and Economic Development of Economy*, 24(4), 1456–1478, 2018.

[13] Podviezko, A.; Podvezko, V. (2015). Influence of data transformation on multicriteria evaluation result, *Procedia Engineering*, 51(1), 151–157, 2015.

[14] Stanujkic, D.; Magdalinovic, N.; Jovanovic, R. (2013). A multi-attribute decision making model based on distance from decision maker's preferences, *Informatica*, 24(1), 103–118, 2013.

[15] Stanujkic, D.; Zavadskas, E.-K.; Karabasevic, D; Turskis, Z.; Keršulienė, V. (2017). New group decision-making ARCAS approach based on the integration of the SWARA and the ARAS methods adapted for negotiations, *Journal of Business Economics and Management*, 18(4), 599–618, 2017.

[16] Stopp, F. (1975). Variantenvergleich durch Matrixspiele, *Wissenschaftliche Zeitschrift der Hochschule für Bauwesen Leipzig*, Heft 2, 1975.

[17] Van Delft, A.; Nijkamp, P. (ed.) (1977). Multi-criteria analysis and regional decision-making, *Studies in Applied Regional Science*, Springer, 1977.

[18] Weitendorf, D. (1976). *Beitrag zur optimierung der räumlichen struktur eines gebäudes: Dissertation A.*, Weimar: Hochschule für Architektur und Bauwesen, 1976.

[19] Yazdani, M.; Jahan A.; Zavadskas, E.-K. (2017). Analysis in Material Selection: Influence of Normalization Tools on COPRAS-G, *Economic Computation and Economic Cybernetics Studies and Research*, 51(1), 59–74, 2017.

[20] Zavadskas, E.-K.; Turskis, Z. (2008). A New Logarithmic Normalization Method in Games Theory, *Informatica*, 19(2), 303–314, 2008.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(6), 988-1006, December 2018.

Indoor Localisation through Probabilistic Ontologies

I. Mocanu, G. Scarlat, L. Rusu, I. Pandelica, B. Cramariuc

Irina Mocanu*
Computer Science Department, University Politehnica of Bucharest
Romania, RO-060042 Bucharest, Splaiul Independentei, 313
*Corresponding author: irina.mocanu@cs.pub.ro

Georgiana Scarlat
Computer Science Department, University Politehnica of Bucharest
Romania, RO-060042 Bucharest, Splaiul Independentei, 313
georgiana.scarlat@cti.pub.ro

Lucia Rusu
Faculty of Economics and Business, Babes Bolyai University of Cluj-Napoca
Romania, 400591, Cluj-Napoca, Teodor Mihali, 58-60
lucia.rusu@econ.ubbcluj.ro

Ionut Pandelica
Agora University of Oradea
Romania, 410526, Oradea, Piata Tineretului, 8
ionut.pandelica@univagora.ro

Bogdan Cramariuc
IT Center for Science and Technology
Romania, 11702, Bucharest, Av. Radu Beller, 25
bogdan.cramariuc@citst.ro

Abstract: For elderly people that are living alone in their homes there is a need to permanently monitor them. One of this aspect consist in knowing their indoor position and motion behavioural status, in real time. One possibility for indoor positioning of an user consists in understanding the images provided by supervising cameras. In this case the main aspect is represented by recognition of objects from these images. Thus, object recognition plays an essential part in understanding the environment and adding meaning to it. This paper presents a method for indoor localisation based on identifying the user's context. The user's context is computed based on object recognition and using a probabilistic ontology. The key element is represented by the probabilistic ontology that describes objects, scenes and relations between them. This ontology contains probabilistic relations that are learned using a large database. Results show that given a set of object detectors with high detection rate and low false positive rate, the system can recognize the user's context with high accuracy.

Keywords: object recognition, detection rate, probabilistic ontology, context identification.

1 Introduction

The high number of elderly who are living alone, or who are spending too much of the day without supervision of specialized people is increasing exponentially. With the development of Active and Assisted Living (AAL) technologies the localisation of persons is easier.

Indoor positioning/localisation systems (IPS) is similar with GPS and can be used successfully to locate people or objects inside buildings via mobile devices (smartphones or tablets). IPS relies on cameras mounted on walls or ceiling that work together to recognize user's context or objects. Results are in highly accurate position. Like GPS, IPS systems can detect the direction of movement, and it can predict the path based on this information so that accurate position remains as the space is displaced [20].

Hospitals and medical centers can benefit from this indoor localisation systems for staff, patients and managerial purposes. IPS staff includes: rapid location of colleagues in the building, finding records and mobile devices, notifications when and where patients are checked. IPS for patients that are seek can achieve the below benefits: automatic checking of the building's entrance, turn-by-turn dynamic way finding meetings/appointments with relevant information based on location, way finding back to the parked car.

For people with Alzheimer, eHealth and mHealth solutions includes complex IPS technologies, which are essential for the pursuit of both indoor and outdoor patients [1]. Therapy Acceptance and Commitment (ACT) encourages seniors (patients in general) in two directions: (1) acceptance of thoughts and emotions, difficult and undesirable, and (2) adoption and simultaneous promotion, of actions and behaviors in daily practice, which is a consistent of individual values. ACT includes mindfulness exercises that promotes contact with the present [1].

The goal of the system proposed in this paper is to offer a reliable context detection of a user in his home. The user's context will be represented by surrounding objects. Thus the main problem that must be solved consist in object recognition. Object recognition has evolved very much in the past decade but the current state of the art solutions are still far from what the human brain can do. Moreover, training object detectors involves a big amount of resources such as computational power and large image training set. Hence, object recognition is the main part of the proposed system - especially extracting the meaning from the object configurations found and detect the type of the scene in which they are in. For this scope, we propose a probabilistic ontology that describes objects, scenes and relations between them.

The system is described by a generic probabilistic ontology which can be instantiated according to the set of objects and scenes that need to be recognized. The ontology contains probabilistic relations that are learned using a large database containing thousands of images (LabelMe image database [15]) of annotated images with object and scene information. The results of analyzing the co-occurrence and spatial relations between objects are then used to improve the object recognition process.

For implementing a semantic information based solution for scene recognition, a large number of object detectors are needed because each scene can have so many different object configurations. Training so many object detectors is one problem, but it can be solved given the necessary resources. Another important problem is that using so many object detectors to scan through an image can be time consuming. The solution proposed in this paper tries to reduce the number of image scans to a minimum. This can be achieved if the algorithm would somehow "knows" what objects to search for. This would be possible in scenarios in which some objects are already detected and the next searched objects would be the ones that are relevant to the object context found so far. By doing this, the algorithm could recognize the scene after only a small number of object scans.

The rest of the paper is organized as follows: Section 2 presents some existing methods for object recognition. Section 3 contains the proposed method for improving scene recognition by using semantic information organized as a probabilistic ontology model. Section 4 presents the evaluation of the proposed system. Conclusions and future work are given in Section 5.

2 Related work

The IPS systems have made significant progress in the last years. The utility of those types of systems on GPS Tracking Devices is appreciated both for seniors, and also for health care professionals (HCPs) and Caregivers. Bellow are described some of the most popular systems and applications for HCPs and caregivers [11].

Balance is an application geared specifically for Alzheimer's Caregivers, which works on iPhone and iPad. The balance features are: Alzheimers disease references and information, Alzheimer's caregiving and advice, advanced medication management features (refill date, start date and dosage), native scheduling features, adds categories, relevant to caregivers, "Doctor diary" for logging symptoms and taking notes, news about Alzheimer's.

Mobicare is a simple, straightforward and free iPhone and Web application. Their features are: profile of loved ones who are receiving personal information, including birth date, gender, basic insurance information, the contact information for one physician, basic symptoms, tracking based on 15 preset choices (i.e. insomnia, wandering, etc.), basic medication, tracking but with some limitations.

Dementia Caregiver Solutions is an informational application for dementia caregivers. Its features are: perform advices for addressing the difficult behaviors associated with Alzheimer's and other types of dementia. They also have bookmark or "star" articles you wish to read in the future.

Object recognition is a vastly studied topic for which many systems were developed, however none of them are even close to the performance with which human brain can recognize objects, even though there are many variations in light, shape and color. The process through which the human brain accomplishes object recognition with very high speed and accuracy has been intensely studied by neurologists. Based on [2]: "The ability to rapidly recognize objects despite substantial appearance variation, is solved in the brain via a cascade of rflexive, largely feed forward computations that culminate in a powerful neuronal representation in the inferior temporal cortex. However, the algorithm that produces this solution remains poorly understood". In [3] suggests that "object recognition does not only involve physical properties (such as shape, color and texture) of the objects but also semantic information which includes the understanding of its use, previous experience with the object and how it relates to others".

Some approaches in object recognition were purely based on visual properties of the objects. For example, paper [5] tries to detect general classes of objects, not just specific ones by using part-based modelling and recognition of objects. The pictorial structure models were first introduced in [4]. They describe how a set of object parts arranged in a flexible configuration are used to model an object. The object parts represent their localized visual properties. The flexible object parts configuration is represented by pair-wise object connections. This approach is suitable for generic recognition problems because of the complex description of object visual features.

Another machine learning approach is presented in [18]. Their solution is capable of processing images extremely fast and achieves high detection rates. Integral images are used to speed up computations. Also an important aspect is the fact that they train extremely efficient classifiers using an algorithm based on AdaBoost, but that are used only a small number of visual features which are selected as being critical features. Many regions of the image than on background areas are discarded in the early stages of the algorithm. This is accomplished by combining increasingly more complex classifiers in a cascade. A major advantage of this solution is that it can run in real-time applications at 15 frames per second.

Both of the previously described systems are very promising but they lack semantic information about objects, which is crucial in recognizing objects with large variations in shape, such as furniture objects and others.

A synergy between Google and Toronto University research team obtain a remarkable result called MultiModel, which solve multiple translation tasks, image captioning with COCO dataset, a speech recognition corpus, and an English parsing task. This model can caption images, categorize them, then translate to French and German and construct parse trees, by spanning multimple domains. MultiModel used encoder-decoder architectures, and applied to neural machine translation Extended Neural. GPU is another model which used a recurrent stack of gated convolutional layers and ByteNet used left-padded convolutions in the decoder. Compared to Extended Neural GPU and ByteNet, MultiModel idea improves efficiency.and obtained good results in image classification [7].

The Inception deep convolutional architecture was called GoogLeNet or Inception-v1, than the was refined by the introduction of batch normalization in Inception-v2, or by additional factorization ideas in the third iteration - Inception-v3, later Inception-v4 with similarly expensive hybrid Inception-ResNet versions for both residual and non-residual Inception networks [17].

Another results was offered by CapsNet, which used a 3 layer architecture for convolutional neural networks (CNNs) for translated replicas of learned feature detectors. The primary capsules are the lowest level of multi-dimensional entities and corresponds to inverting the rendering process. The second layer (PrimaryCapsules) is a convolutional capsule layer with 32 channels of convolutional 8D capsules. The third Layer (DigitCaps) has one 16D capsule per digit class and each of these capsules receives input from all the capsules in the layer below.The implementation is in TensorFlow using the Adam optimizer [16].

There are, however, approaches that use semantic information concerning objects. One such approach is presented in paper [12, 13]. Object recognition and scene understanding is strongly influenced by semantic and context-based information from a psychological point of view. Therefore, they use the context information to improve object recognition based on visual properties. Their approach presents how to extract context probability maps from images. Also, based on these maps, they learn specific configurations for a set of object classes. The final goal is to filter out false positives.

There are new methods based on deep learning for object detection and recognition. In the context of object detection, several network architectures have been proposed [8], [9] and found to outperform methods based on traditional hand-crafted features. Most of these models were trained on RGB datasets such as PASCAL VOC to predict the bounding boxes of objects from images.

The main difference between the solutions presented in paper [12] and the solutions presented in this paper is that, here the context is used not only to remove objects that don't fit the context, but also for inferring what other objects can be found in the same context. For example, if a keyboard was detected at a previous step, then there is a high probability that a mouse may also be present in the same scene. This reasoning is used to make the system converge faster in order to recognize the scene by running the most relevant object detectors for the current context. For example, if a TV was detected, the system shouldn't run an object detector for finding a car, but should run an object detector for finding a coach. Also, this system is designed in a generic way, thus allowing the use of any objects and scenes. This means that entities modeled by the ontology are abstract and can be mapped to any set of objects and scenes as long as these have the needed correlations between them (objects are relevant for the chosen set of scenes).

3 System description

The system uses as stored data both a pre-trained probabilistic ontology model containing object and scene entities and relations among them and a set of object recognizers mapped on the object entities contained by the probabilistic ontology model, as given in Figure 1.

The object recognition module can run, at request, a specific object recognizer and provide the result as a pair of object confidence and object bounding box. ***The inference module*** represents the main reasoning algorithm and, at each iteration, sends a request to ***the object***

recognition module for finding a certain object. Based on the found/not found object results, the ***inference module*** uses the ***probabilistic ontology*** to filter out false positives, to determine what object to inquire about next and to determine if a scene can be recognized.

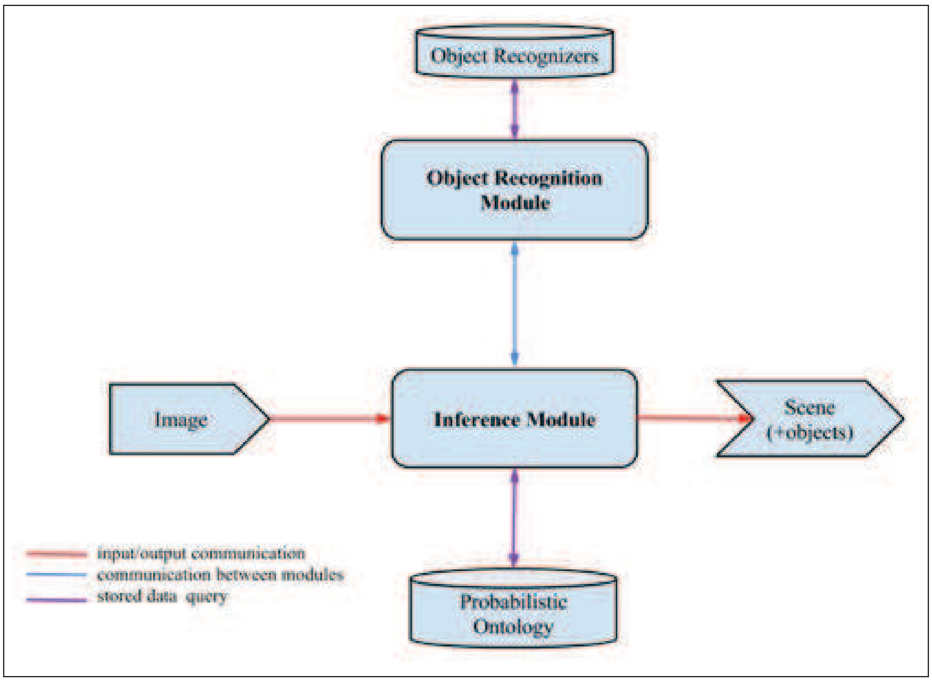


Figure 1: System architecture

In order to implement a reliable system for scene recognition, we consider the following steps:

- Selecting the ontology's structure: finding the set of relevant scenes and objects and mapping the meaningful relations among them into ontology relation types with associated attributes that reveal their semantic understanding.
- Choosing a large training data set that contains diverse context configurations for the chosen set of objects.
- Implementing a component that scans through the training dataset to compute object probabilities, scene probabilities and relation probabilities and aggregate all this information into a ***probabilistic ontology*** model.
- Implementing a object recognition method that allows testing the system and analyzing the influence on performance that object detection rate has on it.
- Doing ontology-based reasoning using already found objects to determine if a new found object belongs or not to the current context, thus filtering object recognition false positives.
- Doing ontology-based reasoning using previous object search results to determine what other objects can appear in the current scene in order to make the number of object recognition scans needed to recognize a scene much lower.
- Doing ontology-based reasoning to recognize a scene based on the previously found objects.

The system can run in two modes. The first one is the mode in which the system is used for running tests batches and computing system statistics. The tests are run on a big test database (approximately 3000 images) and the computed statistics consist of scene recognition accuracy, mean number of iterations needed to converge to recognise a scene and total number of removed false positives from the entire test set. This data is used to evaluate the system and analyze how different changes and improvements can affect its performance.

The second mode is used for viewing the analyzed image and obtaining the label associated to the image. An object is searched into the image - if the object is found then the most probable scene's name will appear. After that, correlated objects with already found are searched in the image. The rectangles for founded objects are displayed on the input image and the recognized scene's name is provided.

3.1 Object recognition

The scene recognition problem can also be approached using only image processing algorithms that analyze low level information about color, shapes and texture. This approach has the advantage that the classification model is easier to learn and design, but the disadvantage is that this method cannot distinguish very well between different indoor scenes such as living room, bedroom and so on. This is due to the fact that indoor scenes usually have similar colors, shapes and textures. Such scenes with similar visual features can only be distinguished using more top level information about the image such as objects and correlations between them.

Moreover, neurological studies [20] show that even the human brain cannot distinguish well objects and scenes unless there is some semantic meaning attached to them. As a result, scene recognition is best approached using semantic information regarding the scene.

Using object recognition for solving a scene recognition problem has the advantage that once the objects are found, recognizing the scene becomes a much simpler problem. However, there is a huge disadvantage to this approach. Object detectors can sometimes generate false positives. An object that is "alien" to the current object context can affect the final recognized scene very much, especially if there are only a few objects in the scene. The approach presented in this paper shows how contextual information can also be used to filter out false positives, thus increasing overall scene recognition accuracy.

Both ideas presented above about reducing the number of object scans and filtering out false positives are based on the relations that exist between objects. Some objects are strongly related among them (for example: keyboard and mouse, table and chair, etc.) and others are hardly related and can't be seen together in the same scene very often (for example: tree and TV, refrigerator and car, etc.). However, these relations are not always applicable (for example: keyboard can appear in a scene without a mouse). Therefore, they should be modeled stochastically. In this paper, the chosen model for representing the relations between objects is a probabilistic ontology.

The ontology used in this paper contains as entities objects and scenes and as relations the stochastic relations among objects and among scenes and objects. Every ontology entity has an apriori probability and every relation is also described by the probability of it to be true in a scene. The object to object relations from the ontology are described by an attribute called *Interpretation* that helps distinguish between positive and negative relations. Positive relations are those for which the objects are semantically connected to each other, meaning that this makes it more probable for them to appear in the same scene. The negative relations are exactly the opposite and they mean that the two objects are less likely to be found together.

For filtering out false positives obtained as a result of object detection, the current approach uses the previously found objects in the scene and the relations that connect them to the newly found object. If these relations are positive ones, this will increase the detection confidence of the checked object, but if the relations have a negative interpretation then this will decrease the detection confidence. If the newly found object's detection confidence has decreased considerably after this update, then it is considered to be a false positive and it will be eliminated from the found objects list.

If the first detected object is a false positives there are not afterward corrections applied. However, a prevention method is used in order to decrease the chances of this to happen. This measures is to use the false positive rate of each object detector as an influencing factor in choosing what object to search for in the case when no objects were found so far. This means that objects whose object detectors have lower false positive rates are preferred in the initial steps of the algorithm.

Choosing what objects to search for so that a scene can be recognised faster uses the positive and negative relations between previously searched objects and current objects that are candidates for search. This means that candidate objects that are in positive relations with previously found objects with high detection confidence are more suitable to be searched for next than candidate objects that are in negative relations with previously found objects.

Also, this reasoning is applied not only for the searched and found objects, but also for the searched and not found objects. If an object was not found after scanning the current scene, then candidate objects that are in a negative relation with it are more suitable for the next search than the ones that are in a positive relation with it. This reasoning is applied so that the next object choosing criteria can be more complex even if no objects were found yet.

The weight of the influence the not found objects have on choosing a new candidate is smaller than the weight for the found objects. This is justified by the fact that even though an object is not found in a scene, it does not necessarily mean that the current context is not suitable for it. For example, in a kitchen scene there might not be a stove object found, but this does not mean that other objects related to it cannot exist in that scene. However, if there are two equally suitable objects candidates according to their relations with already found objects in the scene, a tie breaker between them can be the same criteria based on not found objects after scanning the image. This reasoning is useful if the algorithm is in a state when all the previous searched objects are not found and it should try to search objects from different contexts than the ones searched before so that the chances of finding a new object increase.

All the criteria for choosing the next object to scan the image described earlier are combined into a fitness formula and the candidate object with the highest fitness value is chosen. The fitness value can also take into account the object's associated object detector false positive rate in case no objects were found yet in order to avoid the situation when the first found object is a false positive. In order to recognize a scene after some objects were found, the proposed solution uses the semantic information stored as relations between scenes and objects inside the ***probabilistic ontology***. The relation ***inScene*** is described by the probability of an object to be in a certain scene. At each step of the algorithm, the set of already found objects in the scene is used together with the relations between them and the entire set of scenes to compute the a fitness value for each scene.

The number of objects needed to recognize a scene can vary very much depending on the objects and how correlated they are with a certain scene. In some cases, only a few objects can be enough to know the scene as long as their detection confidence is big enough. For example, if a stove object was found and it has a very high detection confidence, then the most probable scene by far is kitchen.

Every time a new object is found, the fitness value is computed for each scene and if the scene with the highest fitness value is much beyond average then it is returned as the recognized scene and the object recognition process ends. The probabilities contained in the ontology are computed based on a large annotated image dataset, such that it can be applicable in general cases. The entities and the topology of the ontology can vary according to the use case it is designed for, but it is important that they reflect meaningful semantic information about objects and how they relate to one another and scenes and what objects are most probable to be contained by them.

Probabilistic ontology

The probabilistic ontology can be modeled using any topology structure and any entity set as long as these are relevant to the current use case. However, there are some restrictions on how to build the ***probabilistic ontology***. More precisely, the entity set must contain both scenes and objects. The object set has to be identical to the object set used by the ***object recognition module***, and the objects must be relevant for the scene set.

The general structure of the ***probabilistic ontology*** is given in Figure 2. The possible relations are: *inScene*, *is-a*, *Object-Object relation*, *hasVisualFeature*.

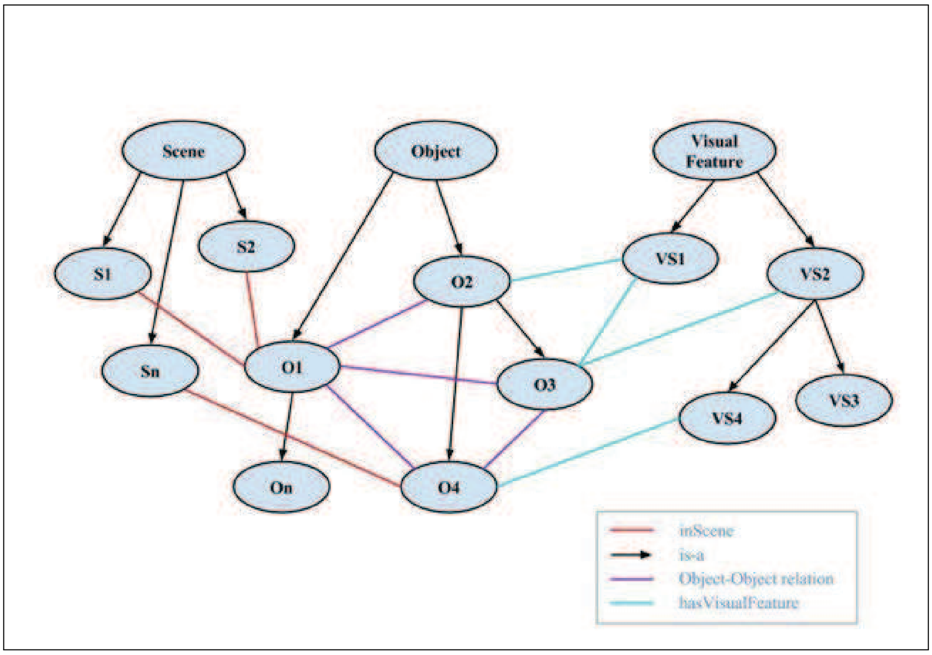


Figure 2: Ontology structure

There is no restriction regarding the stochastic relations between objects, except the fact that the relations have to be either positive or negative and weights have to be provided for each relation. The relation between objects and scenes is restricted to be of the type **inScene**, which reflects how probable it is for an object to be in a scene. Also, this relation should fully connect all the objects and scenes. Another restriction regarding the ontology building is that there should be no 0 value probabilities (in the case there are, they should be replaced with a very small value, close to 0).

An example of a part of the ***probabilistic ontology*** is given in Figure 3.

Each object to object relation contained by the ***probabilistic ontology*** has a fixed format. We consider the following relations properties:

- **Name**: A string that uniquely identifies each relation
- **Interpretation**: Represents how the relation should be interpreted during reasoning (POSITIVE, NEGATIVE).

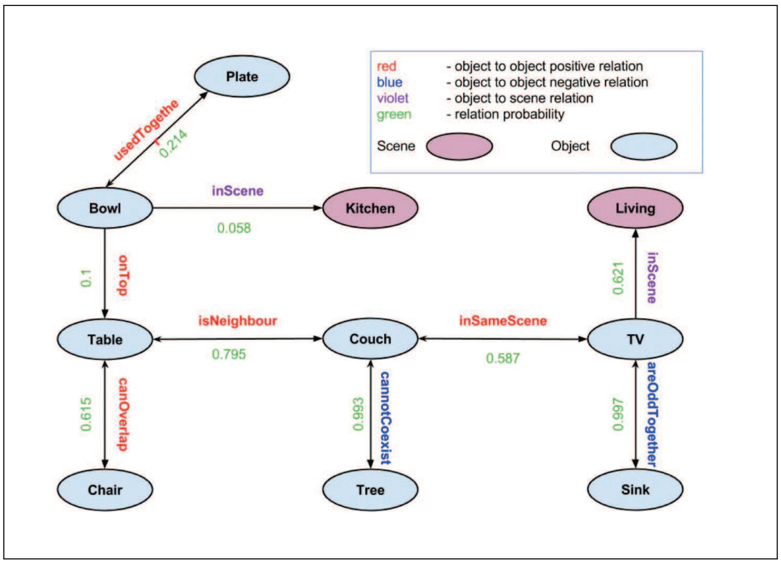


Figure 3: Ontology example

- **Check Weight**: Represents how important is this relation for checking newly detected objects and filtering out false positives.

- **Next Weight**: Represents how important this relation is for determining the next object to query the Object *Recognition Module*.

- **Check Rule**: Some relations are based on spatial requirements. This rule is used to check if the spatial requirements are met. For the relations that have no spatial meaning attached to them, this rule should be NONE.

This ontology contains as entities objects from a house and the following scenes: kitchen, living room, street, office.

We pointed relationship description between objects:

- **usedTogether**: Two objects are often used together. The object pairs that are in this relation are given by the user according to common sense information about the objects in the object set.

- **inSameScene**: Two objects often appear together in the same scene. The object pairs are determined by the object recognition algorithm according to the probability of the co-occurrence of each pair of objects.

- **cannotCoexist**: Two objects can't both be present in the same scene. The object pairs are determined by the object recognition algorithm according to the probability of the co-occurrence of each pair of objects.

- **areOddTogether** Two objects may appear together in the same scene, but their combination seems unnatural. The object pairs that are in this relation are given by the user according to common sense information about the objects in the object set.

- **onTop**: Most of the cases when the two objects appear together, one of them is placed on top of the other. The object pairs that are in this relation are given by the user according to common sense information about the objects in the object set.

- **canOverlap**: Most of the cases when the two objects appear together, their areas overlap. The object pairs that are in this relation are given by the user according to common sense information about the objects in the object set.
- **isNeighbour**: Most of the cases when two objects appear together, they are very close to one another. The object pairs that are in this relation are given by the user according to common sense information about the objects in the object set.

Properties of these relationship between objects are syntetize in Table 1. The rules are the following:

$$\textbf{OnTopRule} : Center(object1).y > Center(object2).y$$

$$\textbf{IsNeighbourRule} : Distance(object1, object2) \leq \frac{Diagonal(object1) + Diagonal(object2)}{2}$$

$$\begin{aligned}\textbf{CanOverlapRule} : Left(object1).x < Left(object2).x < Right(object1).x \\ AND \\ Left(object1).y < Left(object2).y < Right(object1).y\end{aligned}$$

Table 1: Properties of relationship between objects

Relation Name	Interpretation	CheckWeight	NextWeight	CheckRule
usedTogether	POSITIVE	0.8	0.95	NONE
inSameScene	POSITIVE	0.6	0.9	NONE
cannotCoexist	NEGATIVE	0.98	0.98	NONE
areOddTogether	NEGATIVE	0.75	0.85	NONE
onTop	POSITIVE	0.8	0.7	OnTopRule
canOverlap	POSITIVE	0.8	0.65	CanOverlapRule
isNeighbour	POSITIVE	0.8	0.7	IsNeighbourRule
inScene	BELONGING	0	0	NONE

The values for the relation *CheckWeight* and *NextWeight* attributes are chosen manually so that they reflect how reliable a relation is for checking for false positives (*CheckWeight*) and how reliable a relation is for predicting the presence of one of its referred objects when the other one is already found in the scene (*NextWeight*). For example, the relation **usedTogether** describes a much stronger bond between objects than the relation **inSameScene**, therefore its *CheckWeight* is much bigger (0.8 vs. 0.6). However, both usedTogether and inSameScene are relations between objects that are frequently found together, therefore their *NextWeight* is very big (0.95 and 0.9).

These weights are used so that not all relations influence in the same way false positive filtering and next object choosing, because each of them has a different semantic interpretation and should affect the reasoning process in its own custom way. The *CheckWeight* and *NextWeight* attributes have the role of quantifying semantic relation attributes as numbers in the [0, 1] interval, that can be used as components in the **inference module** reasoning formulas.

The current ***probabilistic ontology*** contains the relation between objects and scenes: **inScene** describes how probable is for an object to appear in a scene.

The probabilities of each object, each scene and each relation were computed based on the LabelMe database [15] that currently contains 78840 annotated images.

3.2 Object recognition

The **object recognition module** has access to a set of predefined object detectors which are run at the request of the **inference module** on the input image or on a region of the input image. The result that is supplied by this module is represented by a tuple of the form (*FoundObject*, *BoundingBox*, *Confidence*). The bounding box represents the area in the image where the found object is placed. This information is used by the ***inference module*** to check ontology rules that have spatial requirements. The confidence of object detection is also used by the ***inference engine*** as a measure of how much the detected object influences reasoning regarding other related objects.

An important aspect worth mentioning is that not all the object detectors are necessarily run in order to recognize a scene, The **inference module** decides to inquire about a certain object which it believes it is more probable to appear in the image based on previous detections, and when it has enough information to infer the scene with a high probability, the system returns and no more detectors are run. This means that, given a powerful **inference module** that obtains reliable information from all the other components, the final result can be obtained very fast, avoiding the costly image scans that the object detectors apply.

The **object recognition module** is responsible for running object detectors on request. Also, this component has access to information regarding each object recognizer that is relevant to the **inference module**. More precisely, the **object recognition module** can provide information about each object detector's detection rate and false positive rate. This information is meaningful for the **inference module** because it influences what object to inquire about.

3.3 Inference module

The **inference module** represents the system's component that implements the main algorithm for scene recognition. A scene can be inferred based on the objects recognized using the **object recognition module**. The information regarding found or not found objects is used to interrogate the **probabilistic ontology**. This ontology contains information about the stochastic relations between scenes and objects and between objects. The algorithm starts by choosing a first object to interrogate the **object recognition module** about. The first object is chosen based on false positive rate. Therefore, the first interrogated object is the one with the lowest false positive rate. This is done because at the initialization of the system, there is no information available about the input image, therefore it is important not to start with invalid information that can compromise future reasoning. After the initialization step when the first object is chosen, the algorithm enters a repetitive loop. At each iteration, the **object recognition module** is queried about the existence of a certain object inside the input image. If the object is found, then its confidence and bounding box are available. Next, a set of inference rules are used. The **first inference rule** has the role of filtering out false positives. The information used to update an object's detection confidence is represented by the positive and negative relations between the current object and the previously found objects. If the newly found object is in a positive relation with a previously found object then its detection confidence will increase. On the other hand, if the current object is in a negative relation with a previously found object, then its detection confidence will decrease. A visual description of the **first inference rule** is given in Figure 4.

The **first inference rule** are the following rules (R_i), i=1,7:

- *(R1)* Compute the list of previously found objects rejected by the current found object.
- *(R2)* Compute rejection factor of the found object REJ_{FOUND} according to each rejected object relation.

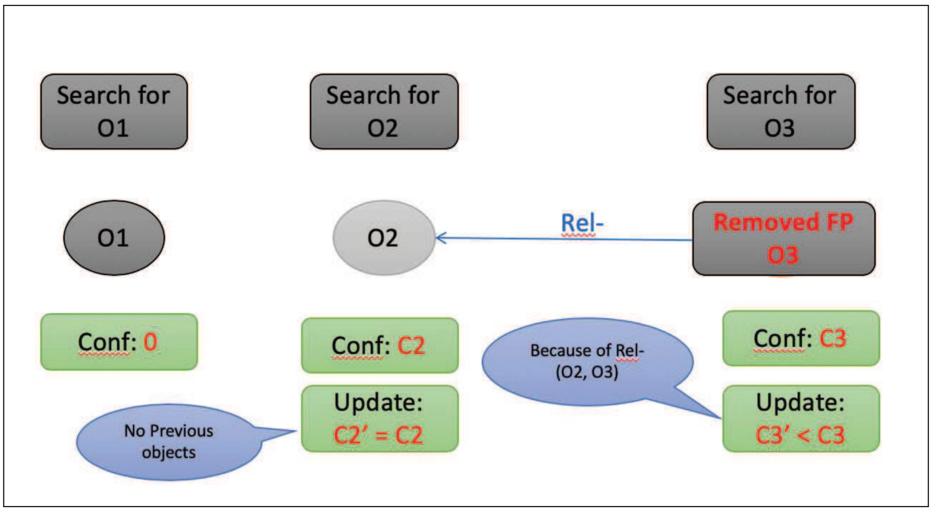


Figure 4: Description of the first inference rule

- *(R3)* Compute the list of previously found objects attracted by the current candidate object.
- *(R4)* Compute the found object attraction factor. $ATTR_{FOUND}$ according to each attracted object relation
- *(R5)* Update the current found object's detection $C(O)$ rate according to the rejection and attraction factors already computed: $ATTR_{FOUND}$, REJ_{FOUND}

$$C(O) = C(O) * (1 - \frac{REJ_{FOUND}}{|Objects_{REJECTED}|} + \frac{ATTR_{FOUND}}{|Objects_{Attracted}|})$$

- *(R6)* If the found object detection confidence $C(O)$ has decreased more than 20%, then it is filtered out.

The **second inference rule** has the role of determining what the next searched object should be. This rule is important because it helps the algorithm converge faster to the recognized scene and it avoids running all the object recognizers on the input image. In order to determine what object to search for next, information about the positive and negative relations from the **probabilistic ontology** is used against the set of previously found and not found object sets. Therefore, the next object to inquire the **object recognition module** about is the one that best fits the current context. More precisely, not searched objects that are in a positive relation with previously found objects will have a bigger chance at being next and not searched objects that are in a negative relation with previously found objects will have a smaller chance at being next. A visual description of the **second inference rule** is given in Figure 5.

Similar reasoning is used also in the case of previously not found objects (objects that were searched for in the input image but were not found), but in this case the positive relations have a negative impact on the new object's chance at being next and the negative relations have a positive impact on the new object's chance at being next. However, the reasoning based on previously found objects has a bigger weight in influencing the detection rate than the one on previously not found objects. Treating these two cases differently is justified by the fact that, even though an object is not found in the input image, this does not necessarily mean that the object cannot belong to the current context.

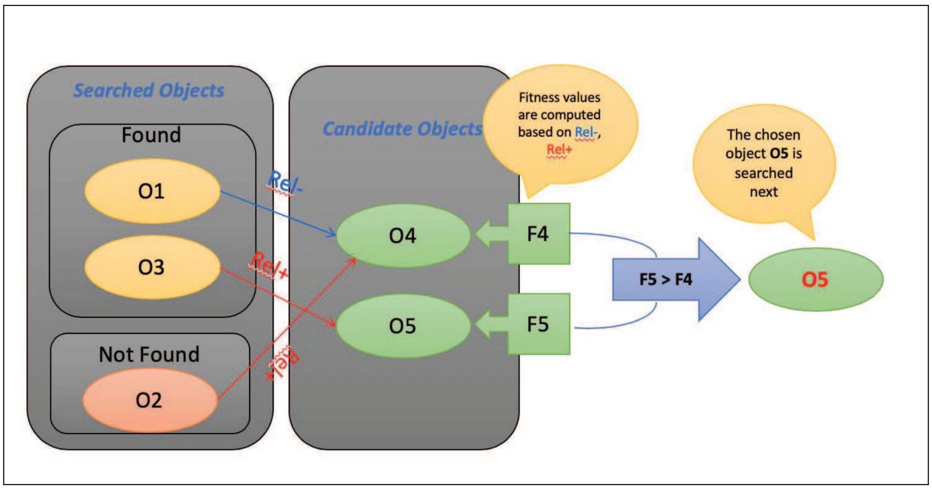

Searched Objects
Found
O1
O3
Not Found
O2
Rel-
Rel+
Rel+
Candidate Objects
O4
O5
F4
F5
Fitness values are computed based on Rel-, Rel+
F5 > F4
The chosen object O5 is searched next
O5

Figure 5: Description of the second inference rule

The scene recognition algorithm reasoning can be compromised if the first found object is a false positive. This can lead to filtering out other following correct detection as false positives and the final result becomes meaningless. Therefore, when choosing the next object to search for, the false positive rate of the object detector is also taken into consideration if no objects have been found so far. This reduces the risk of starting the algorithm with a false positive detection.

- *(R1)* Obtain the list of previously found objects rejected by the current candidate object $Objects_{REJECTED}$
- *(*R2) Compute the found object rejection factor REJ_{FOUND} according to each rejected object relation
- *(R3)* Obtain the list of previously found objects attracted by the current candidate object $Objects_{ATTRACTED}$
- *(R4)* Compute the found object attraction factor $ATTR_{FOUND}$ according to each attracted object relation
- *(R5)* Repeat the steps above for previously not found objects and compute $REJ_{NOT-FOUND}$ and $ATTR_{NOT-FOUND}$ factors.
- *(R6)* Compute the fitness of the current candidate object:

  $$F(O_{CAND}) = (ATTR_{FOUND} - REJ_{FOUND}) + \alpha * (REJ_{NOT-FOUND}) - ATTR_{NOT-FOUND})$$

  where α represent a sub-unit weight for decreasing the influence of the not found objects.

  If there was no object found, the fitness value is also influenced by the false positive rate (FP): FP$(Detector_{O_{CAND}}$) of the candidate objects:

  $$F(O_{CAND}) = (1 - FP(Detector_{O_{CAND}}) * F(O_{CAND})$$

- *(R7)* Choose the next object to search for from the list of candidate objects by finding the one with the biggest fitness value:

  $$O_{NEXT} = argmax(F(O_{CAND}))$$

The **third inference rule** has the role of determining the fitness value of each scene for the current image. If the scene with the highest fitness value is bigger than a threshold proportional to the average scene fitness, then the algorithm returns the scene and the object searching process is ended. For computing the fitness value of each scene the Inference Module uses the set of objects found so far to interrogate the **probabilistic ontology** about the probability of having each found object in the current candidate scene. The ontology relation that contains information about the probability of an object to appear in a scene is called **inScene**. This is a mandatory relation that should exist in any instance of the **probabilistic ontology**, no matter what object set and scene set is used or what other relations are used between objects. A visual description of the **third inference rule** is given in Figure 6.

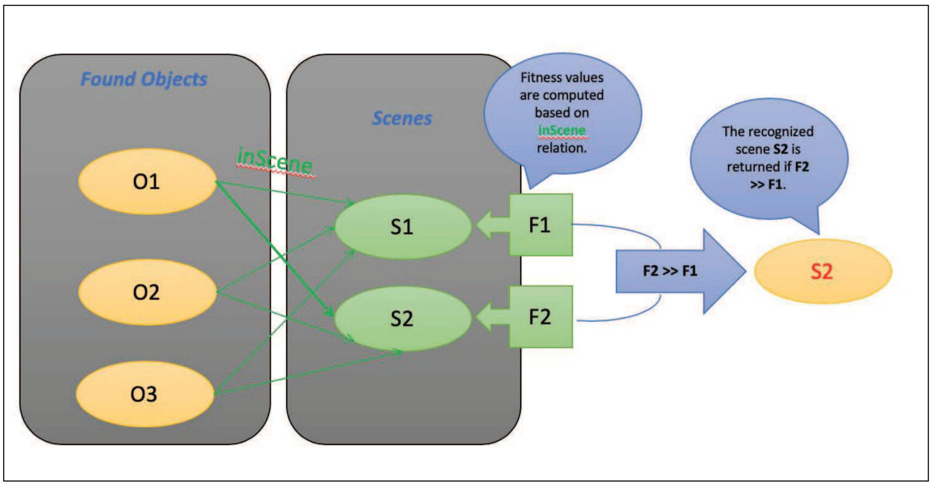

Found Objects
O1
O2
O3
inScene
Scenes
S1
S2
F1
F2
Fitness values are computed based on inScene relation.
F2 >> F1
The recognized scene S2 is returned if F2 >> F1.
S2

Figure 6: Description of the second inference rule

The fitness value for a scene S is computed as:

$$Fitness(S) = \sum_{O \in Objects_{FOUND}} log\frac{P(inScene(O,S)}{P(O)}$$

where $Objects_{FOUND}$ represents the set of the previously found objects, *P(inScene(O,S))* represents the probability of the ***inScene*** relation between object O and scene S and *P(O)* represents the apriori probability of object O (obtained from the ***probabilistic ontology***).

The best candidate scene is chosen by:

$$Scene_{BEST} = argmax(Fitness(S))$$

The best candidate scene is returned as the final recognized scene if it's fitness is bigger than a threshold proportional to the average scene fitness:

$$Fitness(Scene_{BEST}) > \delta * \frac{1}{|Scenes|} * \sum_{s \in Scenes} Fitness(S) \tag{1}$$

If no scene is recognized at the current iteration, then the object searching process continues and the previous three inferences are applied again at each iteration until a scene can be inferred or all the objects have been searched for. In the latter situation, the returned scene is the one with the biggest fitness value, without taking into consideration the equation 1.

4 System evaluation

The system is implemented to be flexible regarding replacing any of the components implementation. Every module communicates with the others through an interface, making the components loosely coupled. Therefore, in order to replace the current implementation of a module all that is needed is to implement the module interface. The scene recognition application is written in Java language using Eclipse as IDE. For object recognition algorithm we use the YOLO network [19]. The scene recognition application was tested using images from LabelMe database [15]. This database includes images for many indoor and outdoor scene types. For testing the system, only images with relevant scene types were used: kitchen, office, living room and street. The images are annotated with scene information under the attribute ***scenedescription***. After the ***probabilistic ontology*** probabilities are computed based on object and scenes co-occurrences inside the training image set, the model is stored inside an XML file. The XML ontology is then parsed by the application and mapped into Java objects that are used by the ***inference module***.

The pseudocode given in algorithm 1 that describes the reasoning algorithm implemented in the ***inference module***:

Algorithm 1 Reasoning algorithm

```
procedure REASONINGALGORITHM($Imageimage$)
    $searchObject \leftarrow getObjectWithLowestFP()$
    $foundObjectsSoFar \Leftarrow []$
    $notFoundObjectsSoFar \Leftarrow []$
    while true do
        $(confidence, boundBox) \Leftarrow findObject(searchObject, image)$
        $updatedConfidence \Leftarrow checkFoundObject(searchObject, confidence, foundObjectsSoFar)$
        if $updatedConfidence - confidence > threshold$ then
            $foundObjectsSoFar \Leftarrow foundObjectsSoFar \cup searchObject$
        else
            $notFoundObjectsSoFar \Leftarrow notFoundObjectsSoFar \cup searchObject$
            if $[] \neq foundObjectsSoFar$ then
                $sceneFitnessList \Leftarrow computeAllSceneProbabilities()$
                $bestScene \Leftarrow maxFitnessScene(sceneFitnessList)$
                $meanFitness \Leftarrow meanFitnessValue(sceneFitnessList)$
                if $bestScene.fitness > \epsilon * meanFitness$ then
                    return $bestScene$
                end if
            end if
        end if
        $searchObject = getNextMostProbableObj(foundObjSoFar, notFoundObjSoFar)$
    end while
end procedure
```

The ***object recognition module*** has access to a set of object detectors and to information regarding their performance: detection rate and false positive rate. We use YOLO (You Only Look Once) network [19]. It is a very robust method, which is almost invariant to position and lightning. It simultaneously predicts multiple bounding boxes and class probabilities for those boxes. YOLO trains on full images and directly optimizes detection performance. YOLO reasons globally about the image when making predictions. Unlike sliding window and region proposal-based techniques, YOLO sees the entire image during training and test time so it implicitly encodes contextual information about classes as well as their appearance. And also YOLO learns generalizable representations of objects.

For testing the system, only images with relevant scene types were used: kitchen, office, living room and street. The images are annotated with scene information under the attribute *scenedescription*. The database contains many object types that appear in many combinations. The images in the database are annotated with object information. For the current system, the relevant annotated object information is represented by the object outline polygon and the "verified": flag. The object polygon is used to obtain the object's bounding box for the object recognition stubs and the "verified" flag is used to select the object's confidence. If an object is verified, then it was annotated correctly and the object recognition will give it a high confidence, otherwise a lower confidence value is assigned. Some examples of LabelMe annotated images can be seen in Figure 7.

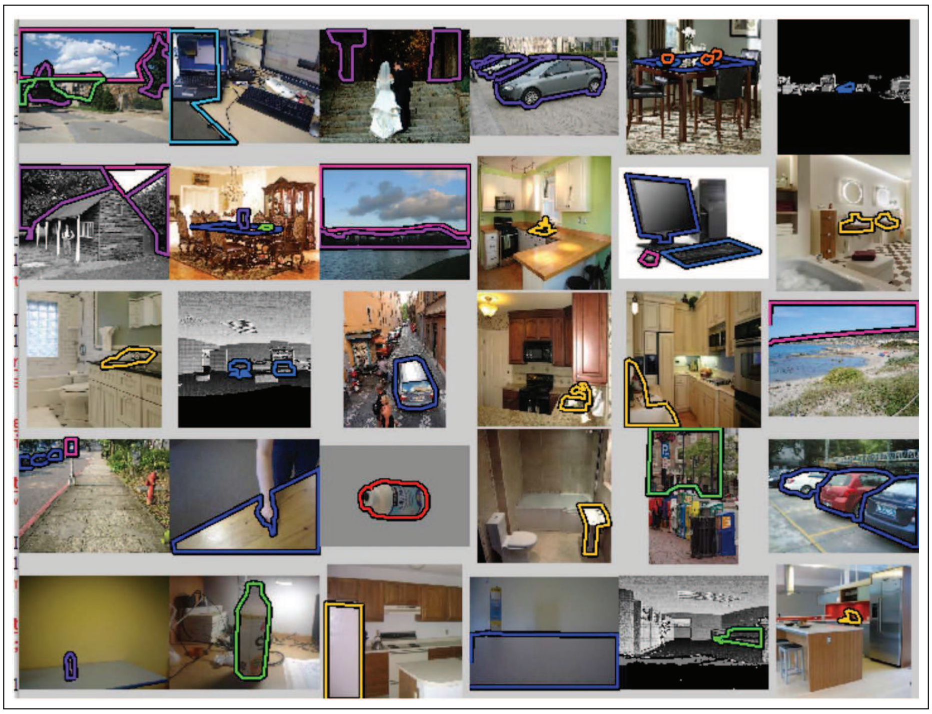


Figure 7: LabelMe annotated images example

Figure 8 shows some examples of recognising kitchen, office, living room. The current system was evaluated on a test set containing a total of 3381 annotated images from LabelMe database [15]. The aggregated evaluation results obtained after running the application on the test database can be seen in Table 2.

Table 2: System evaluation results using stubs

Scene Name	Accuracy	Mean Iterations	Removed FP	Test Images
kitchen	89%	7	16	651
office	90%	9	35	914
living	91%	8	30	775

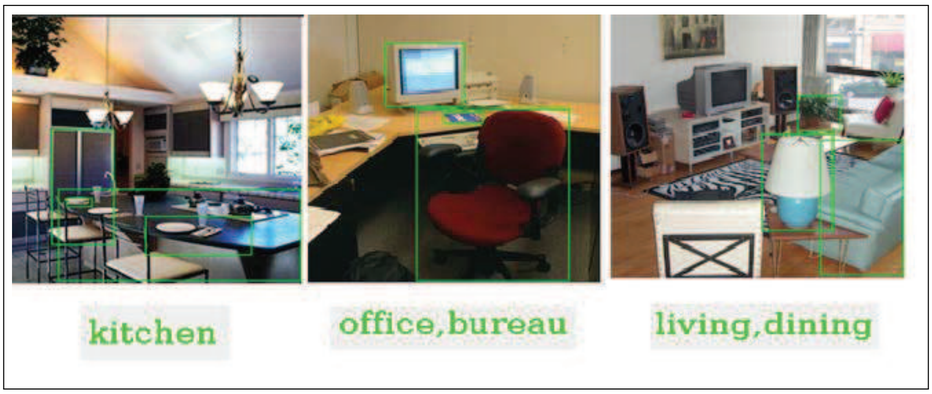


Figure 8: Example of scene recognition

5 Conclusions and future work

This paper presents a method for context detection of a user in an indoor space. The proposed method is based on the results of an object recognition process. In order to be able to recognize a wide range of scenes, the number of objects that need to be recognized can become very big. Running a large number of object detectors can be time consuming; therefore the current approach uses semantic information about objects and scenes to speed up the scene recognition process and to eliminate false positives that can have a negative impact on the final result. This semantic information is organized as an object and scene probabilistic ontology model. Reasoning is done using the stochastic relations between objects and between objects and scenes and its outcome can influence in which order objects are searched for or if a newly detected object is eliminated for being a false positive. Scene recognition is influenced by the semantic relations between scenes and objects, and the object recognition process ends as soon as there are enough objects found to determine the scene. Results show that given a set of object detectors with high detection rate and low false positive rate, the system can recognize a scene with high accuracy and in a small number of iterations.

As future work, the ontology model can be extended to meet domain-specific requirements because it is easily adaptable to different domains.

Acknowledgment

This research and paper was co-founded by both the Executive Unit for Financing Higher Education, Research and Development and Innovation through Partnership Program, project "Mobility pattern assistant for elderly people", project number PN-II-PT-PCCA-2013-4-2241, by University Politehnica of Bucharest, through the "Excellence Research Grants" Program, UPB - GEX. Identifier: UPB-EXCELENTA-2016 "Optimizarea Activitatilor Zilnice Folosind Deep Learning Implementata pe Sisteme Reconfigurabile" / Daily Activities Using Deep Learning Implemented on FPGA, Contract number 8 / 26.09.2016 (code 341) and by the grant of the Romanian National Authority for Scientific Research and Innovation, CCCDI - UEFISCDI and of the AAL Programme with co-funding from the European Union's Horizon 2020 research and innovation programme project "IONIS - Improving the quality of life of people with dementia and disabled persons", project number AAL2017-AAL-2016-074-IONIS (Contracts 52/20017 and 53/2017).

Bibliography

[1] Burm, C. (2015). Dementia and Elderly GPS Tracking Devices, http://www.aplaceformom.com/blog/4-29-15-dementia-and-elderly-gps-tracking-devices/, last accessed October 2018.

[2] DiCarlo, J.; Zoccolan, D.; Rust, N. C.(2012). How does the brain solve visual object recognition, *Neuron*, 73(3), 415–434, 2012.

[3] Enns, J. T.; (2004). *The Thinking Eye, The Seeing Brain: Explorations in Visual Cognition*, W. W. Norton Company, ISBN: 0393977218, 2004

[4] Fischler, M.A. ; Elschlager, R.A., (1973). The Representation and Matching of Pictorial Structures, *IEEE Transactions on Computer*, 22(1), 67–92, 1973.

[5] Felzenszwalb, P.F, Huttenlocher, D.P.; (2005), Pictorial Structures for Object Recognition, *International Journal of Computer Vision*, 61(1):55–79, 2005.

[6] Gupta, S.; Girshick, R.; Arbelaez, P.; Malik, J. (2014). Learning Rich Features from RGB-D Images for Object Detection and Segmentation, *ECCV*, 345–360, 2014.

[7] Kaiser, L.; Gomez, A. N.; Shazeer, N.; Vaswani, A.; Parmar, N.; Jones,l.; Uszkoreit,J (2017). One Model To Learn Them All, http://arxiv.org/abs/1706.05137, last accessed October 2018.

[8] Li, B.; Wu, T.; Shuai1, S.; Zhang, L.; Chu, R., (2017). Object Detection via Aspect Ratio and Context Aware Region-based Convolutional Networks, arXiv:1612.00534v2 , https://arxiv.org.

[9] Leal-Taixe, L. (2016). Multiple Object Tracking with Context Awareness, http://arxiv.org/abs/1411.7935, last accessed October 2018.

[10] Maturana, D.; Scherer., S. (2015). Voxnet: A 3d Convolutional Neural Network for Realtime Object Recognition, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 922–928, 2015.

[11] Napoletan, A. (2015). 10 Best (and Worst) Apps for Caregivers, https://www.aplaceformom.com/blog/best-and-worst-apps-for-caregivers-07-03-2013/, last accessed October 2018.

[12] Perko, R.; Leonardis, A., (2010). Context Awareness for Object Detection, *Computer Vision and Image Understanding*, 114(6), 700–711, 2010.

[13] Rehman, Z.; Kifor C.K. (2016). An Ontology to Support Semantic Management of FMEA Knowledge, *International Journal of Computers Communications* & *Control*, 11(4), 507-521, 2016.

[14] Ren, S.; He, K.; Girshick, R.B.; Sun, J. (2015). Faster R-CNN: towards real-time object detection with region proposal networks, *Advances in Neural Information Processing Systems*, 91–99, 2015.

[15] Russell, B. C.; Torralba, A.; Murphy, K. P.; Freeman, W. T. (2008). LabelMe: a Database and Web-Based Tool for Image Annotation, *International Journal of Computer Vision*, 77(1-3), 157–173, 2008.

[16] Sabour, S.; Frosst, N.; Hinton, G. E. (2017). Dynamic Routing Between Capsules, Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA,3859–3869, https://arxiv.org/pdf/1710.09829.pdf, last accessed October 2018.

[17] Szegedy, C.; Ioffe, S.; Vanhoucke, V. (2016). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning, http://arxiv.org/abs/1602.07261, last accessed October 2018.

[18] Viola, P. A.; Jones, M. J. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features, *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1-9, 2001.

[19] YOLO network, https://pjreddie.com/darknet/yolo/, last accessed October 2018.

[20] https://senion.com/indoor-positioning-system/, last accessed October 2018.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(6), 1007-1031, December 2018.

Simulation of Rapidly-Exploring Random Trees in Membrane Computing with P-Lingua and Automatic Programming

I. Pérez-Hurtado, M.J. Pérez-Jiménez, G. Zhang, D. Orellana-Martín

Ignacio Pérez-Hurtado*, Mario J. Pérez-Jiménez, David Orellana-Martín
Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Seville, Seville, Spain
{perezh,marper,dorellana}@us.es
*Corresponding author: perezh@us.es

Gexiang Zhang
School of Electrical Engineering
Southwest Jiaotong University, Chengdu, Shichuan, China
zhgxdylan@126.com

Abstract: Methods based on Rapidly-exploring Random Trees (RRTs) have been widely used in robotics to solve motion planning problems. On the other hand, in the membrane computing framework, models based on Enzymatic Numerical P systems (ENPS) have been applied to robot controllers, but today there is a lack of planning algorithms based on membrane computing for robotics. With this motivation, we provide a variant of ENPS called Random Enzymatic Numerical P systems with Proteins and Shared Memory (RENPSM) addressed to implement RRT algorithms and we illustrate it by simulating the bidirectional RRT algorithm. This paper is an extension of [21][a]. The software presented in [21] was an ad-hoc simulator, i.e, a tool for simulating computations of one and only one model that has been hard-coded. The main contribution of this paper with respect to [21] is the introduction of a novel solution for membrane computing simulators based on automatic programming. First, we have extended the P-Lingua syntax –a language to define membrane computing models– to write RENPSM models. Second, we have implemented a new parser based on Flex and Bison to read RENPSM models and produce source code in C language for multicore processors with OpenMP. Finally, additional experiments are presented.

Keywords: Membrane Computing, RENPSM, robotics, RRT, P-Lingua.

[a] Reprinted (partial) and extended, with permission based on License Number 501431225 ©[2018] IEEE 7th International Conference on Computers Communications and Control (ICCCC)

1 Introduction

Robots are machines oriented to objectives equipped with actuators, sensors and computation units acting under physical constraints. Regardless of their morphology, they should accomplish tasks by acting in the real world. This is one of the main reasons by which robot motion planning is an eminent research area in robotics [8, 11, 21]. In general terms, the problem of motion planning can be defined in the configuration space of a robot as follows: *Given a start configuration state, a goal configuration state, a geometric description of the robot, and a geometric description of the environment, find a path that moves the robot gradually from start to goal.*

A configuration state is a specification of the positions of all robot points relative to a fixed coordinate system. This is usually expressed as a vector of positions and orientations, for example, a rigid-body robot in a 2D world can be expressed as a vector (x, y, θ) representing the center (x, y) of the robot in a fixed coordinate system and its yaw angle θ, i.e, the heading angle of the robot. Since the shape of the robot is described, all of its points are then known.

Several constraints can be added to this problem, the most common is to reach the goal while never touching any obstacle in the environment. Others can also be added, for example, a social robot could restrict configuration states in order to guarantee the human comfort.

The configuration space of a robot can also be constrained by the type of movements the robot can perform. In this sense, nonholonomic robots are those that cannot instantly modify its direction without employing rotation in-place. On the other hand, holonomic robots can do it (assuming zero mass). For example, a holonomic robot in a 2D world can move along the x axis and the y axis, as well as modify its yaw angle if needed. But a nonholonomic robot can only move forward/backward and/or modify its yaw angle. This is the typical case of dual-wheeled mobile robots and cars.

Classical path planning algorithms have been widely adapted and applied to the problem of motion planning with constraints in robots, for example, in [26], an application of the Dijkstra algorithm for robot path-planning was presented. In such solutions, the general problem is usually divided into two smaller problems: the *global path planning* problem, as described above; and the *local path planning*, where the robot tries to connect two consecutive states in real-time considering features not included in the global plan as, for example, dynamic obstacles. The accumulated error during the local planning conducts to periodically recompute the global plan. For this reason, the computational complexity of global planners is a critical point regarding to real-time constraints. Many efforts have been made to provide good global planners. For example: in [25], a search algorithm, called D^*, was presented for path planning in real-time environments. In [15], a variant of the classical search algorithm A^* is applied to grids with blocked and unblocked cells. In [12], a tool for global path planning, called Rapidly-Exploring Random Trees (RRT), was presented.

The class of RRT algorithms for global path planning is based on the randomized exploration of the configuration space before moving the robot by building a tree in memory where nodes represent states that can be reached by the state of the corresponding parent in a fixed amount of time, furthermore each edge contains a velocity reference to reach the state in the child node from the state in the parent node. It is currently one popular method in robot motion planning due to its good properties. The computed RRT can be used together with search algorithms or, as presented in [13], the RRT generation algorithm can be used by itself as a path planning algorithm, where two RRTs are built simultaneously, one beginning from the initial configuration and another one beginning from the ending configuration (*bidirectional RRT*).

In order to follow the path in safe manner, a local planner module should be executed considering dynamic obstacles. Finally, each motor of the robot must be able to reach and maintain velocity references for fixed periods of time. This is the function of a type of software called *controller* on-board of the robot. Thus, *robot control* [1] is the branch of robotics dedicated to the study and practice of controlling robots.

Robot controllers are usually based on common silicon microprocessors, but in the recent years, some classes of *membrane systems* [16] have been in use for modelling them [18] [19] [20] [29]. Membrane systems are models of computation based on the structure and functions of the living cells. In a membrane system, there are objects being evolved inside compartments according to rules applied in a non-deterministic, maximally parallel way. They have been used as a new technique to attack the P versus NP problem [22], and several applications have been also studied: stochastic P systems for modelling biological phenomena [24], probabilistic P systems for modelling real ecosystems [2], spiking neural P systems incorporating fuzzy reasoning, for fault diagnosis and learning [27], and others.

With respect to robot control, numerical P systems (NPS) were used for modelling and simulating robot controllers [20], although the initial application of NPS was related to economical processes [17]. A variant called *enzymatic numerical P systems* (ENPS) [18] was introduced and applied to the distributed control of a swarm of mobile robots. Indeed, reactive and proportional-integral-derivative (PID) dual-wheeled robot controllers have been successfully designed and simulated by means of ENPS, as well as software simulation tools [29]. This variant has been also used [19] to address *robot localization* problem [8], where the robot must know its position in the environment by using sensors.

In [21], following [23], a new variant of ENPS called *random enzymatic numerical P systems with proteins and shared memory* (RENPSM, for short) was introduced. New syntactical ingredients were included to fit the requirements of the RRT algorithm:

- *Random numbers*: The algorithm uses a randomized method to explore the physical space, therefore random numbers must be generated.

- *Shared memory*: The algorithm is parallelized using processes sharing common variables, and a distinguished membrane, called *shared memory*, is included. At any instant, each membrane can read from or write to it.

- *Proteins*: In order to synchronize the sequential execution of the algorithm, proteins are used.

This paper is an extension of [21]. The software presented in [21] was an ad-hoc simulator, i.e, a tool to simulate computations of one and only one model that has been hard-coded. Ad-hoc simulators can be optimized for specific hardware architectures, but they are less debug-friendly than generic simulators, since changes in the model imply changes in the source code of the simulator. On the other hand, P-Lingua [3, 7, 33] is a language to define membrane computing models, there are several simulators based on P-Lingua, most of them are implemented in the pLinguaCore library [33] in Java language. In this paper, we provide a novel approximation taking the advantages of ad-hoc and generic simulators by using automatic programming. We have implemented a tool for parsing P-Lingua files defining RENPSM models and generating source code in C and OpenMP for ad-hoc simulators. Thus, we have a flexible way to debug since we are using a language to define the models instead of hard-coding them in the source code. Moreover, the generated source code is able to run on multicore processors by using OpenMP. Furthermore, additional virtual experiments are presented in this paper.

This paper is structured as follows. In the next section, some notions about robot path planning are introduced. In Section 3, the rapidly-exploring random trees (RRTs) are described with some details. Section 4 is devoted to introduce random enzymatic numerical P systems with proteins and shared memory. In Section 5, a RENPSM model for the bidirectional RRT algorithm is described. In Section 6, the original software presented in [21] based on C++ and ROS [30] is explained, including some experimental results. In Section 7, the new software presented in this paper is introduced, including an extension of the P-Lingua syntax for RENPSM, as well as additional experiments. Finally, conclusions and future work are drawn.

2 Robot path planning

In general terms, robot path planning can be solved by applying a solution based on three modules:

- *Global planner*: It receives the desired ending configuration of the robot, its safety radius and current localization, as well as the precomputed position of the static obstacles in the environment and current information of sensors and odometry. The current dynamic obstacles detected by the sensors are added to the static ones in order to generate a more descriptive information of the environment. The odometry is used to obtain the current velocity of the robot when kinodynamic constraints are considered. Then, the global planner computes a plan from the starting configuration x_{init} to the desired final configuration x_{end} of the robot. The plan is represented as a sequence of local goals $\{g_i | 1 \leq i \leq n\}$, where $g_1 = x_{init}$ and $g_n = x_{end}$. Each goal can be reachable from the previous goal considering the constraints of the problem, i.e, avoiding static obstacles, nonholonomic and kinodynamic constraints, etc. RRT algorithms and other similar algorithms can be used for this task.

- *Local planner*: It receives the sequence of local goals generated by the global planner, as well as the current information of sensors, localization and odometry, then it sends velocity references to the controller in order to command the robot along the path. Several algorithms such as *the dynamic window approach* [5], *pure pursuit* [4], and *potential fields* [10] algorithms, among others and variants, can be used.

- *Controller*: It receives velocity references from the local planner and manages the power of the motors to fit each reference and maintain it constant or accelerate or reduce it until the next one.

In Figure 1, it is represented the general robot path planning cycle. First the robot computes a global plan from its current pose to the desired ending pose; if the plan can be generated, i.e, the robot could reach the destination considering all the constraints, then the local planner receives a sequence of intermediate goals and sends velocity references to the PID controller in order to follow the path in a safe manner until reaching the destination; if some error occurs, for example, a dynamic obstacle is too close to the next local goal or the robot is too far from the next local goal (considering a predefined threshold), then the global plan is recomputed from the current robot position. Notice that if there is a dynamic obstacle too close to the ending configuration, then the global plan cannot be found.

3 Rapidly-exploring Random Trees

An RRT [12] is a randomized tree structure for rapidly exploring in memory a *state space* X from an initial state x_{init}. It can be successfully used for nonholonomic and kinodynamic global path planning in robotics [13].

Nodes in an RRT represent possible reachable states, for mobile robots in a 2D world which is given by (x, y, θ) where (x, y) are the Cartesian coordinates of the robot position and θ is the heading angle. However, the heading angle can be omitted in order to reduce the size of the tree.

It is assumed that a fixed *obstacle region* $X_{obs} \subseteq X$ must be avoided, so the nodes of the RRT are states in X_{free}, the complement of X_{obs} in X.

Edges in an RRT represent transitions between reachable states, each of which is labelled with the velocity reference u that the robot should execute for a fixed period of time Δt in order to change the corresponding states. For a mobile robot in a 2D world, the velocity reference can be represented by the pair of linear and angular velocities (v, ω) to be sent to the controller. On the other hand, if θ has been omitted, the edges in the RRT are labelled with instant linear velocities. Thus, a holonomic robot can reach a state x_1 from another state x_0 connected by an edge labelled with u by applying $x_1 = x_0 + u \cdot \Delta t$. In the case of nonholonomic robots, the local planner should select the best sequence of motions in order to approximate x_1, for example, if

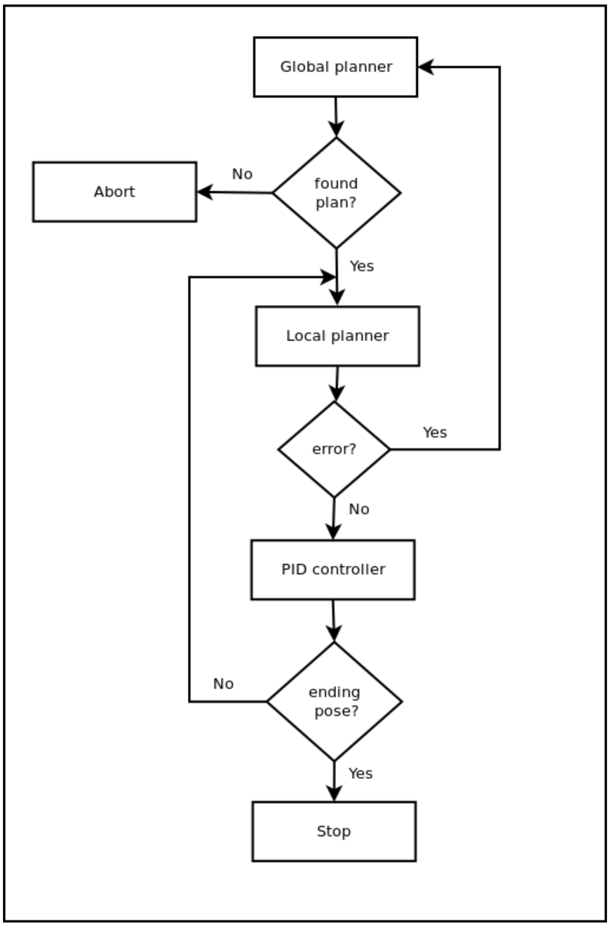


Figure 1: Robot path planning cycle

the robot can rotate in-place, a naive solution is to perform a rotation in-place before developing the motion in a straight line.

Algorithm 1 GENERATE_RRT

```
Require: $x_{init}, K, \rho, \Delta t, X, X_{obs}, d_{min}$
$V_\tau \leftarrow \{x_{init}\}$; $E_\tau \leftarrow \emptyset$;
for $k = 1$ to $K$ do
    $x_{rand} \leftarrow$ RANDOM_STATE($X$);
    EXTEND($\tau, x_{rand}, \rho, \Delta t, X_{obs}, d_{min}$);
end for
return $\tau = (V_\tau, E_\tau)$
```

Algorithm 1 is an iterative method to generate an RRT using the function EXTEND defined in Algorithm 2, where:

- x_{init} is the initial state.
- K is the number of iterations to build the RRT.
- ρ is a prefixed distance metric.
- Δt is a fixed amount of time for transitions.

Algorithm 2 EXTEND

```
Require: $\tau, x, \rho, \Delta t, X_{obs}, d_{min}$
$x_{near} \leftarrow$ NEAREST_NEIGHBOR($x, \tau$);
if DISTANCE($x, x_{near}$) $\geq d_{min}$ then
    $u \leftarrow$ SELECT_INPUT($x, x_{near}$);
    if $\neg$ COLLISION($x_{near}, u, \Delta t, X_{obs}$) then
        $x_{new} \leftarrow$ NEW_STATE($x_{near}, u, \Delta t$);
        $V_\tau \leftarrow V_\tau \cup \{x_{new}\}$
        $E_\tau \leftarrow E_\tau \cup \{(x_{near}, x_{new})\}$
        if DISTANCE($x$, $x_{new}$) $< d_{min}$ then
return Reached;
        else
return Advanced;
        end if
    else
return Trapped;
    end if
else
return Trapped;
end if
```

- X is the state space.
- X_{obs} is the obstacle state space.
- d_{min} is the minimum distance threshold according to ρ in order to include a new node in the RRT.
- $\tau = (V_\tau, E_\tau)$ is the RRT generated.
- RANDOM_STATE(X) is a function to get a random state from X
- NEAREST_NEIGHBOR(x, τ) is a function to get the closest node to x in τ according to ρ.
- DISTANCE(x, x_{near}) is a function to get the distance of x to x_{near} according to ρ.
- SELECT_INPUT(x, x_{near}) is a function to get the velocity input that should be commanded to the robot in order to achieve state x from x_{near}.
- COLLISION($x_{near}, u, \Delta t, X_{obs}$) is a function returning *true* if a collision could be produced moving the robot from state x_{near} by applying the input u for Δt time considering the obstacles in X_{obs}.
- NEW_STATE($x_{near}, u, \Delta t$) is a function to get a new state x_{new} by applying the input u to the robot for Δt time starting at state x_{near}.

The function EXTEND tries to add a new node to the RRT τ considering a reference x. If the function fails, then it returns *Trapped*; if the new node is closer than d_{min} to x, then it returns *Reached*; and if the new node is far from x considering d_{min}, then the function returns *Advanced.* Figure 2 describes a RRT generated after 5000 iterations by using Algorithm 1 with the Euclidean distance and omitting the heading angle.

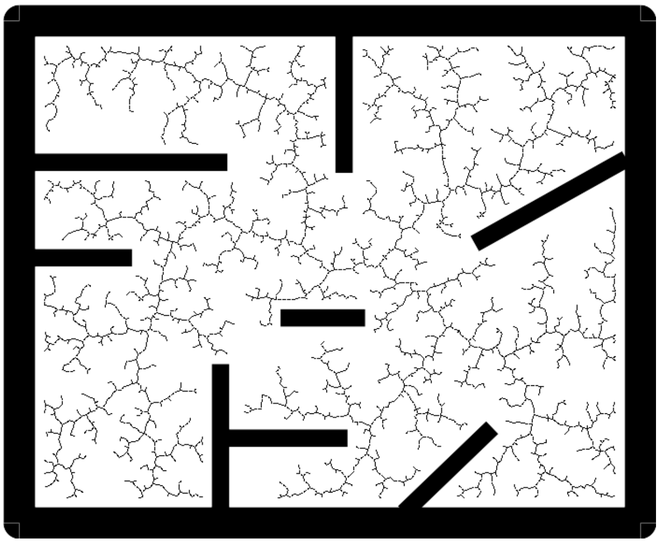


Figure 2: RRT generated after 5000 iterations

In [13], a bidirectional RRT algorithm was introduced for path planning. The main idea of this algorithm is to create two RRTs: τ_a starting at x_{init} and τ_b starting at x_{end}. If τ_a and τ_b are connected in a prefixed number K of iterations, then a path is returned; otherwise the function returns failure.

Algorithm 3 GENERATE_PATH

```
Require: $x_{init}, x_{end}, K, \rho, \Delta t, X, X_{obs}, d_{min}$
$V_{\tau_a} \leftarrow \{x_{init}\}$; $E_{\tau_a} \leftarrow \emptyset$;
$V_{\tau_b} \leftarrow \{x_{end}\}$; $E_{\tau_b} \leftarrow \emptyset$;
for $k = 1$ to $K$ do
    $x_{rand} \leftarrow$ RANDOM_STATE($X$);
    if EXTEND($\tau_a, x_{rand}, \rho, \Delta t, X_{obs}, d_{min}$) $\neq$ Trapped then
        if EXTEND($\tau_b, x_{new}, \rho, \Delta t, X_{obs}, d_{min}$) $=$ Reached then return PATH($\tau_a$, $\tau_b$);
        end if
    end if
    SWAP($\tau_a$, $\tau_b$);
end for
return Failure
```

Algorithm 3 is the bidirectional RRT algorithm presented in [13], where:

- x_{init} is the initial state.
- x_{end} is the ending state.
- $\tau_a = (V_{\tau_a}, E_{\tau_a})$ is an RRT starting at x_{init}.
- $\tau_b = (V_{\tau_b}, E_{\tau_b})$ is an RRT starting at x_{end}.
- PATH(τ_a, τ_b) is a function to compute a path from the initial node of τ_a to the initial node of τ_b. Both RRTs must be connected.
- SWAP(τ_a, τ_b) is a procedure to interchange the values of τ_a and τ_b.

The rest of variables have the same meaning than the variables used in Algorithms 1 and 2.

In this paper we propose the Algorithm 4 as a parallel version of the bidirectional RRT algorithm, where τ_a and τ_b are built at the same time.

Algorithm 4 GENERATE_PATH_PARALLEL

```
Require: $(x_{init}, x_{end}, K, \rho, \Delta t, X, X_{obs}, d_{min}$
$V_{\tau_a} \leftarrow \{x_{init}\}$; $E_{\tau_a} \leftarrow \emptyset$;
$V_{\tau_b} \leftarrow \{x_{end}\}$; $E_{\tau_b} \leftarrow \emptyset$;
for $k = 1$ to $K$ do
    $x_{rand,a} \leftarrow$ RANDOM_STATE($X$);
    $x_{rand,b} \leftarrow$ RANDOM_STATE($X$);
    loop
        $result_a$ = EXTEND($\tau_a, x_{rand,a}, \rho, \Delta t, X_{obs}, d_{min}$);
        $result_b$ = EXTEND($\tau_b, x_{rand,b}, \rho, \Delta t, X_{obs}, d_{min}$);
    end loop
    if $result_a \neq$ Trapped then
        if EXTEND($\tau_b, x_{new}, \rho, \Delta t, X_{obs}, d_{min}$) = Reached then return PATH($\tau_a$, $\tau_b$);
        end if
    end if
    if $result_b \neq$ Trapped then
        if EXTEND($\tau_a, x_{new}, \rho, \Delta t, X_{obs}, d_{min}$) = Reached then return PATH($\tau_a$, $\tau_b$);
        end if;
    end if;
end for;
return Failure
```

4 Random enzymatic numerical P systems with proteins and shared memory

In this section a variant of *enzymatic numerical P systems* incorporating new features is presented, in order to simulate RRT algorithms.

A random enzymatic numerical P systems with proteins and shared memory (RENPSM, for short) of degree (p, q), $p, q \geq 1$ is a tuple $(H, \mu, P, E_{mem}, E_{mem}(0), \{P_h(0), Var_h, Var_h(0), Pr_h) \mid h \in H\}, \mathcal{R}, h_a, h_b)$, where:

1. $H = \{1, \ldots, p \cdot q\} \cup \{v, mem\}$, $mem \notin \{1, \ldots, p \cdot q\}$, $v \notin \{mem, 1, \ldots, p \cdot q\}$, is the set of labels of the system;

2. μ is a dynamic membrane structure (rooted tree) initially consisting of one skin membrane with label v including two inner membranes labelled respectively with $h_a \in \{1, \ldots, p \cdot q\}$ and $h_b \in \{1, \ldots, p \cdot q\}$, $h_a \neq h_b$, in such manner that along the computation only child membranes of h_a and h_b will be created with labels in $\{1, \ldots, p \cdot q\}$. In Figure 3, it is represented the initial membrane structure;

3. *mem* is the label of a distinguished component (the shared memory of the system);

4. P is a finite set of objects, called catalyzer proteins, and $P_h(0)$ is the protein initially associated with region labelled by h;

5. E_{mem} is a finite set of variables, called enzymes, disjoint with Var_{mem}, and $E_{mem}(0)$ is the initial values of the enzymes;

6. $Var_h, h \in H$, is a finite set of variables $x_{j,h}$ associated with region labelled by h (a membrane or the shared memory), its values are natural numbers and the value of $x_{j,i}$ at time $t \in \mathbf{N}$ is denoted by $x_{j,i}(t)$;

7. $Var_h(0)$ is a vector that represents the initial values for variables in Var_h;

8. $Pr_h, h \in H$, is a finite set of programs associated with region labelled by h, having the following syntactical format $F(x_{1,h}, \ldots, x_{k_F,h}) \xrightarrow{e(F);\alpha(F)} c_1|v_1, \ldots, c_{n_F}|v_{n_F}$, where:

   - $F(x_{1,h}, \ldots, x_{k_F,h})$ is a computable function (the production function), being $x_{1,h}, \ldots, x_{k_F,h} \in Var_h$;
   - $c_1|v_1, \ldots, c_{n_F}|v_{n_F}$ is the repartition protocol associated with the program, being $c_1, \ldots, c_{n_F}$ natural numbers specifying the proportion of the current production distributed to variables $v_1, \ldots, v_{n_F} \in Var_h \cup Var_{par(h)} \cup Var_{ch(h)}$, where $par(h)$ is the parent of h and $ch(h)$ is the set of child of h in μ;
   - $e(F) \in E_h$ is an enzyme and $\alpha(F) \in P$ is a protein, both of them associated with program F, if no enzyme or protein is used in a program then it will be omitted;

9. $\mathcal{R}$ is a finite set of rules of the following form:

   - *Protein evolution rules*: $[\,\alpha \to \alpha'\,]_h$, where $h \in H, \alpha \in P$ and $\alpha' \in P$.
   - *Writing-only communication rules between the shared memory and the membranes*

     $$(h\,,\, X_h\,/Y_{h,mem}\,,\, mem)^W_\alpha$$

     where $X_h \in Var_h, Y_{h,mem} \in Var_{mem}, \alpha \in P$ in such manner that there is, at most, one rule for each membrane $h \in \{1, \ldots, p \cdot q\}$. Variables $Y_{h,mem}, Y_{h',mem}$ should be different for two membranes h, h'.
   - *Reading-only communication rules between the shared memory and the membranes:*

     $$(h\,,\, X_h\,/Y_{mem}\,,\, mem)^R_\alpha$$

     where $X_h \in Var_h, Y_{mem} \in Var_{mem}, \alpha \in P$. Variable Y_{mem} is the same for each $h \in \{1, \ldots, p \cdot q\}$.
   - *Membrane creation rules*:

     $$\Big[\ \Big[\ X_{1,h}\,,\, X_{2,h},\,,\,\ldots\,,\, X_{n,h}\ \Big]_h\ \Big]_{h'}\ ;\,\alpha$$

     where $h, h' \in \{1, \ldots p \cdot q \cdot r\}$ are different, $\alpha \in P$ and $X_{1,h}\,,\,\ldots\,,\,X_{n,h} \subseteq Var_h$.

The term *region* h ($h \in H$) is used to refer to membrane h in the case $h \in \{1, \ldots, p \cdot q\} \cup \{v\}$, as well as to refer to the shared memory in the case $h = mem$.

Next, we describe the semantics of RENPSHs. A *configuration* of a RENPSH at any instant t is described by the current membrane structure μ, together with proteins and all values of the variables and enzymes associated with all regions.

The *initial configuration* is $(\mu, E_{mem}(0), \{P_h(0), Var_h(0)|h \in H\})$, where $\mu = [[\,]_{h_a}[\,]_{h_b}]_v$. We will call μ_a (resp. μ_b) to the membrane structure rooted in membrane h_a (resp. h_b).

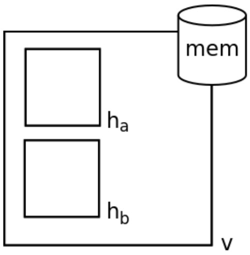


Figure 3: The initial membrane structure with a representation of the shared memory.

A program $F(x_{1,h},\dots,x_{k_F,h}) \overset{e_F;p_F}{\longrightarrow} c_1|v_1,\dots,c_{n_F}|v_{n_F}$ associated with a region is applicable to a configuration $\mathcal{C}_t$ at moment t, if the value of $e(F)$ at that instant is greater than $min\{x_{1,h}(t),\dots,x_{k_F,h}(t)\}$ and protein $\alpha(F)$ is inside the region h of $\mathcal{C}_t$. When applying such a program, variables associated with $\mathcal{C}_t$ are processed as follows: first, the value $F(x_{1,h}(t),\dots,x_{k_F,h}(t))$ is computed as well as the value

$$q(t) = \frac{F(x_{1,h}(t),\dots,x_{k_F,h}(t))}{c_1+\cdots+c_{n_F}}$$

This value represents the *unary portion* at instant t to be distributed among variables $v_1,\dots,v_{n_F}$ according to the repartition expression. Thus, $q(t)\cdot c_s$ is the contribution added to the current value of v_s $(1 \le s \le n_F)$, at step $t+1$. So, $v_s(t+1) = v_s(t) + q(t)\cdot c_s$ and $v_s(t)$ become zero, i.e, it is assumed that variable v_s is "consumed" when the production function is used and other variables retain their values. Each program in each membrane can only be used once in every computation step, and all the programs are executed in parallel.

A protein evolution rule $[\,\alpha \to \alpha'\,]_h$ is applicable to a configuration $\mathcal{C}_t$ at moment t if protein α is in membrane h of $\mathcal{C}_t$. When applying such a rule the protein α in h evolves to protein α' in h. These rules are applied in a maximal manner.

A writing-only communication rule between the shared memory and the membranes,

$$(h\,,\,X_h\,/Y_{h,mem}\,,\,mem)^W_\alpha$$

,

is applicable to a configuration $\mathcal{C}_t$ at moment t if protein α is in membrane h of $\mathcal{C}_t$. When applying such a rule the value $X_h(t)$ is assigned to the variable $Y_{h,mem}(t+1)$ of the shared memory, that is $Y_{h,mem}(t+1) \leftarrow X_h(t)$. These rules are applied in a maximal manner.

A reading-only communication rule between the shared memory and the membranes,

$$(h\,,\,X_h\,/Y_{mem}\,,\,mem)^R_\alpha$$

is applicable to a configuration $\mathcal{C}_t$ at moment t if protein α is in membrane h of $\mathcal{C}_t$. When applying such a rule the value $Y_{mem}(t)$ is assigned to the variable $X_h(t+1)$ of membrane h, that is $X_h(t+1) \leftarrow Y_{mem}(t)$. These rules are applied in a maximal manner.

A membrane creation rule

$[\,[X_{1,h}\,,\,\dots\,,\,X_{n,h}\,]_h\,]_{h'}\,;\,\alpha$ is applicable to a configuration $\mathcal{C}_t$ at moment t if protein α is in membrane h' of $\mathcal{C}_t$. When applying such a rule, a new membrane labelled by h is created in such manner that h' is the parent of h and the set of its variables is $Var_h = \{X_{1,h}\,,\,\dots\,,\,X_{n,h}\}$.

Given a random enzymatic numerical P system with proteins and shared memory Π, we say that configuration $\mathcal{C}_t$ at time t yields configuration $\mathcal{C}_{t+1}$ in one transition step if we can pass from $\mathcal{C}_t$ to $\mathcal{C}_{t+1}$ by applying in parallel each program in each membrane only once, and by applying the rules in a maximal parallel way following the previous remarks. A *computation* of Π is a (finite or infinite) sequence of configurations such that: (a) the first term is the initial configuration of the system; (b) for each $n \geq 2$, the n-th configuration of the sequence is obtained from the previous configuration in one transition step; and (c) if the sequence is finite (called *halting computation*) then the last term is a *halting configuration* (a configuration where no rule of the system is applicable to it). All the computations start from an initial configuration and proceed as stated above; only halting computations give a result, which is encoded by the objects present in the output region i_{out} associated with the halting configuration. If $\mathcal{C} = \{\mathcal{C}_t\}_{t<r+1}$ of Π ($r \in \mathbb{N}$) is a halting computation, then the *length* of $\mathcal{C}$, denoted by $|\mathcal{C}|$, is r. For each i ($1 \leq i \leq q$), we denote by $\mathcal{C}_t(i)$ the finite multiset of objects over Γ contained in all membranes labelled by i at configuration $\mathcal{C}_t$.

5 Simulation of one iteration of the bidirectional RRT algorithm for path planning

The input of the bidirectional RRT algorithm generating a global path for a robot trajectory consists of the following parameters $(x_{init}, x_{end}, K, \rho, \Delta t, X, X_{obs}, d_{min})$, where:

- x_{init} is the initial state.
- x_{end} is the ending state.
- K is the number of iterations to find the path.
- Δt is a fixed amount of time for transitions.
- X is the state space.
- $X_{obs} \subseteq X$ is the obstacle state space.
- d_{min} is the minimum distance threshold according to some distance metric ρ in order to include a new node in the RRT.

For mobile robots in a 2D environment, the state space is given by (x, y, θ), i.e, the Cartesian coordinates (x, y) and the heading angle θ of all the possible robot poses. However, the angle θ has been omitted in this solution to reduce the size of the problem and the state space is given by (x, y) considering the Euclidean distance as distance metric. In this case, a holonomic robot can follow the RRT by performing motions in straight line, otherwise a nonholonomic robot can include rotations in-place. Moreover, any state or position $(i, j) \in \{0, \dots p-1\} \times \{0, \dots q-1\}$ in an Euclidean space constrained by a rectangle of p width and q height distance units can be encoded by the natural number $j \cdot p + i + 1$. In such a manner, given a natural number n encoding a state (i, j), the following holds: $i = rm(n-1, p)$ and $j = qt(n-1, p)$.

One iteration of the parallel bidirectional RRT algorithm defined in Algorithm 4 will be simulated by a RENPSM of degree (p, q)
$\Pi = (H, \mu, P, E_{mem}, E_{mem}(0),$
$\{(P_h(0), Var_h, Var_h(0), Pr_h) \mid h \in H\}, \mathcal{R}, h_a, h_b)$
defined as follows:

- $H = \{1, \dots, p \cdot q\} \cup \{v, mem\}$, $v \notin \{1, \dots, p \cdot q\}$, $mem \notin \{1, \dots, p \cdot q\}$.
- $\mu = [[\,]_{h_a}[\,]_{h_b}]_v$ with $h_a \in \{1, \dots, p \cdot q\}$, $h_b \in \{1, \dots, p \cdot q\}$ and $h_a \neq h_b$. We call μ_a to the membrane structure rooted on h_a and μ_b to the one rooted on h_b.

- $P = \{\alpha_i \mid 1 \leq i \leq 18\}$, and $P_h(0) = \{\alpha_1\}$, for each $h \in H$.
- $E_{mem} = \{FlagA_{mem}, FlagB_{mem}, p \cdot q + 1\}$ and $E_{mem}(0) = \{p \cdot q + 1\}$.
- The set of variables is:
  - $Var_h = \{X_{1,h}, X_{2,h}, Y_{1,h}, Y_{2,h}, Z_{1,h}, Z_{2,h}, D_h\}$, for each $h, 1 \leq h \leq p \cdot q$.
  - $$\begin{aligned} Var_{mem} = \; & \{X_{1,mem}, X_{2,mem}, X_{3,mem}, X_{4,mem}\} \cup \\ & \{Y_{1,mem}, Y_{2,mem}, Y_{3,mem}, Y_{4,mem}\} \cup \\ & \{Z_{1,mem}, Z_{2,mem}, Z_{3,mem}, Z_{4,mem}\} \cup \\ & \{U_{1,mem}, U_{2,mem}, U_{3,mem}, U_{4,mem}\} \cup \\ & \{A_{mem}, B_{mem}, NA_{mem}, NB_{mem}, Halt_{mem}\} \cup \\ & \{A_{h,mem}, B_{h,mem} \mid 1 \leq h \leq p \cdot q\}. \end{aligned}$$
  - Initially, all variables in $Var_h (h \neq h_a \wedge h \neq h_b)$ and all variables in Var_{h_a} and Var_{h_b} different to $Y_{1,h}, Y_{2,h}$, are equal to zero. Besides, initially the tuple (Y_{1,h_a}, Y_{2,h_a}) (resp. (Y_{1,h_b}, Y_{2,h_b})) provides the position of the initial state of the robot h_a (resp. the position of the final state of the robot h_b).
  - If the value of variable $Halt_{mem}$ is equals to 1, then the computation stops.

- Next, the finite set of programs P_{r_h} and the set of rules $\mathcal{R}$ of the system are defined according to the requirements to simulate the bidirectional RRT algorithm.
- In order to synchronize the sequence of an iteration, for each $h \in H$ the protein evolution rules $[\,\alpha_i \rightarrow \alpha_{i+1}\,]_h$, for $1 \leq i \leq 17$, and $[\,\alpha_{18} \rightarrow \alpha_1\,]_h$ are considered.
- Four random numbers are generated in the shared memory:

$$\begin{cases} \textit{Production function} : F(X_{1,mem}) = \\ \textit{Random}(i, 0 \leq i < p) \\ \textit{Repartition protocol} : 1|X_{1,mem} \\ \textit{Protein} : \alpha_1 \end{cases}$$

$$\begin{cases} \textit{Production function} : F(X_{2,mem}) = \\ \textit{Random}(i, 0 \leq i < q) \\ \textit{Repartition protocol} : 1|X_{2,mem} \\ \textit{Protein} : \alpha_1 \end{cases}$$

$$\begin{cases} \textit{Production function} : F(X_{3,mem}) = \\ \textit{Random}(i, 1 \leq i < p) \\ \textit{Repartition protocol} : 1|X_{3,mem} \\ \textit{Protein} : \alpha_1 \end{cases}$$

$$\begin{cases} \textit{Production function} : F(X_{4,mem}) = \\ \textit{Random}(i, 1 \leq i < q) \\ \textit{Repartition protocol} : 1|X_{4,mem} \\ \textit{Protein} : \alpha_1 \end{cases}$$

- Each membrane $h \in \mu_a$ will read the random numbers $X_{1,mem}$, $X_{2,mem}$. Each membrane $h \in \mu_b$ will read the random numbers $X_{3,mem}$, $X_{4,mem}$.

$$\begin{cases} (h\,,\, X_{1,h} \,/\, X_{1,mem}\,,\, mem)^R_{\alpha_2} : h \in \mu_a \\ \\ (h\,,\, X_{2,h} \,/\, X_{2,mem}\,,\, mem)^R_{\alpha_2} : h \in \mu_a \end{cases}$$
$$\begin{cases} (h\,,\, X_{1,h} \,/\, X_{3,mem}\,,\, mem)^R_{\alpha_2} : h \in \mu_b \\ \\ (h\,,\, X_{2,h} \,/\, X_{4,mem}\,,\, mem)^R_{\alpha_2} : h \in \mu_b \end{cases}$$

- For each membrane $h \in \{\mu_a, \mu_b\}$, the distance D_h between its position $(Y_{1,h}, Y_{2,h})$ and the position given by the generated random numbers $(X_{1,h}, X_{2,h})$ is computed. For the remaining membranes, $D_h = p \cdot q + 1$.

$$\begin{cases} \textit{Production function} : F(X_{1,h}, X_{2,h}, Y_{1,h}, Y_{2,h}) = \\ \begin{cases} \sqrt{\sum_{j=1}^{2}(X_{j,h} - Y_{j,h})^2} & \text{if} \quad h \in \{\mu_a, \mu_b\} \\ p \cdot q + 1 & \text{if} \quad h \notin \{\mu_a, \mu_b\} \end{cases} \\ \textit{Repartition protocol} : 1|D_h \\ \textit{Protein} : \alpha_3 \end{cases}$$

- Each membrane h writes its value D_h to the shared memory.

$$\begin{cases} (h\,,\, D_h \,/\, A_{h,mem}\,,\, mem)^W_{\alpha_4} : h \in \mu_a \\ \\ (h\,,\, D_h \,/\, B_{h,mem}\,,\, mem)^W_{\alpha_4} : h \in \mu_b \end{cases}$$

- The minimum of all distances $A_{h,mem}$ is computed in the shared memory.
  - *Production function*: $F(A_{1,mem}, \ldots, A_{p \cdot q,mem}) = \min\{A_{1,mem}, \ldots, A_{p \cdot q,mem}\}$
  - *Repartition protocol*: $1|A_{mem}$
  - *Protein*: α_5

- The minimum of all distances $B_{h,mem}$ is computed in the shared memory.
  - *Production function*: $F(B_{1,mem}, \ldots, B_{p \cdot q,mem}) = \min\{B_{1,mem}, \ldots, B_{p \cdot q,mem}\}$
  - *Repartition protocol*: $1|B_{mem}$
  - *Protein*: α_5

- Variable (enzyme) $FlagA_{mem}$ is set to zero if $A_{mem} \leq Threshold.$
  - *Production function*: $F(A_{mem}) = \begin{cases} 0 & \text{if } A_{mem} \leq Threshold \\ p \cdot q + 1 & \text{otherwise} \end{cases}$
  - *Repartition protocol*: $1|FlagA_{mem}$
  - *Protein*: α_6

- Variable (enzyme) $FlagB_{mem}$ is set to zero if $B_{mem} \leq Threshold.$

- *Production function*: $F(B_{,mem}) = \begin{cases} 0 & \text{if } B_{mem} \leq Threshold \\ p \cdot q + 1 & \text{otherwise} \end{cases}$
- *Repartition protocol*: $1|FlagB_{mem}$
- *Protein*: α_6

- The label $near_a$, corresponding to the closer membrane to the randomly generated point for μ_a, is obtained.
  - *Production function*: $F(A_{1,mem}, \ldots, A_{p \cdot q,mem}) = \text{arg-min}\{A_{1,mem}, \ldots, A_{p \cdot q,mem}\}$
  - *Repartition protocol*: $1|NA_{mem}$
  - *Protein*: α_7
  - *Enzyme*: $FlagA_{mem}$

- The label $near_b$, corresponding to the closer membrane to the randomly generated position for μ_b, is obtained.
  - *Production function*: $F(B_{1,mem}, \ldots, B_{p \cdot q,mem}) = \text{arg-min}\{B_{1,mem}, \ldots, B_{p \cdot q,mem}\}$
  - *Repartition protocol*: $1|NB_{mem}$
  - *Protein*: α_7
  - *Enzyme*: $FlagB_{mem}$

- The position of membrane $near_a$ is computed.

$$\begin{cases} Production\ function : F(NA_{mem}) = rm(NA_{mem} - 1, p) \\ Repartition\ protocol : 1|Y_{1,mem} \\ Protein : \alpha_8 \\ Enzyme : FlagA_{mem} \end{cases}$$

$$\begin{cases} Production\ function : F(NA_{mem}) = qt(NA_{mem} - 1, p) \\ Repartition\ protocol : 1|Y_{2,mem} \\ Protein : \alpha_8 \\ Enzyme : FlagA_{mem} \end{cases}$$

- The position of membrane $near_b$ is computed.

$$\begin{cases} Production\ function : F(NB_{mem}) = rm(NB_{mem} - 1, p) \\ Repartition\ protocol : 1|Y_{3,mem} \\ Protein : \alpha_8 \\ Enzyme : FlagB_{mem} \end{cases}$$

$$\begin{cases} Production\ function : F(NB_{mem}) = qt(NB_{mem} - 1, p) \\ Repartition\ protocol : 1|Y_{4,mem} \\ Protein : \alpha_8 \\ Enzyme : FlagB_{mem} \end{cases}$$

- The unitary vectors are created in the shared memory.

$$\begin{cases} \textit{Production function}: \\ F(X_{1,mem}, X_{2,mem}, Y_{1,mem}, Y_{2,mem}) = \\ \frac{X_{1,mem}-Y_{1,mem}}{\sqrt{\sum_{j=1}^{2}(X_{j,mem}-Y_{j,mem})^2}} \\ \textit{Repartition protocol}: 1|U_{1,mem} \\ \textit{Protein}: \alpha_9 \\ \textit{Enzyme}: FlagA_{mem} \end{cases}$$

$$\begin{cases} \textit{Production function}: \\ F(X_{1,mem}, X_{2,mem}, Y_{1,mem}, Y_{2,mem}) = \\ \frac{X_{2,mem}-Y_{2,mem}}{\sqrt{\sum_{j=1}^{2}(X_{j,mem}-Y_{j,mem})^2}} \\ \textit{Repartition protocol}: 1|U_{2,mem} \\ \textit{Protein}: \alpha_9 \\ \textit{Enzyme}: FlagA_{mem} \end{cases}$$

$$\begin{cases} \textit{Production function}: \\ F(X_{3,mem}, X_{4,mem}, Y_{3,mem}, Y_{4,mem}) = \\ \frac{X_{3,mem}-Y_{3,mem}}{\sqrt{\sum_{j=3}^{4}(X_{j,mem}-Y_{j,mem})^2}} \\ \textit{Repartition protocol}: 1|U_{3,mem} \\ \textit{Protein}: \alpha_9 \\ \textit{Enzyme}: FlagB_{mem} \end{cases}$$

$$\begin{cases} \textit{Production function}: \\ F(X_{3,mem}, X_{4,mem}, Y_{3,mem}, Y_{4,mem}) = \\ \frac{X_{4,mem}-Y_{4,mem}}{\sqrt{\sum_{j=3}^{4}(X_{j,mem}-Y_{j,mem})^2}} \\ \textit{Repartition protocol}: 1|U_{4,mem} \\ \textit{Protein}: \alpha_9 \\ \textit{Enzyme}: FlagB_{mem} \end{cases}$$

- Variable (enzyme) $FlagA_{mem}$ is set to zero if there is collision for μ_a.
  - *Production function*: $F(Y_{1,mem}, Y_{2,mem}, U_{1,mem}, U_{2,mem}) =$
    $\begin{cases} 0 & \text{if } COLLISION(Y_{1,mem}, Y_{2,mem}, \\ U_{1,mem}, U_{2,mem}) & \\ p \cdot q + 1 & \text{otherwise} \end{cases}$
  - *Repartition protocol*: $1|FlagA_{mem}$
  - *Protein*: α_{10}
  - $COLLISION$ is a function returning *true* if there are static obstacles in a linear trajectory starting at $(Y_{1,mem}, Y_{2,mem})$ and applying a motion $(U_{1,mem}, U_{2,mem})$ for Δt time.

- Variable (enzyme) $FlagB_{mem}$ is set to zero if there is collision for μ_b.
  - *Production function*: $F(Y_{3,mem}, Y_{4,mem}, U_{3,mem}, U_{4,mem}) =$
    $\begin{cases} 0 & \text{if } COLLISION(Y_{3,mem}, Y_{4,mem}, \\ U_{3,mem}, U_{4,mem}) & \\ p \cdot q + 1 & \text{otherwise} \end{cases}$

- *Repartition protocol*: $1|FlagB_{mem}$
- *Protein*: α_{10}
- $COLLISION$ is a function returning *true* if there are static obstacles in a linear trajectory starting at $(Y_{3,mem}, Y_{4,mem})$ and applying a motion $(U_{3,mem}, U_{4,mem})$ for Δt time.

- Positions of new membranes are computed in the shared memory.

$$\begin{cases} \textit{Production function}: \\ F(Y_{1,mem}, U_{1,mem}) = round(Y_{1,mem} + U_{1,mem} \cdot \Delta t) \\ \textit{Repartition protocol}: 1|Z_{1,mem} \\ \textit{Protein}: \alpha_{11} \\ \textit{Enzyme}: FlagA_{mem} \end{cases}$$

$$\begin{cases} \textit{Production function}: \\ F(Y_{2,mem}, U_{2,mem}) = round(Y_{2,mem} + U_{2,mem} \cdot \Delta t) \\ \textit{Repartition protocol}: 1|Z_{2,mem} \\ \textit{Protein}: \alpha_{11} \\ \textit{Enzyme}: FlagA_{mem} \end{cases}$$

$$\begin{cases} \textit{Production function}: \\ F(Y_{3,mem}, U_{3,mem}) = round(Y_{3,mem} + U_{3,mem} \cdot \Delta t) \\ \textit{Repartition protocol}: 1|Z_{3,mem} \\ \textit{Protein}: \alpha_{11} \\ \textit{Enzyme}: FlagB_{mem} \end{cases}$$

$$\begin{cases} \textit{Production function}: \\ F(Y_{4,mem}, U_{4,mem}) = round(Y_{4,mem} + U_{4,mem} \cdot \Delta t) \\ \textit{Repartition protocol}: 1|Z_{4,mem} \\ \textit{Protein}: \alpha_{11} \\ \textit{Enzyme}: FlagB_{mem} \end{cases}$$

- The membranes labelled by NA_{mem} and NB_{mem} will read the positions corresponding to the new membranes from the shared memory.

$$\begin{cases} (NA_{mem}\,,\, Z_{1,NA_{mem}}\,/\,Z_{1,mem}\,,\, mem)^{R}_{\alpha_{12}} \\ \\ (NA_{mem}\,,\, Z_{2,NA_{mem}}\,/\,Z_{2,mem}\,,\, mem)^{R}_{\alpha_{12}} \end{cases}$$

$$\begin{cases} (NB_{mem}\,,\, Z_{1,NB_{mem}}\,/\,Z_{3,mem}\,,\, mem)^{R}_{\alpha_{12}} \\ \\ (NB_{mem}\,,\, Z_{2,NB_{mem}}\,/\,Z_{4,mem}\,,\, mem)^{R}_{\alpha_{12}} \end{cases}$$

- A child membrane with position $(Z_{1,NA_{mem}}, Z_{2,NA_{mem}})$ is created in μ_a.

$$\left[\ \left[\ \begin{matrix} X_{1,h} & X_{2,h} \\ Y_{1,h} & Y_{2,h} \\ Z_{1,h} & Z_{2,h} \\ D_h & \end{matrix}\ \right]_h\ \right]_{NA_{mem}}$$

Being $h = Z_{2,NA_{mem}} \cdot p + Z_{1,NA_{mem}} + 1$.
This rule is mediated by protein α_{13}.

- A child membrane with position $(Z_{1,NB_{mem}}, Z_{2,NB_{mem}})$ is created in μ_b.

$$\left[\left[\begin{array}{ll} X_{1,h} & X_{2,h} \\ Y_{1,h} & Y_{2,h} \\ Z_{1,h} & Z_{2,h} \\ D_h & \end{array} \right]_h \right]_{NB_{mem}}$$

  Being $h = Z_{2,NB_{mem}} \cdot p + Z_{1,NB_{mem}} + 1$.
  This rule is mediated by protein α_{13}.

- Each membrane in μ_a reads the position of the new membrane created in μ_b

$$\begin{cases} (h\,,\, X_{1,h} \,/\, Z_{3,mem}\,,\, mem)^R_{\alpha_{14}} : h \in \mu_a \\ \\ (h\,,\, X_{2,h} \,/\, Z_{4,mem}\,,\, mem)^R_{\alpha_{14}} : h \in \mu_a \end{cases}$$

- Each membrane in μ_b reads the pose of the new membrane created in μ_a

$$\begin{cases} (h\,,\, X_{1,h} \,/\, Z_{1,mem}\,,\, mem)^R_{\alpha_{14}} : h \in \mu_b \\ \\ (h\,,\, X_{2,h} \,/\, Z_{2,mem}\,,\, mem)^R_{\alpha_{14}} : h \in \mu_b \end{cases}$$

- For each membrane $h \in \{\mu_a, \mu_b\}$, the distance D_h between its position $(Y_{1,h}, Y_{2,h})$ and the position given by the new membrane in the other membrane structure is computed. For the remaining membranes, $D_h = p \cdot q + 1$.

$$\begin{cases} \textit{Production function} : F(X_{1,h}, X_{2,h}, Y_{1,h}, Y_{2,h}) = \\ \begin{cases} \sqrt{\sum_{j=1}^{2} (X_{j,h} - Y_{j,h})^2} & \text{if} \quad h \in \{\mu_a, \mu_b\} \\ p \cdot q + 1 & \text{if} \quad h \notin \{\mu_a, \mu_b\} \end{cases} \\ \textit{Repartition protocol} : 1|D_h \\ \textit{Protein} : \alpha_{15} \end{cases}$$

- Each membrane h writes its value D_h to the shared memory.

$$\begin{cases} (h\,,\, D_h \,/\, A_{h,mem}\,,\, mem)^W_{\alpha_{16}} : h \in \mu_a \\ \\ (h\,,\, D_h \,/\, B_{h,mem}\,,\, mem)^W_{\alpha_{16}} : h \in \mu_b \end{cases}$$

- The minimum of all distances $A_{h,mem}$ is computed in the shared memory.
  - *Production function*: $F(A_{1,mem}, \dots, A_{p\cdot q,mem}) = \min\{A_{1,mem}, \dots, A_{p\cdot q,mem}\}$
  - *Repartition protocol*: $1|A_{mem}$
  - *Protein*: α_{17}

- The minimum of all distances $B_{h,mem}$ is computed in the shared memory.
  - *Production function*: $F(B_{1,mem}, \dots, B_{p\cdot q,mem}) = \min\{B_{1,mem}, \dots, B_{p\cdot q,mem}\}$

  - *Repartition protocol*: $1|B_{mem}$
  - *Protein*: α_{17}

- If $A_{mem} \leq d_{min}$ or $B_{mem} \leq d_{min}$ then the RRTs have been connected and the computation must halt.

  - *Production function*: $F(A_{mem}, B_{mem}) = \begin{cases} 1 & \text{if } A_{mem} \leq Threshold \vee B_{mem} \leq Threshold \\ 0 & \text{otherwise} \end{cases}$
  - *Repartition protocol*: $1|Halt_{mem}$
  - *Protein*: α_{18}

6 A simulator based on C++ and ROS

A C++ simulator has been developed within the ROS [30] framework. It can be downloaded from `https://github.com/RGNC/renpsm`. The experiments have been conducted by using a dual-wheeled nonholonomic robot (the Pioneer 3-DX) in two virtual environments. The software is composed by three modules:

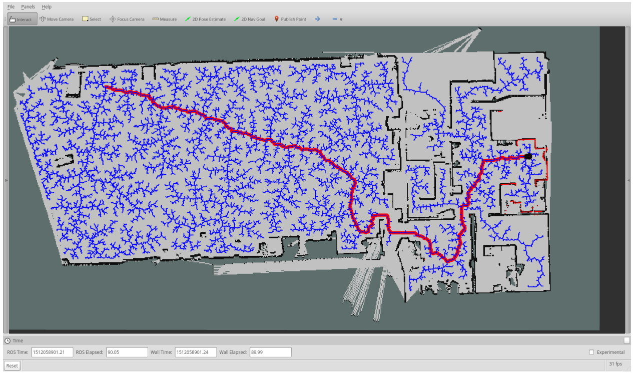
- MobileSim module [31]: It receives the static information about the map, as well as motion commands (v, ω) and generates the wheels odometry and information related to sensors (laser rangefinder for obstacle detection). It moves the simulated robot in the virtual environment.
- RENPSM module: It receives the information about the map, as well as the information about odometry and sensors and the goal of the robot. It computes a bidirectional RRT by using a RENPSM simulator and finally it sends a sequence of motion commands to the MobileSim module.
- RVIZ module [32]: This module is used for visualization. It receives the static information about the map, as well as all the information generated by the MobileSim module and several visual markers generated by the RENPSM module. It shows to the user all the information in real-time by using a 3D representation of the environment and the robot.

We have used two virtual environments, Figures 4 and 5 show the corresponding RVIZ visualization for each one before starting the robot motion, i.e, after generating the bidirection RRT by using the RENPSM module. Figure 6 is a second simulation for environment 2.

The first environment has been used for experimental validation of the RENPSM model by generating several simulations and comparing the resulting RRT visualizations with the ones generated with a conventional RRT software.

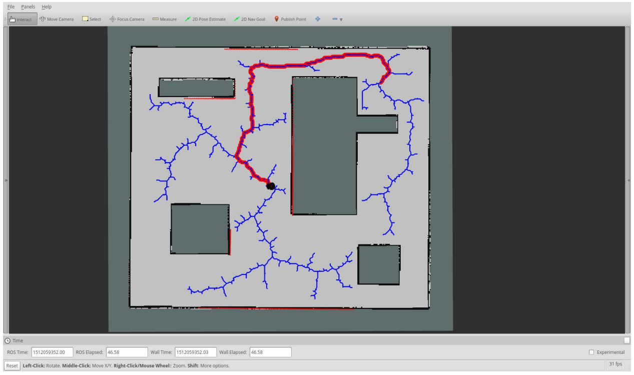
The second environment has been used for benchmarking, generating 1435 simulations by fixing the starting point and the goal of the robot and measuring the cost in distance of the generated path. The results are shown in Table 1.

We have measured the cost of an optimal path generated by hand (about $10m$) and, as expected, the cost of the best path generated by the bidirectional RRT is larger than the optimal cost, since the algorithm generates the first feasible path that can be found.



ROS Time: 1512058901.21 ROS Elapsed: 90.05 Wall Time: 1512058901.24 Wall Elapsed: 89.99

Figure 4: RVIZ visualization of a simulation in environment 1



ROS Time: 1512059352.00 ROS Elapsed: 46.58 Wall Time: 1512059352.03 Wall Elapsed: 46.58

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift**: More options.

Figure 5: RVIZ visualization of a simulation in environment 2

7 Generation of ad-hoc RENPSM simulators with P-Lingua

P-Lingua is a language to define membrane computing models [3, 7, 14, 33], allowing to write definitions in a friendly way, as its syntax is close to standard scientific notation. In this section, we present an extension of P-Lingua for RENPSM. A parser based on Flex and Bison [35, 36] has been implemented to generate ad-hoc simulators for the defined models by using automatic programming. The source code of the simulators is generated in C language and OpenMP [37], providing multi-threading simulators for multicore processors. Finally, some experimental results are presented.

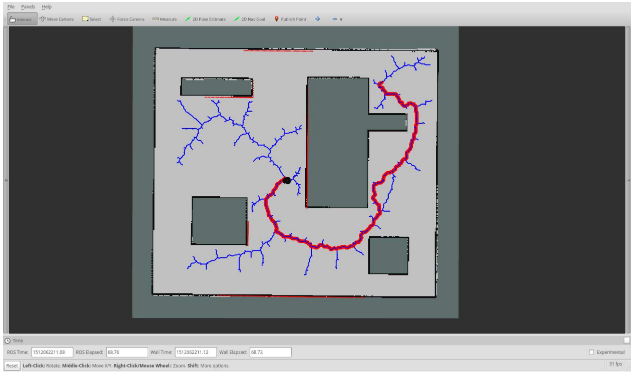
7.1 Extension of P-Lingua for RENPSM

New syntactic ingredients have been included to define models based RENPSM:

- The first line of a P-Lingua file defining a RENPSM model must be `@model<renpsm>`

Table 1: Benchmarking results

Min. cost	11.77 m
Max. cost	17.96 m
Average cost	13.42 m
Standard deviation	0.795 m
Experiments	1435



ROS Time: 1512062211.08 ROS Elapsed: 68.76 Wall Time: 1512062211.12 Wall Elapsed: 68.73

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift**: More options.

Figure 6: A second simulation in environment 2

- The initial membrane structure is defined as a two level cell-like membrane structure:

  $$@\texttt{mu} = [[]'\texttt{h1}[]'\texttt{h2}...[]'\texttt{hN}]'\texttt{v}$$

  where `h1, h2,...,hN` and `v` are numeric labels in $\mathbb{N}$. The label 0 is used the shared memory. P-Lingua variables can optionally be used instead of literals.

- Each RENPSM variable includes at least one index. The last index of each variable is the label of the compartment containing the variable. For instance, the variable $X_{1,mem}$ is written `X{1,0}`. For the sake of simplicity, indexes will be omitted in the rest of this section.

- Proteins are written as common objects in P-Lingua.

- Production rules are written in the next manner:

  $$\texttt{X} < -\texttt{func}(\texttt{param1}, \texttt{param2}, ..., \texttt{paramN}), \texttt{protein}?\texttt{enzyme}$$

  where `X` is a RENPSM variable; `func` is a production function with parameters $\texttt{param1}, \texttt{param2}, ..., \texttt{paramN}$, `protein` is an optional protein related to the rule and `enzyme` is an optional enzyme. There are a fixed set of production functions that can be used. In this paper, we have implemented the functions for the model in Section 5. More functions could be implemented in C language. The rule will be executed if the protein is present and the enzyme has a value greater than zero.

- Reading-only and writing-only communication rules are respectively written as follows:

  $$\texttt{X} < -\texttt{Y}, \texttt{protein}?\texttt{enzyme}$$

  $$\texttt{Y} < -\texttt{X}, \texttt{protein}?\texttt{enzyme}$$

  where `X` is a variable in the membrane structure, `Y` is a variable in the shared memory, `protein` is an optional protein related to the rule and `enzyme` is an optional enzyme. The rule will be executed if the protein is present and the enzyme has a value greater than zero.

- Membrane creation rules are written as follows:

$$[[]'\texttt{h1}]'\texttt{h2}, \texttt{protein}?\texttt{enzyme}$$

where `h1`, `h2` are labels that can be given by numeric literals, P-Lingua variables or even RENPSM variables; `protein` is an optional protein related to the rule and `enzyme` is an optional enzyme. The rule will be executed if the protein is present and the enzyme has a value greater than zero.

- Protein evolution rules are written in the next manner:

$$[\texttt{p1} < -\texttt{p2}]'\texttt{h}$$

where `p1 and p2` are proteins and `h` is the label of a compartment, given by a numeric literal or a P-Lingua variable.

- We have created an special iterator `x in H` to iterate all the membrane labels in the tree structure rooted at `H` (with a depth-first order).
- Other ingredients of P-Lingua, such as modules, indexes, iterators and comments can be used as usual in this extension.

The whole P-Lingua file defining the model presented in 5 can be downloaded from `https://github.com/RGNC/renpsm_openmp/blob/master/birrt_renpsm_test1.pli`.

7.2 A parser to generate ad-hoc RENPSM simulators

A parser based on Flex and Bison [35,36] has been implemented and can be downloaded from `https://github.com/RGNC/renpsm_openmp`.

The website includes instructions for compiling and running the experiments in this paper. The parser reads a P-Lingua file and generates a command-line ad-hoc simulator tool that accepts a PGM file defining the obstacles and generates another PGM file printing the generated RRT over the obstacles image. The source code of the simulator tool is generated in C with OpenMP, the command-line syntax for the generated simulator is:

```
./simulator [-t threads] [-s steps] [-d] [-r seed]
[-m obstacles.pgm] [-o output.pgm]
```

Where:

- `-t threads` is the number of threads to be used. Default is 4. If 1 thread is set, the simulator will be sequential.
- `-s steps` is the maximum number of computational steps to simulate. The simulator stops if the variable $Halt_{mem}$ is set to 1 or the number of steps is reached. Default is 1048576 steps.
- If `-d` is set, debug information will be prompted.
- `-r seed` defines the pseudo-random number generator seed. If no seed is configured, an arbitrary seed based on the current clock time will be used.
- `-m obstacles.pgm` is the PGM file defining the obstacle grid for the collision function.
- `-o output.pgm` is the PGM file to print the output RRT over the obstacle image.

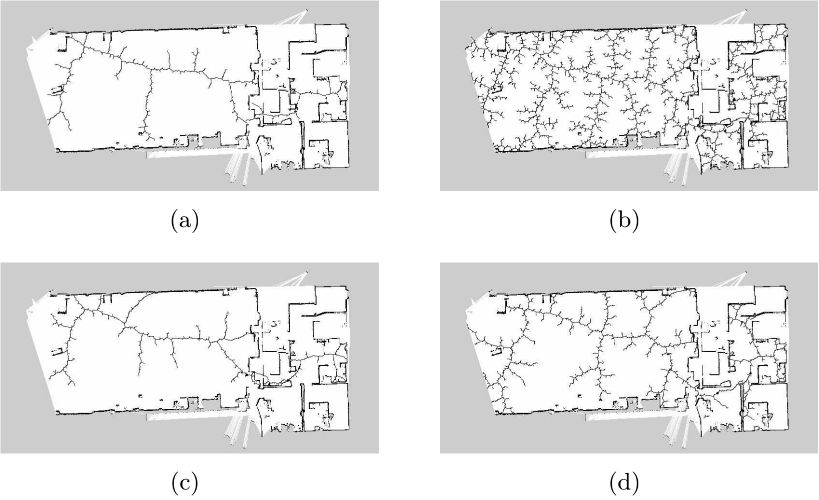


Figure 7: Example outputs for environment 1: (a) Example output 1; (b) Example output 2; (c) Example output 3; (d) Example output 4

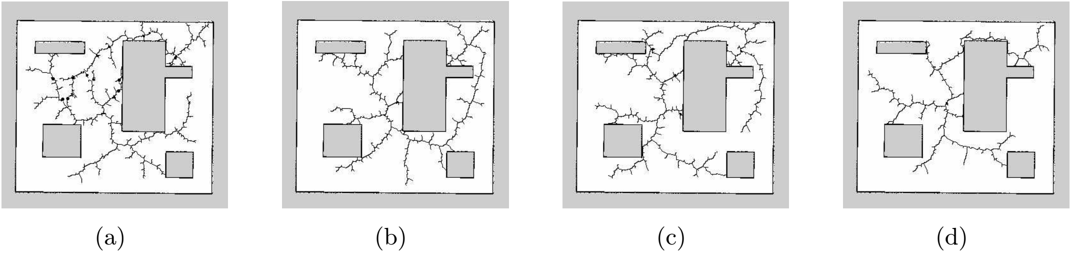


Figure 8: Example outputs for environment 2: (a) Example output 1; (b) Example output 2; (c) Example output 3; (d) Example output 4

7.3 Experimental results

We have used the same two environments presented in Section 6. The environment 1 is defined with the files: `birrt_renpsm_test2.pli`; `office.pgm`.

The environment 2 is defined with the files: `birrt_renpsm_test1.pli`; `map.pgm`.

Each environment uses a PGM file in gray scale for obstacle definition, where each pixel represents a region of 5 cm^2 in the real world. Each environment also uses a P-Lingua file with identical P system definition but different initial parameters p, q, x_0, y_0, x_1, y_1. The problem is to find a RRT representation to navigate from (x_0, y_0) to (x_1, y_1) in the environment of p pixels length and q pixels height given by the corresponding PGM file.

In Figures 7 and 8 there are four example outputs for each environment using the corresponding generated ad-hoc simulator with 8 threads and arbitrary pseudo-random number generator seeds based on the CPU clock time.

8 Conclusions

This paper deals with an algorithm belonging to a family widely used to solve the problem of motion planning in robots, e.g., the RRT algorithms. Such class of algorithms are based on the randomized exploration of the configuration space. This paper is an extension of [21]. In such a work, a variant of Enzymatic Numerical P systems, called *random enzymatic numerical*

P systems with proteins and shared memory (RENPSM, for short) was introduced. Besides, a simplified version of the standard bidirectional RRT algorithm was described by a RENPSM system capturing the semantics of the new variant, where maximal parallelism is used.

The main contribution of this paper with respect to [21] is to provide a novel approximation for software simulation by using automatic programming. We have implemented a tool for parsing P-Lingua files defining RENPSM models and generating source code in C and OpenMP for ad-hoc simulators. Thus, we have a flexible way to debug since we are using a language to define the models instead of hard-coding them in the source code. Moreover, the generated source code is able to run on multicore processors by using OpenMP.

Three main challenges are planned as future work. First, to provide a formal verification of such RENPSM systems, in the sense that they in fact simulate the RRT generation algorithm. The second challenge is to move to the RRT* algorithm [9], a variant of the initial algorithm that is able to approximate optimal motion planning with enough iterations. Finally, to provide real-life robot path planning experiments, by using a nonholonomic robot with kinodynamic and environment constraints.

We also propose to apply the simulation techniques introduced in this paper to other types of P systems walking towards a more generic software tool based on P-Lingua and automatic programming for generation of optimized ad-hoc simulators.

Acknowledgements

This work was supported by National Natural Science Foundation of China (61672437 and 61702428) and by Sichuan Science and Technology Program (2018GZ0086, 2018GZ0185).

Authors from the University of Seville also acknowledge the support of the research project TIN2017-89842-P, co-financed by *Ministerio de Economía, Industria y Competitividad (MINECO)* of Spain, through the *Agencia Estatal de Investigación (AEI)*, and by *Fondo Europeo de Desarrollo Regional (FEDER)* of the European Union.

Bibliography

[1] Astrom, K.J.; Hagglund, T. (1995). *PID Controllers: Theory, Design, and Tuning*, 1995

[2] Colomer, M.A.; Margalida, A.; D. Sanuy,D.; Pérez-Jiménez, M.J. (2011). A bio-inspired computing model as a new tool for modeling ecosystems: The avian scavengers as a case study. *Ecological Modelling*, 222 (1), 33-47, 2011.

[3] Díaz-Pernil, D.; Pérez-Hurtado, I.; Pérez-Jiménez, M.J. ; Riscos-Núñez, A. (2009). A P-Lingua Programming Environment for Membrane Computing. *Lecture Notes in Computer Science*, 5391, 187–203, 2009.

[4] Coulter, C. (1992). *Implementation of the Pure Pursuit Path Tracking Algorithm*, Tech. Report, CMU-RI-TR-92-01, Robotics Institute, Carnegie Mellon University, 1992.

[5] Fox, D.; Burgard, W.; Thrun, S. (1997). The dynamic window approach to collision avoidance, *Robotics and Automation Magazine*, 4 (1), 23-33, 1997.

[6] Gao, Y.; Wu, X.; Liu, Y.; Li, J.M.; Liu J.H.(2017). A Rapid Recognition of Impassable Terrain for Mobile Robots with Low Cost Range Finder Based on Hypotheses Testing Theory, *International Journal of Computers Communications & Control*, 12(6), 813-823, 2017.

[7] García-Quismondo, M.; Gutiérrez-Escudero, R.; Martínez-del-Amor, M.A.; Orejuela-Pinedo, E.; Pérez-Hurtado, I. (2009). P-Lingua 2.0: A software framework for cell-like P systems. *International Journal of Computers Communications* & *Control*, 4(3), 234-243, 2009.

[8] Huang, S.; Dissanayake, G. (2016). *Robot Localization: An Introduction*, Wiley Encyclopedia of Electrical and Electronics Engineering, 2016

[9] Karaman, S.; Frazzoli, E. (2010). Incremental Sampling-based Algorithms for Optimal Motion Planning, *Robotics Science and Systems VI*, 1-9, 2010

[10] Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots, *Int J Robot Res*, 5(1), 90-98, 1986.

[11] Latombe, J.C. (1991). *Robot Motion Planning*, Kluwer Academic Publishers, Boston, MA, 1991.

[12] LaValle, S.M. (1998). *Rapidly-Exploring Random Trees: A New Tool for Path Planning*, Computer Science Dept., Iowa State University, October 1998.

[13] LaValle, S.M. ; Kuffner, J.J. (1999). Randomized kinodynamic planning, *Proceedings IEEE International Conference on Robotics and Automation*, 473-479, 1999.

[14] Martínez-del-Amor, M.A.; Pérez-Hurtado, I.; Pérez-Jiménez, M.J.; Riscos-Núñez, A. (2010). A P-Lingua based simulator for Tissue P Systems, *Journal of Logic and Algebraic Programming*, 79 (6), 374-382, 2010.

[15] Nash, A.; K. Daniel, Koenig,S.; Felner, A. (2010). Theta: Any-Angle Path Planning on Grids, *Journal of Artificial Intelligence Research*, 39, 533-579, 2010.

[16] Păun, G. (2010). Computing with membranes, *Journal of Computer and System Sciences*, 61 (1), 108-143, 2000.

[17] Păun, G., Păun R. (2006). Membrane Computing and Economics: Numerical P Systems, *Fundamenta Informaticae*, 73(1,2), 213–227, 2006.

[18] Pavel, A.; Arsene, O.; Buiu, C. (2010). Enzymatic Numerical P Systems - A New Class of Membrane Computing Systems, *Proceedings of IEEE fifth international conferenced on bio-inspired computing: Theories and applications (BIC-TA)*, 1331-1336, 2010.

[19] Pavel, A.; Vasile, C.; Dumitrache, I. (2012). Robot localization implemented with enzymatic numerical P systems, *Proceedings of the international conference on biomimetic and biohybrid systems*, 204-215, 2012.

[20] Pavel, A.; Buiu, C. (2012). Using enzymatic numerical P systems for modeling mobile robot controllers, *Natural Computing*, 11(3), 387-393, 2012.

[21] Pérez, I.; Pérez-Jiménez, M.J.; Zhang, G.; Orellana-Martín D. (2018). Robot path planning using rapidly-exporing random trees: A membrane computing approach, *2018 IEEE 7th International Conference on Computers Communications and Control, Proc. of* (ICCCC2018), Oradea, Romania, May 08-12, 37-46, 2018.

[22] Pérez-Jiménez, M.J.(2014); The P versus NP problem from Membrane Computing view, *European Review*, 22 (1), 18–33, 2014.

[23] Pérez-Hurtado, I. Pérez-Jiménez, M.J. (2017). Generation of rapidly-exploring random tree by using a new class of membrane systems. *Pre-proceedings of Asian Conference on Membrane Computing* (ACMC2017), Chengdu, China, September 21-25, 534-546, 2017.

[24] Romero-Campero, F.J.; Pérez-Jiménez, M.J. (2008). A Model of the Quorum Sensing System in Vibrio fischeri Using P Systems. *Artificial Life*, 14 (1), 95-109, 2008.

[25] Stentz, A. (1995). The Focussed D* Algorithm for Real-time Replanning, *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 2, 1652-1659, 1995.

[26] Wang, H.; Yu, Y.; Q. Yuan, Q. (2011). Application of Dijkstra algorithm in robot path-planning, *Proceedings of the 2nd International Conference on Mechanic Automation and Control Engineering*, 1067-1069, 2011.

[27] Wang, T.; Zhang, G.; Zhao, J.; He, Z.; Wang, J.; Pérez-Jiménez, M.J. (2015). Fault diagnosis of electric power systems based on fuzzy reasoning spiking neural P systems. *IEEE Transactions on Power Systems*, 30(3), 1182 - 1194, 2015.

[28] Widyotriatmo, A.; Joelianto, E.; Prasdianto, A.; Bahtiar, H.; Nazaruddin, Y.Y. (2017). Implementation of Leader-Follower Formation Control of a Team of Nonholonomic Mobile Robots, *International Journal of Computers Communications & Control*, 12(6), 871-885, 2017.

[29] Zhang, G.; Perez-Jimenez, M.J.; Gheorghe, M. (2017). *Real-life Applications with Membrane Computing*, Series: Emergence, Complexity and Computation, Volume 25. Springer International Publishing, 2017.

[30] `http://www.ros.org`

[31] `http://www.mobilerobots.com/Software/MobileSim.aspx`

[32] `http://wiki.ros.org/rviz`

[33] `http://www.p-lingua.org/wiki/index.php/Main_Page`

[34] `https://developer.nvidia.com/cuda-zone`

[35] `https://github.com/westes/flexl`

[36] `https://www.gnu.org/software/bison/`

[37] `https://www.openmp.org/`

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(6), 1032-1046, December 2018.

A New Hybrid Method in Global Dynamic Path Planning of Mobile Robot

X.R. Song, S. Gao, C.B. Chen, K. Cao, J.R. Huang

Xiaoru Song*, Song Gao, Kai Cao, Jiaoru Huang
School of Electronic and Information Engineering,
Xi'an Technological University
No.2 of Xuefu Middle Road, Xi'an, Shaanxi, China
*Corresponding author: masha0422@163.com
545313193@qq.com, 602507619@qq.com, 674330497@qq.com

Abstract: Path planning and real-time obstacle avoidance is the key technologies of mobile robot intelligence. But the efficiency of the global path planning is not very high. It is not easy to avoid obstacles in real time. Aiming at these shortcomings it is proposed that a global dynamic path planning method based on improved A* algorithm and dynamic window method. At first the improved A* algorithm is put forward based on the traditional A* algorithm in the paper. Its optimized heuristic search function is designed. They can be eliminated that the redundant path points and unnecessary turning points. Simulation experiment 1 results show that the planned path length is reduced greatly. And the path transition points are less, too. And then it is focused on the global dynamic path planning of fusion improved A* Algorithm and Dynamic Window Method. The evaluation function is constructed taking into account the global optimal path. The real time dynamic path is planning. On the basis of ensuring the optimal global optimization of the planning path, it is improved that the smoothness of the planning path and the local real-time obstacle avoidance ability. The simulation experiments results show that the fusion algorithm is not only the shorter length, but also the smoother path compared the traditional path planning algorithms with the fusion algorithm in the paper. It is more fit to the dynamics of the robot control. And when a dynamic obstacle is added, the new path can be gained. The barrier can be bypass and the robot is to reach the target point. It can be guaranteed the global optimality of the path. Finally the Turtlebot mobile robot was used to experiment. The experimental results show that the global optimality of the proposed path can be guaranteed by the fusion algorithm. And the planned global path is smoother. When the random dynamic obstacle occurs in the experiment, the robot can be real-time dynamic obstacle avoidance. It can re-plan the path. It can bypass the random obstacle to reach the original target point. The outputting control parameters are more conducive to the robot's automatic control. The fusion method is used for global dynamic path planning of mobile robots in this paper. In summary the experimental results show that the method is good efficiency and real-time performance. It has great reference value for the dynamic path planning application of mobile robot.

Keywords: Mobile robot, path planning, DWM algorithm.

1 Introduction

Path planning is one of the most basic and important issues of mobile robots [2]. It is one of the independent acts to complete the task. The mobile robot path planning is a more difficult to solve the problem especially in the dynamic environment. It requires the robot process as far as possible not to deviate from the specified path, unless the obstacles encountered when the movement had to bypass the walk. But after avoiding the obstacle, the mobile robot must return to the original path until it reaches the destination [3, 8, 15, 18, 21].

Aim at this problem the dynamic window method is used in this paper. It has a good obstacle avoidance capability in the dynamic environment [6]. But this method does not satisfy the global optimal path planning. The node-based A* algorithm is suitable for global path planning. It has the advantages of simple data, small calculation compared to the biological inspired GA, PSO, etc [7] [9]. But when the environmental space increases, it needs large storage space, low efficiency, and poor real-time decision-making [11]. So the path curvature of the traditional A* algorithm is discontinuous. It will cause the movement parameters jump at the turning point. These are not conducive to the robot control problems [1]. Aim at these problems the algorithm is improved. It makes the path smoother on the basis of the overall optimization of the planning path [12]. so a global dynamic path planning method is proposed that it is combining improved A* algorithm and dynamic window method in the paper. It is proposed the evaluation function taking into account the global optimal path. And the real-time path planning is carried out by using dynamic window method based on the evaluation function. It would make the path smoother on the basis of ensuring the overall optimization of the planning path. At the same time it improves the local obstacle avoidance.

2 The construction of advanced A* algorithm in mobile robot global path planning

The traditional A* algorithm is a kind of global path planning algorithm [9]. It is a typical heuristic search algorithm in artificial intelligence. Aim at solving the problem of a large number of invalid search paths often generated in path planning [24]; the heuristic function is used to improve the search efficiency.

A* algorithm design core is its evaluation function:

$$f(n) = g(n) + h(n) \tag{1}$$

Where,$f(n)$ is the evaluation value of each node. It may be searched in the path plan [14]. The cost evaluation value of the starting point goes through the node n to the target point. $g(n)$ is the actual evaluation value from the starting node to the current node. $h(n)$ is the evaluation value from the current node to the target node (Heuristic function). The $h(n)$ can directly affect the performance of the algorithm. So it tries to approach the actual assessment $h^*(n)$ of the path planning problem. It is needed [20]. But it is also less than $h^*(n)$.

2.1 The design of the optimization heuristic function

Assuming that the $H(n)$ is actual value of the current node to the target node. When $h(n) < H(n)$, more nodes is low efficiency, but it can search for the optimal path. When $h(n) = H(n)$, it is ideal, searching along the shortest path and being the most efficient. When $h(n) > H(n)$, less nodes, high efficiency, but it is often difficult to search for the optimal path [23].

The traditional A* algorithm uses the Manhattan distance or Euclidean distance to define the heuristic function [16], namely

$$h_M(n) = |n_x - g_x| + |n_y - g_y| \tag{2}$$

$$h_E(n) = \sqrt{(n_x - g_x)^2 + (n_y - g_y)^2} \tag{3}$$

Where, n_x, n_y, g_x, g_y is the current node and target point.

A* algorithm searches the 8-neighborhood grid of the current node each time [25]. So these two heuristic function designs are not the best choice taking into account the two-wheeled robot control. Therefore, a heuristic function that is closer to the actual value $H(n)$ is designed with Manhattan distance and Euclidean distance in the paper.

$$h(n) = \begin{cases} \sqrt{2}d_x(n) + d_y(n) - d_x(n) & d_y(n) > d_x(n) \\ \sqrt{2}d_y(n) + d_x(n) - d_y(n) & d_x(n) > d_y(n) \end{cases} \tag{4}$$

Where, $d_x(n) = |n_x - g_x| \quad d_y(n) = |n_y - g_y|$, the rest parameters are the same as in the above formulas (2) and (3).

2.2 The proposition of key point selection strategy

A* algorithm planning optimal path contains a lot of redundant and unnecessary turning points. It is not conducive to the control of the robot [25]. In order to solve this problem, a key node selection strategy is proposed in the paper.

(1) Extract key turning points.

Starting from the second node of the planning path, if the current node is on the same line as the previous node and its parent node, the previous node is redundant. The redundant point is deleted. The path would update. All the path nodes would traverse. A path sequence will eventually get that contains only the starting point, the turning point, and the end point.

(2) Delete the redundant turning point.

After extracting the turning point there may still be redundant points in the trajectory. Assuming that path point after extracting the key turning point is $\{P_k | \, k = 1, 2, L, n\}$, connect P_1P_3, if P_1P_3 do not pass the obstacle, namely the line distance P_1P_3 from the obstacle is greater than the preset threshold, then continue to connect P_1P_4, until the $P_1P_k(k = 3, 4, L, n)$ through obstacles, then connect P_{k-1} and P_1, while delete the middle redundant points, update the path; repeat the above procedure from the node P_2 until there is no redundant turning point in the path. The experimental results of the key point selection strategy are shown in Fig.1. It can be seen that the application of key selection strategy algorithm is better than traditional A* algorithm.

Fig.1 (a) is the original path sequence of the traditional A* algorithm. Fig.1 (b) is the path sequence obtained by advanced A* algorithm to extract the critical turning point.Fig.1 (c) is the final path sequence by advanced A * algorithm removing the redundancy turning point. Simulation experiment 1 results show that the planned path length is reduced greatly. The path length is from 11.455m to 10.946m. And the path transition points are less greatly.

3 The establishment of mobile robot motion model in dynamic window method (DWA)

The dynamic window method has the dynamic obstacle avoidance capability of the robot in the dynamic environment [17]. The multi-group velocity can be sampled in the velocity space (linear velocity v and angular velocity w) [26]. The trajectory of the robot is simulated at these speeds for fixed time interval.After acquiring multiple sets of trajectories the corresponding speed of the optimal trajectory is selected. It is driving the robot movement according to the evaluation index.

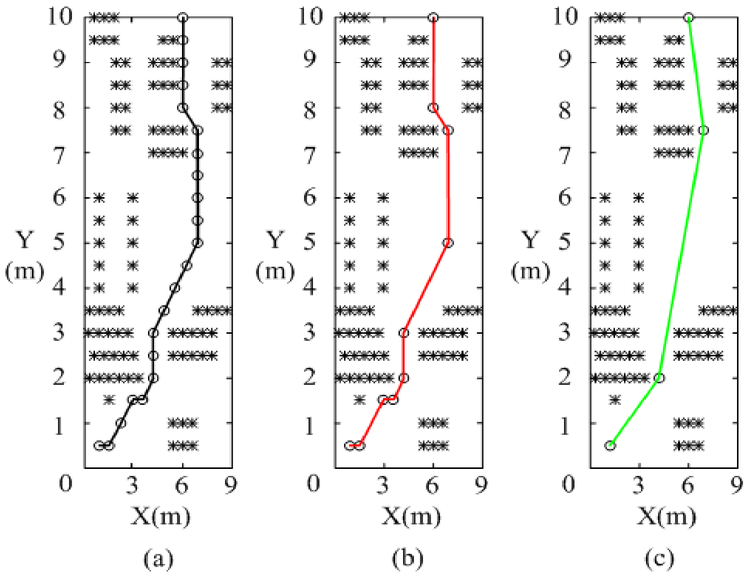


Figure 1: The experimental result of the key point selection strategy (where "-" means the planning path, "***" means the obstacle. "○" means the node.)

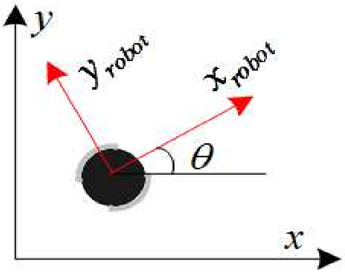


Figure 2: The robot motion diagram

3.1 The establishment of the robot movement model

In the dynamic window algorithm it is need that simulates the movement of the robot trajectory. So it is necessary that the robot movement model is known. It is assumed that the trajectory of the move robot is a circular arc. The arc trajectory is represented by (v_t, w_t). When the robot trajectory is calculated, consider two adjacent moments first. Due to the short distance of motion between adjacent moments (general chute sampling periods ms), the trajectories of the two adjacent points can be viewed as a straight line. That is, the robot moves along the axis. The motion of the robot is shown in Fig. 2.

Therefore, when the robot moves in the Omni directional direction, the trajectory motion model of the two adjacent moments $t+1$ relative to t is always as follows equation (5):

$$\begin{cases} x = & x + v_x \Delta t \cos(\theta_t) - v_y \Delta t \sin(\theta_t) \\ y = & y + v_x \Delta t \sin(\theta_t) - v_y \Delta t \cos(\theta_t) \\ \theta_t = & \theta_t + w_t \Delta t \end{cases} \tag{5}$$

Where, x and y is the robot projection distance of axis x and y. v_x and v_y is the robot moving speed of axis x and y. θ is the angle between the robot and the axis x. Δt is the two adjacent moment.

3.2 Speed sampling

Based on the above-mentioned robot's trajectory motion model, the trajectory can be deduced according to the speed. So sample a lot of speed, calculate the trajectory, and evaluate the track is good or not is only needed [25]. How to sample speed is also a core of DWA. In the two-dimensional speed space (v, w) there exist endless groups of speed. According to the limitations of the robot itself and the environment, the robot sampling speed is controlled in a certain dynamic range.

The speed limit of mobile robot between its own maximum and minimum speed:

$$V_m = \{v \in [v_{\min}, v_{\max}], w \in [w_{\min}, w_{\max}]\} \tag{6}$$

The impact of mobile robot performance by the motor: There is a maximum acceleration and deceleration limit due to the limited motor torque. So there is a dynamic window in the period of sim-period of the mobile robot trajectory forward simulation. The speed is the robot can actually reach:

$$V_d = \{(v, w) \left| \begin{matrix} v \in [v_c - \dot{v}_b \Delta t, v_c + \dot{v}_a \Delta t] \wedge \\ w \in [w_c - \dot{w}_b \Delta t, w_c + \dot{w}_a \Delta t] \end{matrix} \right. \tag{7}$$

Where v_c and w_c are the current speed of the robot. The other speeds are corresponding to the maximum acceleration and the maximum deceleration.

(3) About mobile robot security considerations

In order to allow the robot to stop before the obstacles encountered, so speed range is needed under the maximum deceleration:

$$V_a = \{(v, w) \Big| v <= \sqrt{2 \cdot dist(v, w) \cdot \dot{v}_b} \wedge \sqrt{2 \cdot dist(v, w) \cdot \dot{w}_b}\} \tag{8}$$

Where $dist(v, w)$ is the closest distance from the obstacle according to velocity (v, w).

3.3 Design the evaluation function selection

In the velocity space, there are several groups of sampling speed is feasible, so design evaluation function selects the optimal trajectory. The criteria of the design evaluation function are the robot as far as possible to avoid obstacles during local navigation, and towards the target quickly forward. The design evaluation function is:

$$G(v,w)=\sigma\begin{pmatrix}\alpha\cdot heading(v,w)+\\ \beta\cdot dist(v,w)+\\ \gamma\cdot velocity(v,w)\end{pmatrix}\tag{9}$$

Where $heading(v,w)$ is the azimuth angle evaluation function, it indicates the azimuth deviation (angular deviation) between the end direction of the simulated trajectory and the target at the current set sampling speed (v,w). $dist(v,w)$ is the closest distance of the robot itself to the obstacle at speed (v,w). If there is no obstacle on the track, it is set as a constant. $velocity(v,w)$is the evaluation of the current speed size; $\sigma(\cdot)$ is the smoothing function, α,β,γ is the weight of the three factors. The deviation is smaller, the evaluation is higher value. The distance is greater, it is the higher evaluation value. The higher speed, and the higher evaluation value. These three indicators are normalized and then weighted.

4 The proposition of advanced A* algorithm fusion DWA algorithm path planning method

Dynamic window planning path method is online real-time, with good local dynamic barrier capability. However, the requirements of global optimal path planning are missed. It is a fatal problem; it will fall into the local optimum. So the advanced A * algorithm for global path planning, fusion of dynamic window method for local obstacle avoidance, it ensures the global optimization of the dynamic planning path.

Therefore, in order to avoid the dynamic window method into the local optimal, a dynamic window evaluation function to estimate the global optimal path is proposed:

$$G(v,w)=\sigma\begin{pmatrix}\alpha\cdot \mathrm{PH}eading(v,w)+\\ \beta\cdot dist(v,w)+\\ \gamma\cdot velocity(v,w)\end{pmatrix}\tag{10}$$

Where $PHeading(v,w)$ is the azimuth deviation between the simulated trajectory end point and the current target point at the current speed (v,w). The current target point is the global optimal path sequence point; it is nearest to the current point in the robot forward direction. This evaluation function allows the local path planning to follow the global optimal path contour, thus ensuring the global optimal. This algorithm is shown in Fig. 3.

5 Algorithm simulation verification

In order to verify the validity of the fusion algorithm designed in this paper, two experiments were carried out in the MATLAB simulation environment, the grid map scene (10m × 10m, grid spacing 0.5m),and the coverage of the obstacle was 23%. In the simulation verification experiment, the starting point coordinates are (1.25 m, 0.75 m), the target point coordinates are (5.75 m, 9.75 m). Simulation experiment 2 adds dynamic obstacles based on the experiment 1 to verify the dynamic obstacle avoidance performance of this algorithm.

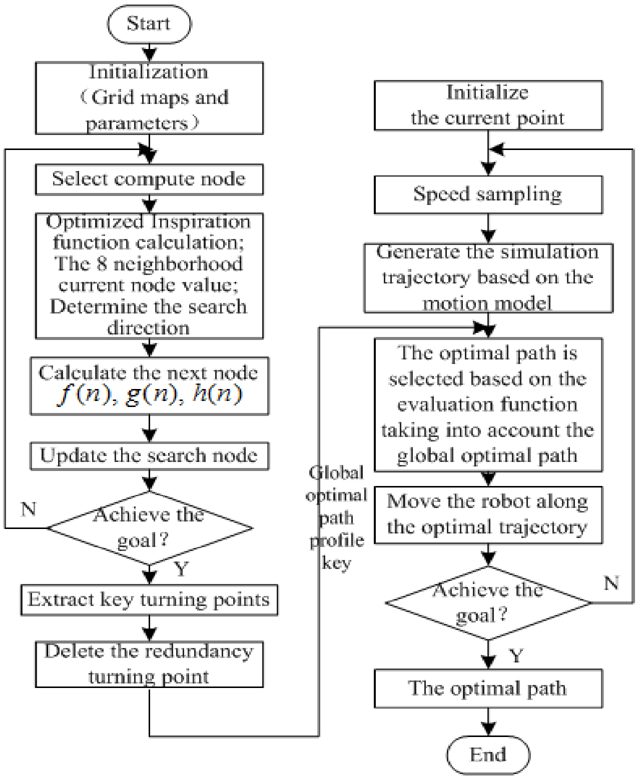

Start
Initialization (Grid maps and parameters)
Select compute node
Optimized Inspiration function calculation; The 8 neighborhood current node value; Determine the search direction
Calculate the next node $f(n)$, $g(n)$, $h(n)$
Update the search node
Achieve the goal?
N
Y
Extract key turning points
Delete the redundancy turning point
Global optimal path profile key
Initialize the current point
Speed sampling
Generate the simulation trajectory based on the motion model
The optimal path is selected based on the evaluation function taking into account the global optimal path
Move the robot along the optimal trajectory
Achieve the goal?
N
Y
The optimal path
End

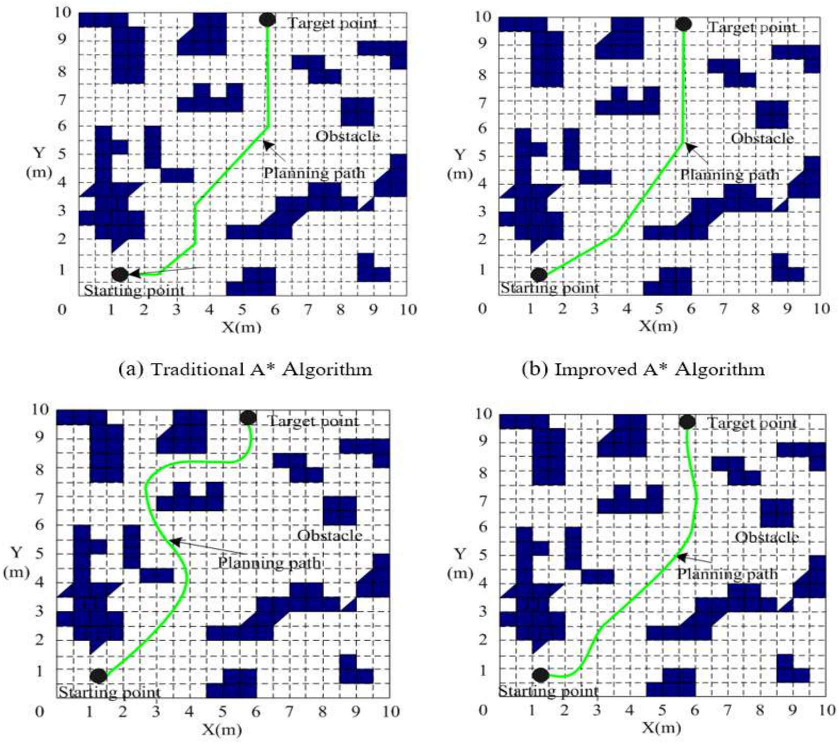
Figure 3: Fusion algorithm for mobile robot global dynamic path planning

Simulation 1: In the above mentioned simulation environment, the mobile robot uses the traditional A* algorithm, the advanced A* algorithm adding a key extraction strategy, the dynamic window algorithm and the fusion algorithm proposed in this paper are used to validate the path planning. The simulation results of the algorithm are shown in Fig.4.

In Fig.4, the blue box is the static obstacle, the black origin is the experimental mobile robot, and green solid line is the robot simulation path. It can be seen from Fig. 4 (a) that the traditional A* algorithm can plan a global optimal path. The length is 11.455m. In Fig. 4 (b), the advanced A* algorithm of key selection strategy can improve the path obviously, not only the redundant points in the traditional A* algorithm path are deleted, but the path length is better than the traditional A* algorithm. The path length is 10.946m. In Fig. 4 (c), the dynamic path method is more smooth and suitable for automatic control of the robot than the A* algorithm, but the path of the dynamic window method is non-global and its path length is about 14.439 m. In Fig. 4 (d), the fusion algorithm uses the evaluation function, the global optimal path is considered, so it can guarantee global optimization compared with the dynamic window algorithm. Compared with the traditional A* algorithm, the fusion algorithm makes the path length greatly optimized, the path length is about 11.126m.The planning path is smoother, so the curvature change of the path is continuous .These are more in line with the dynamic control of the robot.

Simulation 2: Based on the simulation experiment 1, the dynamic obstacle (5.75, 6.0) is added, the real-time dynamic obstacle avoidance simulation results of the mobile robot are shown in Fig. 5.

In Fig. 5, the red point is the artificial dynamic obstacle, the purple solid line is the simulation path of the robot, and the rest is consistent with the simulation in Figure 4.Form Fig. 5 (a) and Fig. 5 (b), it can be seen that the traditional A* algorithm and the advanced A* algorithm cannot avoid the sudden dynamic obstacle. In Fig. 5 (c), the path delimited by the dynamic window method, it can avoid the dynamic obstacle, but it is not global optimal. In Fig.5 (d), the fusion algorithm in this paper can avoid the dynamic obstacle, bypassing the obstacle to the target point, and the planned path to maintain global optimality; it achieves the expected purpose of global dynamic path planning for mobile robots.



(a) Traditional A* Algorithm

(b) Improved A* Algorithm

(c) Dynamic Window Method

(d) Fusion Algorithm in The Paper

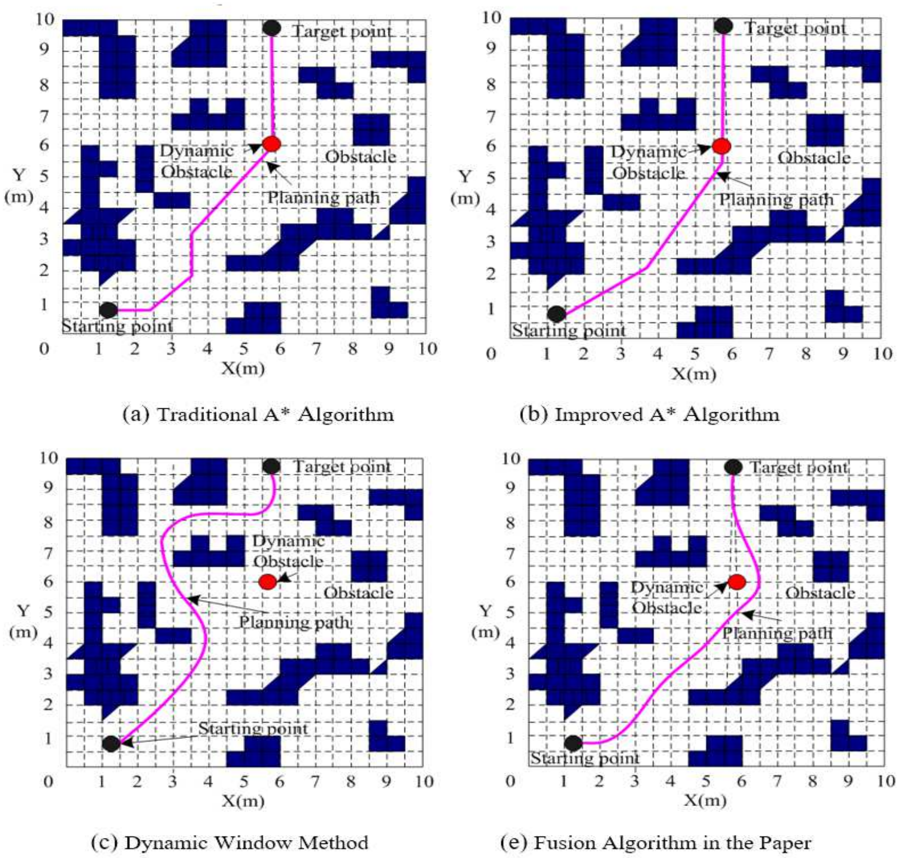
Figure 4: Various algorithmic path planning results

6 Implementation and experimental results

The experimental platform used in this article is the TurtleBot 2 two-wheel-drive robot produced by iRobot. The robot is equipped with the Kinect depth sensor developed by Microsoft as the visual sensor, Korea's Yujin Kobuki mobile base, it is shown in Fig. 6.

In this experiment, we choose our school international science and technology cooperation base in Shaanxi Province, autonomous system and intelligent control international joint research center of unmanned system laboratory. The experimental environment of this project was about $10mx10m$. The experimental environment was randomly, it is shown in Fig.7.

Robot uses Kinect sensor to obtain the actual scene of the three-dimensional point cloud data, combined with the odometer and other information through the SLAM (...) algorithm, it can obtain grid map in the interior scene. After the robot receives map of SLAM constructed, the navigation algorithm is invoked, that is, the advanced A* path planning algorithm for global path walking. In the process of obstacle avoidance, when the robot encounters the dynamic obstacle, calls the local DWA algorithm, the real-time dynamic obstacle avoidance is performed,



(a) Traditional A* Algorithm

(b) Improved A* Algorithm

(c) Dynamic Window Method

(e) Fusion Algorithm in the Paper

Figure 5: The simulation results of robot dynamic obstacle avoidance

Figure 6: The TurtleBot 2 mobile robot used in the experiment

the mobile robot global dynamic path plan is realized. Fig.8 shows a block diagram of the data processing for robot path planning.

In the experiment the environment local map in the movement is shown in Fig. 9. It is

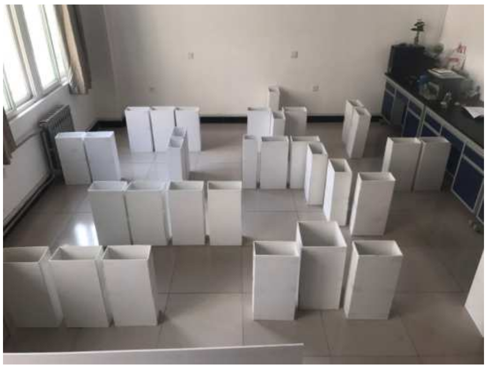


Figure 7: Laboratory to build mobile robot navigation and obstacle avoidance experimental

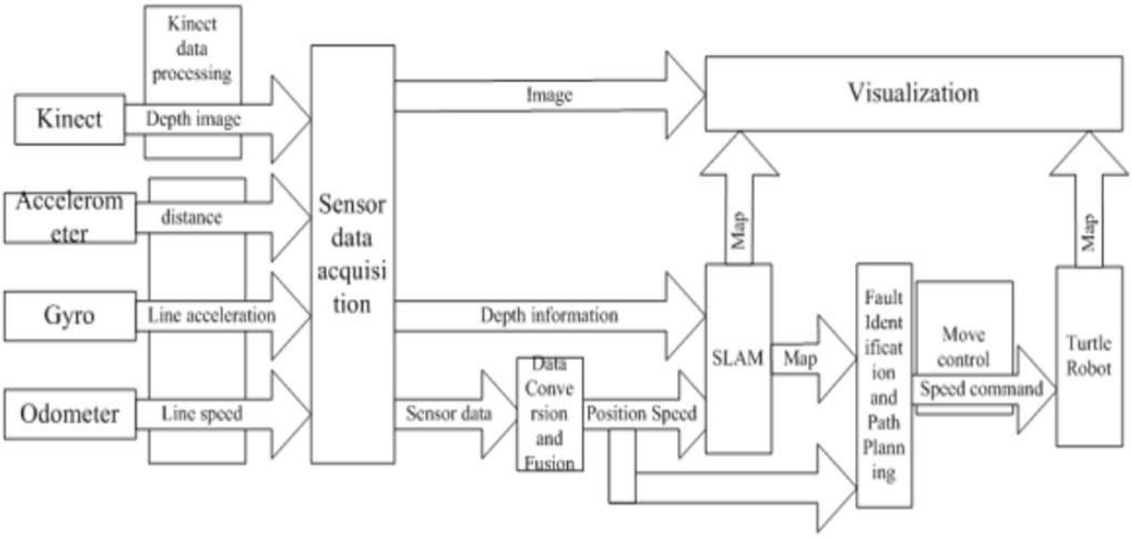


Figure 8: The block diagram of data processing for robot path planning

established by the Kinect visual sensor of the robot.

In the experiment, the initial scan was done using a speed of 20cm/s forward and rotational speed of 20cm/s, this is considered an average speed, as the robot is capable of 50cm/s forward and rotational speeds. The speed of the robot in the course of the road is appropriately slow, it ensures that the environmental data is fully analyzed and the environmental maps are established. As shown in Fig. 9, the surrounding dark area is Kinect has not yet obtained the environmental information, the blue-green area is the obstacle obtained by the expansion operation during the movement, the black part is known as an obstacle after detecting and recognizing. If a dynamic obstacle is encountered (such as a dynamic pedestrian or man-made obstacle) during the course of the process, an optimal line is re-planed and the robot smoothly bypasses the obstacle to the target point.

This experiment was carried out in unmanned system laboratory, about 10m * 10m labo-

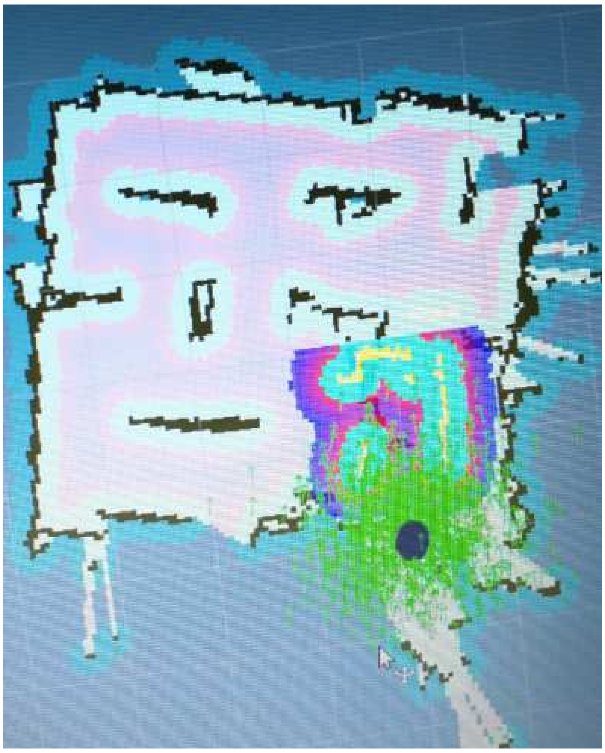


Figure 9: The map of the robot in the indoor scene

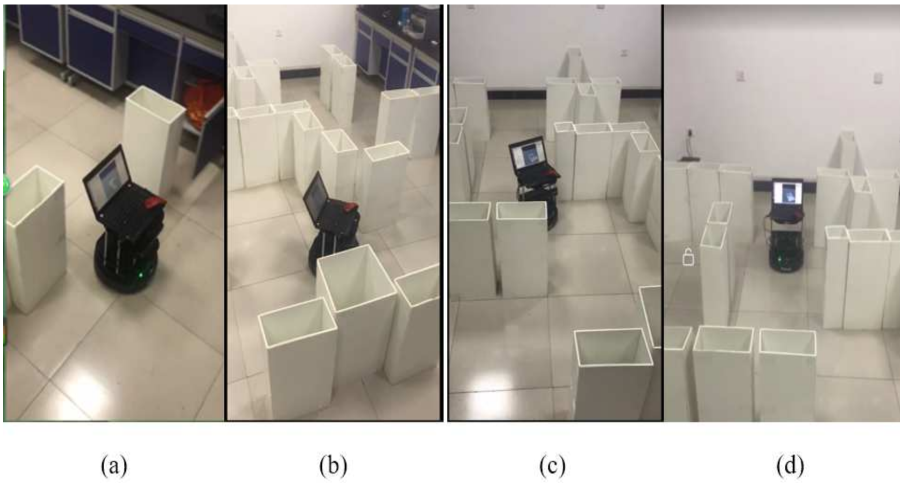


Figure 10: Indoor automatic dynamic obstacle avoidance path planning

ratory, a number of robot autonomous global navigation obstacle avoidance experiment is performed, the experimental results shown in Fig. 10 and Fig. 11. In Fig. 10, the sequence diagram of the global path planning experiment results is shown, it carried out by the mobile robot in the artificially set static obstacle experiment environment. Fig. (a) is the initial entry of the test

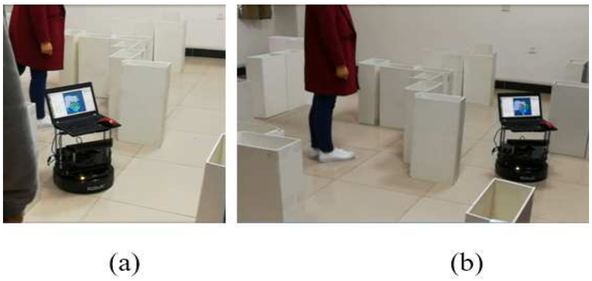


Figure 11: Dynamic obstacle

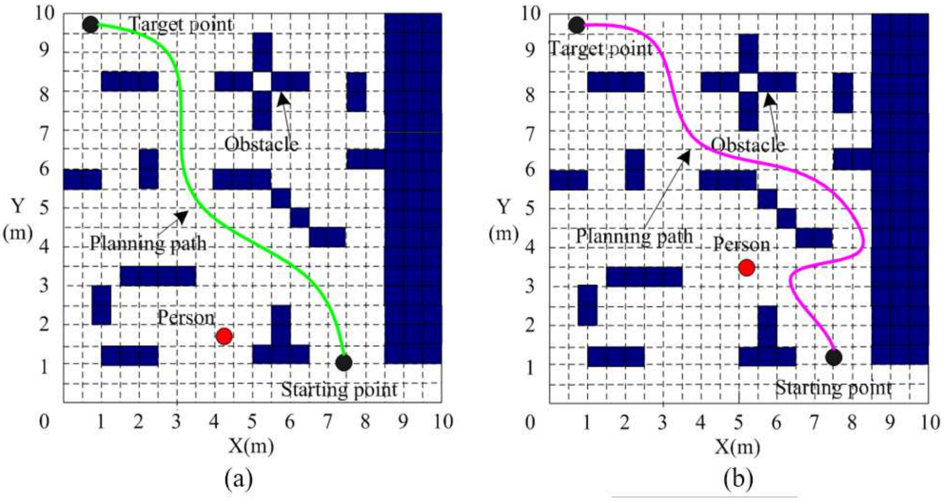


Figure 12: Dynamic obstacle avoidance experiment results

area, Fig. (b) and Fig. (c) are normal operations along a pre-defined path, Fig. (d) is near the end of the path.

Fig. 10 shows the normal operation of the mobile robot in the static obstacle of the experimental environment, the sudden emergence of human obstacle , as shown in Fig. 11 (a), mobile robot is running along the specified path, suddenly appeared in a red clothes (set the dynamic obstacle), the pre-planned path is blocked. Turble robot detects a fault, re-planning the path is re-planned; the final Fig. (b) Shows the path of the robot is re-planned, bypassing the obstacle, and returning to the original set end.

In Fig. 12, the blue is a static obstacle, the black circle is the mobile robot in the experiment, and the red is the dynamic obstacle (this experiment selects the person as a dynamic obstacle).In Fig.(a), the green solid line indicates the path planning of the robot when the dynamic obstacle person is initially immobile. In Fig. (b), the purple line indicates the re-planned path; it is obtained after the dynamic obstacle person randomly sets the obstacle point in the robot running. The results of the final experiment show that in the relatively complex environment, the robot can perform the global dynamic path planning more accurately, and it can dynamically bypass the obstacle and reach the target point in real time to realize the global path planning of real-time obstacle avoidance.

7 Conclusion

Aim at the problem of mobile robot path planning, a global dynamic path planning method based on advanced A* algorithm fusion dynamic window method is proposed. Based on the redundancy of the traditional A* algorithm, this paper designs a key selection strategy, the redundant points in the traditional A* algorithm can be removed, the performance of the global path planning is improved. Based on the dynamic window algorithm, an evaluation function considering the global optimal path is constructed; it can avoid the local optimal optimization of the dynamic window algorithm and ensure the global optimality of the path planning. Therefore, compared with the traditional A* algorithm, the path of this algorithm is more smooth, the curvature change is continuous, and the motion control parameters can be output. These are more conducive to the feedback control of the mobile robot and can be obstructed in real time. Compared with the dynamic window algorithm, the global optimization of the planned path can be ensured. Finally, the simulation and experimental results show that the improved A* algorithm and DWA combined method have good stability and real-time performance for the global real-time dynamic path planning of mobile robots, which provides a reference for the motion control of the mobile robot in the global dynamic environment.

Acknowledgment

This paper is grateful to the following projects: National Key Research and Development Program (2016YFE0111900), Shaanxi International Science and Technology Cooperation Project (2018KW-022 and 2017KW-009), Shaanxi Provincial Natural Science Fund (2017JM6041 and 2014 JM2-6093), also thanks to Xi'an Industrial University of International Science and Technology Cooperation Base Independent System and Intelligent Control International Joint Research Center of the unmanned system laboratory and autonomously intelligent control research and innovation team support for this subject.

Conflict of interest

The authors declare no conflict of interest.

Bibliography

[1] Abd, M.E.; Ahmed, M.Y.; Amgad, M.B.; Yehia, Z.E. (2018). Fixed ground-target tracking control of satellites using a nonlinear model predictive control, *Mathematical Modelling of Engineering Problems*, 5(1), 11–20, 2018.

[2] Bhattacharya, P.; Gavrilova, M.L. (2008). Roadmap-based path planning - using the Voronoi diagram for a clearance-based shortest path, *IEEE Robotics & Automation Magazine*, 15(2), 58–66, 2008.

[3] Dai, Y.; Zhu, X.; Chen, L.S.; Liu, H.; Zhang, T.; Liu, S.J.A. (2015). New Multi-Body Dynamic Model of A Seafloor Miner And Its Trafficability Evaluation, *International Journal of Simulation Modelling*, 14(4), 732–743, 2015.

[4] Dai, Y.; Pang, L.; Chen, L.; Zhu, X.; Zhang, T. (2016); A New Multi-Body Dynamic Model of a Deep Ocean Mining Vehicle-Pipeline-Ship System and Its Integrated Motion Simulation, *Journal of Mechanical Engineering*, 62(12): 757-763, 2016.

[5] Dragic, M.; Sorak, M. (2016). Simulation for Improving the Performance of Small and Medium Sized Enterprises, *Journal of Simulation modeling*, 15(4), 597-610, 2016.

[6] Eele, A.J.; Richard A. (2015). Path-planning with avoidance using nonlinear branch-and-bound optimization, *Journal of Guidance Control & Dynamics*, 32(2), 384–394, 2015.

[7] Endres, F.; Hess, J.; Sturm, J.; Cremers, D.; Burgard, W. (2014). 3D Mapping with an RGB-D Camera, *IEEE Transactionson Robotics*, 30(1), 177–187, 2014.

[8] Gao, Y.; Wu, X.; Liu, Y.; Li, J.M.; Liu J.H.(2017). A Rapid Recognition of Impassable Terrain for Mobile Robots with Low Cost Range Finder Based on Hypotheses Testing Theory, *International Journal of Computers Communications* & *Control*, 12(6), 813-823, 2017.

[9] Gao, X.; Zhang, T. (2015). Robust RGB-D simultaneous localization and mapping using planar point features, *Robotics and Autonomous Systems*, 72, 1–14, 2015.

[10] Genco, A.; Viggiano, A.; Magi, V. (2018). How to enhance the energy efficiency of HVAC systems, *Mathematical Modelling of Engineering Problems*, 5(3), 153-160, 2018.

[11] Glasius, R.; Komoda, A.; Gielen, S.C.A.M. (1995). Neural network dynamics for path planning and obstacle avoidance, *Neural Networks*, 8(1), 125–133, 1995.

[12] Herrera, C.D.; Kannala, J.; Heikkila. J. (2012). Joint depth and color camera calibration with distortion correction, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(10), 2058–2064, 2014.

[13] Kaess, M. (2015). Simultaneous localization and mapping with infinite planes, *IEEE Int. Conf. on Robotics and Automation*, 4605–4611, 2015.

[14] Lei, W.J.; Cheng, X.S.; Dai, N. (2014). Multi-model machining path planning based on improved genetic algorithm, *Journal of Mechanical Engineering*, 50(11), 153–161, 2014.

[15] Liu, C.M.; Liu, L.; Liu, C.B. (2018). Analysis of wind resistance of high-rise building structures based on computational fluid dynamics simulation technology, *International Journal of Heat and Technology*, 36(1), 376–380, 2018.

[16] Liu, J.H.; Yang, J.G.; Liu H.P. (2015). Robot global path planning based on ant colony optimization with artificial potential field, *Transactions of the Chinese Society for Agricultural Machinery*, 46(9), 18–27, 2015.

[17] Neerendra, K.; Zoltan, V. (2016). Heuristic Approaches in Robot Navigation, *IEEE International Conference on Intelligent Engineering Systems*, 219–212, 2015.

[18] Sara, J.; Jumel, F.; Simonin, O. (2017). Dynamic multi Agent patrolling Robotic application for service delivery to mobile people, *Revue d'Intelligence Artificielle*, 31(4), 379–4001, 2017.

[19] Saric, T.; Simunovic, G.; Simunovic, K.; Svalina, I. (2016). Estimation of Machining Time for CNC Manufacturing Using Neural Computing, *Journal of Simulation Modeling*, 15(4), p. 663-675, 2016.

[20] Song, X.R.; Chen, H. (2015). Stabilization Precision Control Methods of Photoelectric Aim-Stabilized System, *Optics Communication*, 9(351), 115–120, 2015.

[21] Widyotriatmo, A.; Joelianto, E.; Prasdianto, A.; Bahtiar, H.; Nazaruddin, Y.Y. (2017). Implementation of Leader-Follower Formation Control of a Team of Nonholonomic Mobile Robots, *International Journal of Computers Communications & Control*, 12(6), 871-885, 2017.

[22] Wang, C.; Mao, Y. S.; Du, K. J.; Hu, B. Q.; Song, L.F. (2016). Simulation on Local Obstacle Avoidance Algorithm for Unmanned Surface Vehicle, *Journal of Simulation modeling*, 15(3), 460-472, 2016.

[23] Xiao, Q.K.; Liu, S.Q. (2017). Motion retrieval based on Dynamic Bayesian Network and Canonical Time Warping, *Soft Computing*, 21(1), 267–280, 2017.

[24] Xiao, Q.K.; Song, R. (2017). Motion retrieval based on Motion Semantic Dictionary and HMM inference, *Soft Computing*, 21(1), 255–265, 2017.

[25] Zhang, H.; Hu, Y.L. (2016). Path planning of mobile robot based on improved D* algorithm, *Industrial Control Computer*, 29(11), 73–77, 2016.

[26] Zhu, D.Q.; Sun, B.; Li L. (2015). Algorithm for AUV's 3-Dpath planning and safe obstacle avoidance based on biological inspired model, *Control and Decision*, 30(5), 798–806, 2015.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(6), 1047-1061, December 2018.

A Neutrosophic Approach Based on TOPSIS Method to Image Segmentation

G. Xu, S. Wang, T. Yang, W. Jiang

Guojing Xu
1. Science and Technology on Avionics Integration Laboratory, Shanghai 200233, China
2. China National Aeronautical Radio Electronics Research Institute, Shanghai 200233, China

Shiyu Wang, Tian Yang, Wen Jiang*
School of Electronics and Information, Northwestern Polytechnical University
Xi'an, Shaanxi Province, 710072, China
*Corresponding author: jiangwen@nwpu.edu.cn, jiangwenpaper@hotmail.com

Abstract: Neutrosophic set (NS) is a formal framework proposed recently. NS can not only describe the incomplete information in the decision-making system but also depict the uncertainty and inconsistency, so it has applied successfully in several fields such as risk assessment, fuzzy decision and image segmentation. In this paper, a new neutrosophic approach based on TOPSIS method, which can make full use of NS information, is proposed to separate the graphics. Firstly, the image is transformed into the NS domain. Then, two operations, a modified α-mean and the β-enhancement operations are used to enhance image edges and to reduce uncertainty. At last, the segmentation is achieved by the TOPSIS method and the modified fuzzy c-means (FCM). Simulated images and real images are illustrated that the proposed method is more effective and accurate in image segmentation.

Keywords: uncertainty; neutrosophic set; TOPSIS method; FCM; image segmentation.

1 Introduction

Recently, interests have been growing in the synergistic use of multimodal imaging to get better insights into phenomena or objects. Therefore, the image analysis and processing has become more and more important [10,22,27,57,66]. In the existing image processing technology, image segmentation is one of the key tasks in many methods [2, 9, 46] and computer vision applications [13,14,21,35]. The purpose of image segmentation is to divide images into different regions for future processing based on the given criteria. Detecting generic object categories is a fundamental issue in computer vision [23, 36, 50, 67]. Its core content is to extract and separate the image information from lots of hurdles like noise, nonuniform illumination, and uneven contrast in otherwise homogenous region [8, 31]. For example, in the field of medicine, the complexity of breast cell histopathology (BCH) images makes the reliable segmentation and classification of images quite difficult [20, 53]. Some uncertain information in the images, which is not useful for image segmentation, should be dealt properly or even discarded. In practical applications, getting the completely precise information is sometimes impossible [11,26,34,56,58]. Therefore, how to deal with the uncertain information effectively is still an open issue. There are already some research about it, for example, information fusion techniques is able to decrease the information's imprecision and uncertainty through the redundancy and complementarity, thereby improve the decision accuracy.

To handle the uncertain information, many mathematical tools are presented such as D-S theory [6,25,28,48], neutrosophy theory [45,64], Z-numbers [5,29,62], single valued neutrosophic cross-entropy [7,24,54,61] and D-numbers [12]. D-S theory was introduced by Dempster and then developed by Shafer, which can well describe and process uncertainty information by assigning probability to a set not an element, so it has been widely used in many fields of information processing and fusion.

Neutrosophic set (NS) method reduces the uncertainty information in the image and obtains the better image segmentation results. In neutrosophy set theory, an event A is described by three attributes $\langle A\rangle$, $\langle Anti-A\rangle$ and $\langle Neu-A\rangle$, represents the support, opposition and neutrality of A, respectively. The neutrality $\langle Neu-A\rangle$ is neither $\langle A\rangle$ nor $\langle Anti-A\rangle$, which denotes the uncertainty of evidence, in other words, the unknown part of the information. In many cases, the boundaries between these above concepts are vague and may even be crossed together. Recently, Ali proposed the complex neutrosophic set [1]. Wang .et al extended NS to interval neutrosophic sets (INSs) [51] and single-valued neutrosophic sets (SVNSs) [52] which enriched the NS theory and promoted its development. For example, neutrosophy theory has been applied to multi-criteria decision-making [15, 59, 60]. In addition, there are many researches combine NS to image segmentation. Guo et al. [17, 19] have applied the NS and clustering method to segment image. Zhang et al. [65] use NS based on watershed method. Some other researchers combine NS with some other method [39, 49, 55, 63, 68] to handle image segmentation.

However, some methods do not consider make full use of NS information to segment image. In this paper, a new method with TOPSIS method [32, 37, 44] is proposed to obtain a better image segmentation result. Firstly, the image is transformed into the NS domain. Secondly, two operations, a modified α-mean and β-enhancement operations are used to reduce the set indeterminacy enhance the image edges. Finally, the segmentation is achieved by the TOPSIS method and the modified fuzzy c-means (FCM).

The remainder of this paper is organized as follows. The basic concepts are briefly introduced in Section 2. In Section 3, the new method is proposed. The simulated images with noise segmentation and real images segmentation are illustrated in Section 4, which show the efficiency of the proposed method. Finally, this paper is concluded in Section 5.

2 Fuzzy clustering by FCM

The FCM algorithm makes use of fuzzy membership function which is used to assign a degree of membership for each class [33, 38, 43]. The analysis of FCM algorithm can be defined as the process of organizing objects into groups whose members are similar in some ways [18]. The FCM algorithm belongs to an unsupervised fuzzy clustering algorithm.

Under the condition of a general formulation, the data to be classified can be expressed with a vector $X=\{x_1,x_2,...,x_M\}$. The vector represents a finite data set of dimension M. Assuming that $C>2$ is an integer designating the number of clusters, which sorts the vector X. Let $R^{C\times M}$ be the set of all real $C\times M$ matrices. A fuzzy C-partition of X is represented by a matrix

$$U=[\mu_{ik}]\in R^{C\times M}, \tag{1}$$

where $\mu_{ik}=\mu_i(x_k)$ is the degree of membership of the element x_k in the cluster i. The entries of the partition matrix come in the form U. The constraints following can be verified [4, 40]:

$$\begin{cases} \mu_{ik}\in[0,1], \quad 1\le i\le C,\ \ 1\leqslant k\leqslant M, \\ \sum\limits_{i=1}^{C}\mu_{ik}=1, \quad 1\leqslant k\leqslant M, \\ \sum\limits_{k=1}^{M}\mu_{ik}>0, \quad 1\leqslant i\leqslant C, \end{cases} \tag{2}$$

where U is used to depict the clusters of X, and a partition U of X can be obtained by the minimization of the FCM objective functional [3]:

$$J_q(U,V:X) = \sum_{k=1}^{M}\sum_{i=1}^{C}(\mu_{ik})^q \parallel x_k - v_i \parallel_A^2 \tag{3}$$

where $q \in [1,+\infty)$ decides the the fuzzy degree of classification results [30], and $V = (v_1, v_2, ..., v_C)$ is the vector of the cluster centers. $\parallel x_k - v_i \parallel$ is the distance between the gray level of the k-th pixel and the center of the i-th cluster. It is obvious that $\parallel x \parallel_A = \sqrt{x^T Ax}$ is any inner product norm where A is any positive definite matrix. The approximate optimization of J_q by the FCM algorithm can be obtained by iteration with the following necessary conditions for its local extreme value.

In FCM algorithm [4, 40]: assume $\parallel x_k - v_i \parallel^2{}_A > 0,\ 1 \leq i \leq C,\ 1 \leq k \leq M.$ J_q can be minimized by (U, V), only if [4, 40] as follows:

$$\mu_{ik} = \left[\sum_{j=1}^{C}(\frac{\parallel x_k - v_i\|_A}{\parallel x_k - v_j\|_A})^{\frac{2}{q-1}}\right]^{-1} \tag{4}$$

$$v_i = \frac{\sum\limits_{k=1}^{M}({\mu_{ik}}^q \times x_k)}{\sum\limits_{k=1}^{M}{\mu_{ik}}^q} \tag{5}$$

The FCM algorithm is made up of iterations alternating between Eq. (4) and Eq. (5). The algorithm is restrained to either a saddle point of J_q or a local minimum.

An algorithm of fast version of the FCM can be applied to an image, which is segmented is depending on a $1 - D$ attribute. For example, the gray level. The histogram of image of L levels is represented as HS, where L is the number of gray levels. Each pixel of the image has a characteristic which lies in the discrete set.

$$X = \{0, 1, ..., L - 1\} \tag{6}$$

In the circumstances, a gray level value is represented by each element of the data set. Then $\mu_{il} = \mu_i(l)$ is used to express the degree of membership of the element l to cluster i. The following functional is minimized by the FCM and it is similar to that of Bezdek [3]:

$$J_q(U,V:L) = \sum_{l=0}^{L-1}\sum_{i=1}^{C}(\mu_{il})^q \cdot HS(l) \cdot \parallel l - v_i \parallel_A^2 \tag{7}$$

where μ_{il} shows the membership degree of the gray level l to cluster i.

The FCM is faster than the conventional version processing the whole data set, because that the FCM operates only on the histogram [4]. Where HS be the histogram of image of L levels, L is the number of gray levels, $HS(l)$ is the number of occurrences of the level l. The computation of membership degrees of $HS(l)$ pixels is decreased to that of only one pixel with l as gray-level value. General steps of the algorithm are shown as follows in Fig. 1.

In Fig. 1, number of clusters C and the threshold value ε should be fixed firstly, the C is restricted to a range, where $2 < C < L$, $l = 0, 1, 2, ..., L - 1$. Gain the L gray levels to initialize the membership degrees μ_{il} as follows:

$$\sum_{i=1}^{C}\mu_{il} = 1,\ l = 0, 1, ..., L - 1. \tag{8}$$

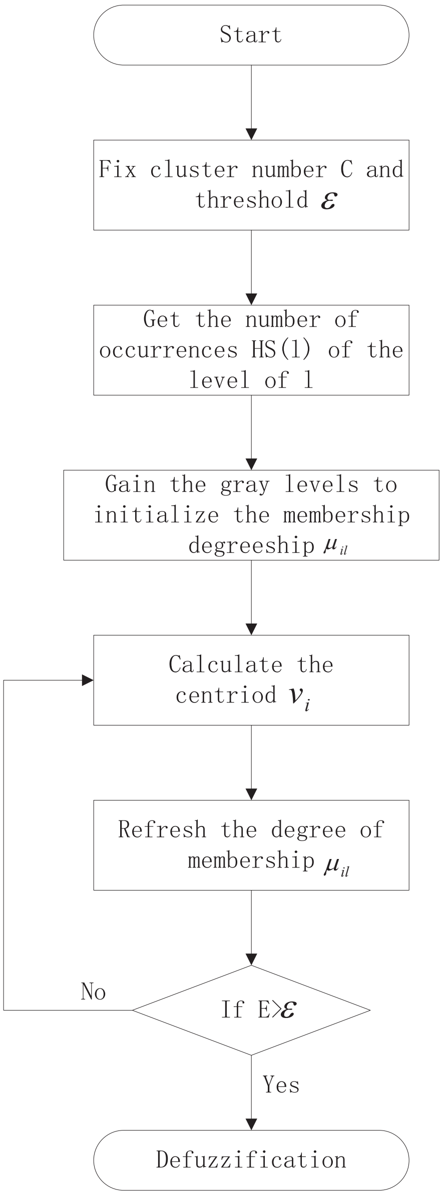

Start
Fix cluster number C and threshold ε
Get the number of occurrences HS(l) of the level of l
Gain the gray levels to initialize the membership degreeship μ_{il}
Calculate the centriod v_i
Refresh the degree of membership μ_{il}
If E>ε
No
Yes
Defuzzification

Figure 1: FCM algorithm flow diagram

Centroid v_i is calculated like this:

$$v_i = \frac{\sum_{l=0}^{L-1}[\mu_{il}^q \cdot HS(l) \cdot l]}{\sum_{l=0}^{L-1}[\mu_{il}^q \cdot HS(l)]},\ i = 1, ..., C \tag{9}$$

Then refresh the membership degrees as follows:

$$\widetilde{\mu}_{il} = \left[\sum_{j=1}^{C} \left(\frac{|l - v_i|_A}{|l - v_j|_A} \right)^{\frac{2}{q-1}} \right]^{-1} \tag{10}$$

The defect measure is presented with E, where $E = \sum_{i=1}^{C} \sum_{l=0}^{L-1} |\widetilde{\mu}_{il} - \mu_{il}|$, and then estimate it as follows: If $(E > \varepsilon)\{\mu_{il} \longleftarrow \widetilde{\mu}_{il}\}$.

Finally defuzzification. In this case, the value of clusters is fixed at some exact number, and the value of q is set to 2. The clustering operation is terminated when $\varepsilon \leq 10^{-3}$, which presents the quality of the clustering results [47].

3 The proposed method

The general steps of the proposed method are shown in Fig. 2. The proposed method is achieved by using the modified α-mean operation, TOPSIS method [41, 42] and the modified FCM. A modified α-mean is used to enhance image edges and reduce uncertainty. The β-enhancement operation is used to enhance the set T, which is suitable for segmentation. The TOPSIS method is used to take full advantage of the NS information.

3.1 Transform the image into NS

Definition 1 (Neutrosophic image)**.** Let U be a universe of the discourse, and $W \subseteq U$ which is composed by the bright pixels. A neutrosophic image P_{NS} is characterized by three membership sets T, I and F [17].

A pixel P in the given image is described as the neutrosophic set $P(t, i, f)$, which indicates that the pixel P belongs to W in the following way: the probability of t is true, probability of i is indeterminate, and f is false.

The pixel $P(i, j)$ in the image domain is transformed into the neutrosophic domain as follows,

$$P_{NS}(i, j) \rightarrow \{T(i, j), I(i, j), F(i, j)\} \tag{11}$$

where $T(i, j)$, $I(i, j)$ and $F(i,j)$ are the membership values. They are calculated by the following equations,

$$T(i, j) = \frac{\overline{g}(i, j) - \overline{g}_{\min}}{\overline{g}_{\max} - \overline{g}_{\min}}, \tag{12}$$

$$F(i, j) = \frac{\overline{g}_{max} - \overline{g}(i, j)}{\overline{g}_{\max} - \overline{g}_{\min}} = 1 - T(i, j), \tag{13}$$

where

$$\overline{g}(i, j) = \frac{1}{w \times w} \sum_{m=i-w/2}^{i+w/2} \sum_{n=j-w/2}^{j+w/2} g(m, n), \tag{14}$$

and $\overline{g}_{\max}$ and $\overline{g}_{\min}$ are the maximum and minimum of all $\overline{g}(i, j)$, respectively. The indefinite degree is obtained by

$$I(i, j) = \frac{\delta(i, j) - \delta_{\min}}{\delta_{\max} - \delta_{\min}}, \tag{15}$$

where

$$\delta(i, j) = |g(i, j) - \overline{g}(i, j)|. \tag{16}$$

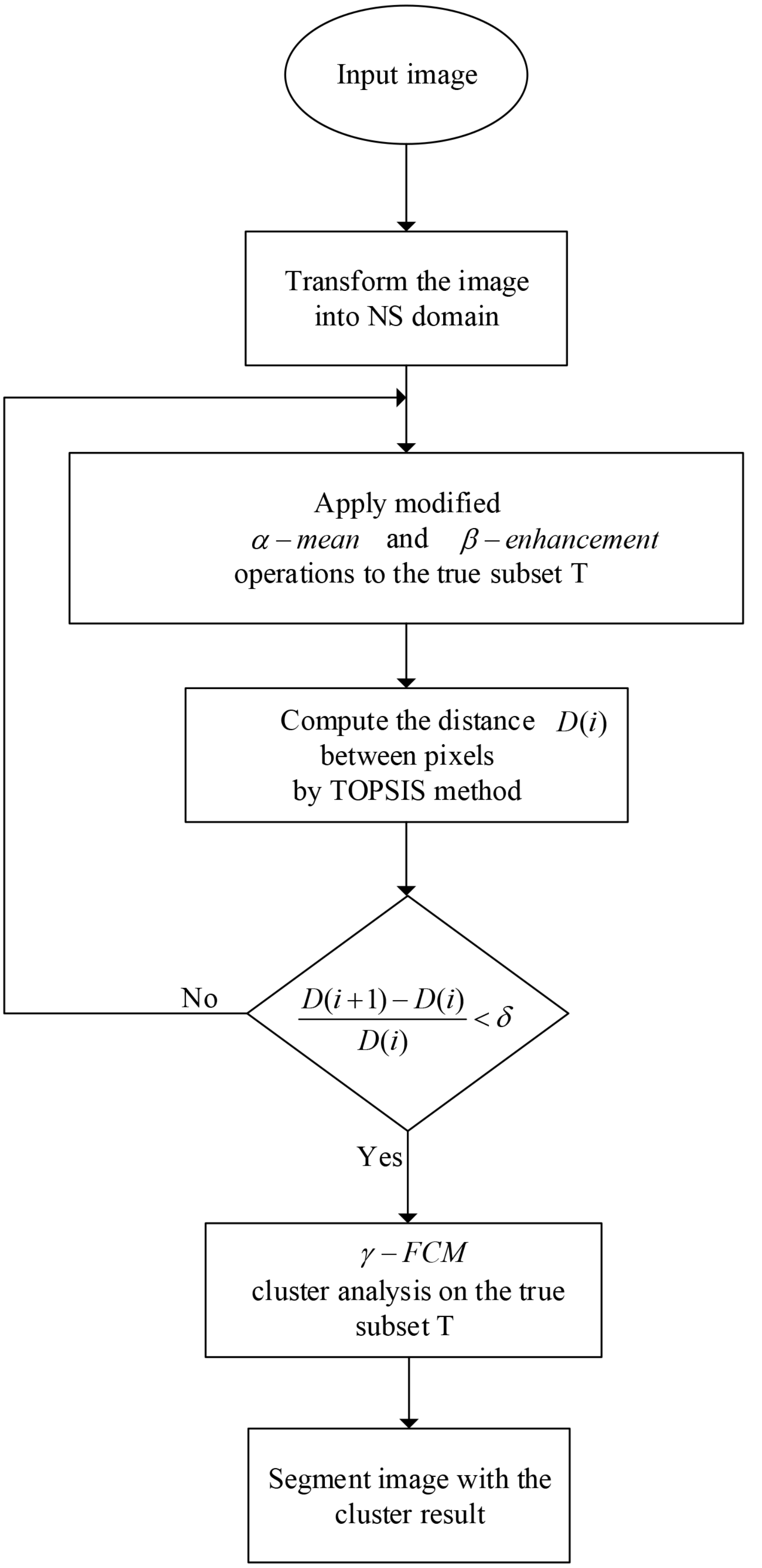

Input image
Transform the image into NS domain
Apply modified $\alpha - mean$ and $\beta - enhancement$ operations to the true subset T
Compute the distance $D(i)$ between pixels by TOPSIS method
$\frac{D(i+1)-D(i)}{D(i)} < \delta$
No
Yes
$\gamma - FCM$ cluster analysis on the true subset T
Segment image with the cluster result

Figure 2: The proposed method

3.2 Take modified α-mean operation on the true subset T

Definition 2 (α-mean operation)**.** The modified α-mean operation for P_{NS}, $\overline{P}_{NS}(\alpha)$ is

$$\overline{P}_{NS}(\alpha) = P(T(\alpha), I(\alpha), F(\alpha)). \tag{17}$$

The α-mean operation of the neutrosophic set is defined as follows,

$$\overline{T}(\alpha) = \begin{cases} T, I < \alpha \\ T_{\alpha}^{m}, I \geqslant \alpha \end{cases} \tag{18}$$

$$\begin{aligned} &T_{\alpha}^{m}(i,j) = \texttt{Median}(T(m,n)), \\ &i - \frac{w}{2} \leqslant m \leqslant i + \frac{w}{2},\ j - \frac{w}{2} \leqslant n \leqslant j + \frac{w}{2} \end{aligned} \tag{19}$$

where $T_{\alpha}^{m}(i,j)$ is the median of $T(m,n)$,

$$\overline{F}(\alpha) = 1 - \overline{T}(\alpha) \tag{20}$$

$$\overline{I_{\alpha}}(i,j) = \frac{\overline{\delta_T}(i,j) - \delta_{T\min}}{\delta_{T\max} - \delta_{T\min}} \tag{21}$$

$$\overline{\delta_T}(i,j) = |\overline{T}(i,j) - \overline{\overline{T}}(i,j)| \tag{22}$$

$$\begin{aligned} &\overline{\overline{T}}(i,j) = \texttt{Median}(\overline{T}(m,n)), \\ &i - \frac{w}{2} \leqslant m \leqslant i + \frac{w}{2},\ j - \frac{w}{2} \leqslant n \leqslant j + \frac{w}{2} \end{aligned} \tag{23}$$

After the modified α-mean operation, the distribution of the elements in I becomes more uniform. Furthermore, the image edges are enhanced, and the membership set T becomes more distinct, which is suitable for segmentation. In this paper, α is 0.85 according to the experimental results.

3.3 Take β-enhancement operation on the true subset T

The intensification operation for the membership μ in fuzzy set is defined as follows

$$\begin{aligned} \mu'(i,j) &= f(\mu(i,j)) \\ &= \begin{cases} 2\mu^2(i,j), \mu(i,j) \leqslant 0.5 \\ 1 - 2(1-\mu(i,j))^2, \mu(i,j) > 0.5 \end{cases} \end{aligned} \tag{24}$$

Definition 3 (β-enhancement operation)**.** A β-enhancement operation for P_{NS}, $P'_{NS}(\beta)$, is defined as follows [17]

$$P'_{NS}(\beta) = P(T'(\beta), I'(\beta), F'(\beta)) \tag{25}$$

$$T'(\beta) = \begin{cases} T, I < \beta \\ T'_{\beta}, I \geqslant \beta \end{cases} \tag{26}$$

$$F'(\beta) = \begin{cases} F, I < \beta \\ F'_{\beta}, I \geqslant \beta \end{cases} \tag{27}$$

where $\beta = \overline{I(i,j)}$ is a dynamic parameter that adjusts the threshold based on the image.

$$T'_{\beta}(i,j) = \begin{cases} 2T^2(i,j), T(i,j) \leqslant 0.5 \\ 1 - 2(1 - T(i,j))^2, T(i,j) > 0.5 \end{cases} \tag{28}$$

$$F'_\beta(i,j)=\begin{cases}2F^2(i,j),F(i,j)\leqslant 0.5\\1-2(1-F(i,j))^2,F(i,j)>0.5\end{cases}\tag{29}$$

$$I'_\beta(i,j)=\frac{\delta'_T(i,j)-\delta'_{T\min}}{\delta'_{T\max}-\delta'_{T\min}}\tag{30}$$

$$\delta'_T(i,j)=|T'(i,j)-\overline{T}'(i,j)|\tag{31}$$

$$\overline{T}'(i,j)=\frac{1}{w\times w}\sum_{m=i-w/2}^{i+w/2}\sum_{n=j-w/2}^{j+w/2}T'(m,n).\tag{32}$$

After the β-enhancement operation, the membership set T becomes more distinct and has less information redundancy, which is appropriate for segmentation.

3.4 TOPSIS method based on NS

Definition 4. Assume that an completely accurate pixel $P_{NSture}(i,j)$=[1,0,0], in the image is described that it belongs to W without any doubt. Moreover, $P_{NSfalse}(i,j)$=[0,1,1] is described that it does not belongs to W. $D(i)$ in the image indicates the distance between the pixel $P_{NS}(i,j)$ and pix $P_{NSture}(i,j)$

$$\begin{aligned}D(i)&=\frac{D(P_{NS}(i,j),P_{NSture}(i,j))}{D(P_{NSture}(i,j),P_{NSfalse}(i,j))}\\&=\frac{\sqrt{(T(i)-1)^2+(I(i)-0)^2+(F(i)-0)^2}}{\sqrt{(1-0)^2+(0-1)^2+(0-1)^2}}\end{aligned}\tag{33}$$

In the method, the information of the pixels are made full use. In addition to the probability T and F that the pixels is in a category or not, its degree of neutrality I is also considered. In other words, uncertain information in the obtained image is involved in the clustering process. Accordingly, our method is more accurate and more reasonable in image segmentation.

3.5 Modified FCM based on NS

Considering the effect of indeterminacy, two sets, T and I are composed into a new value for FCM clustering:

$$X(i,j)=\begin{cases}T(i,j),I(i,j)\leqslant\gamma\\\overline{T_\gamma}(i,j),I(i,j)>\gamma\end{cases}\tag{34}$$

Here, γ=0.85 is determined by experiments.

Then the modified FCM is applied to the subset T as introduced in Section 2. First, the pixel information in the matrix $X(i,j)$ is transformed into a vector x, then the image modified information vector x is subjected to the following recursive algorithm for cluster segmentation.

$$J_q(U,V:X)=\sum_{k=1}^{M}\sum_{i=1}^{C}(\mu_{ik})^q\parallel x_k-v_i\parallel_A^2\tag{35}$$

$$\mu_{ik}=\left[\sum_{j=1}^{C}(\frac{\parallel x_k-v_i\|_A}{\parallel x_k-v_j\|_A})^{\frac{2}{q-1}}\right]^{-1}\tag{36}$$

$$v_i = \frac{\sum\limits_{k=1}^{M} ({\mu_{ik}}^q \times x_k)}{\sum\limits_{k=1}^{M} {\mu_{ik}}^q} \tag{37}$$

where $J_q(U, V : X)$ can be minimized by (U, V).

The proposed method can be summarized as the following steps:

Step 1. Transform the image into NS domain using Eq. (12) to Eq. (16).

Step 2. Perform the modified α-mean and β-enhancement operations on the subset T through Eq. (17) to Eq. (32).

Step 3. Compute the distance $D(i)$ using Eq. (33).

Step 4. If $(D(i+1)\text{-}D(i))/D(i) < \delta$, go to Step 5; else go back to Step 2.

Step 5. Apply the modified FCM to the subset T by Eq. (34) to Eq. (37).

Step 6. Segment image.

4 Experiments and discussions

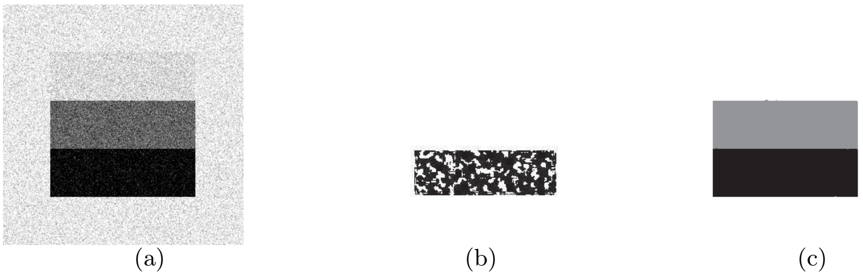


Figure 3: **Simulated image 1 used for segmentation:** (a) original simulated image, (b) segmentation result of Guo's method [17], (c) segmentation result of proposed method

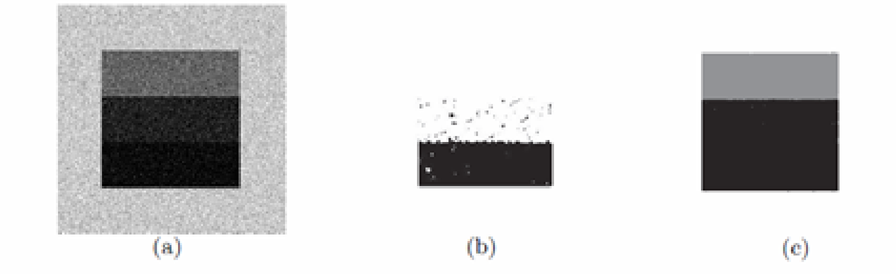


Figure 4: **Simulated image 2 used for segmentation:**(a) original simulated image, (b) segmentation result of Guo's method [17], (c) segmentation result of proposed method

In order to test the effectiveness of the proposed method, the Gaussian noise whose expectation is 0 and the variance is 0.015, that is, $N(0, 0.015)$ is added to two simulated images. As shown in Fig. 3(a), the upper gray belt is close to the background gray, which is easy to be confused. In Fig. 4(a), gray belts of two layers below are difficult to be distinguished. Fig. 3(b) and Fig. 4(b) are the segmentation results of the Guo's method [17]. In Fig. 3(c) and Fig. 4(c), segmentation results of the proposed method in this paper, the shape and size of image are segmented better. The three regions are distinct and easy to be recognized by the proposed approach. The edges of the image are also more clear. The error segmentation rates of Fig. 3(b) and Fig. 4(b) are 0.424 and 0.611, which show that many pixels are wrongly segmented by the Guo's method [17]. The error segmentation rates of Fig. 3(c) and Fig. 4(c) are 0.243 and 0.384, which perform that a less of pixels are wrongly segmented by the proposed method.

Then let's consider the consequent of dividing some actual images. Fig. 5(a) and Fig. 6(a) are real lena images with different noise levers. The error segmentation rates are 0.815 and 0.718 in Fig. 5(b) and Fig. 6(b) segmented by Guo, respectively. However, the false judgement rate of our proposed method is just 0.190 and 0.367 in Fig. 5(c) and Fig. 6(c). In the split photos, hat, hair and woman's face, three main areas in the image are segmented more correctly and clearly. More pixels are segmented correctly and the regions are more consistent and homogenous in Fig. 5(c) and Fig. 6(c), which are better to be distinguished.

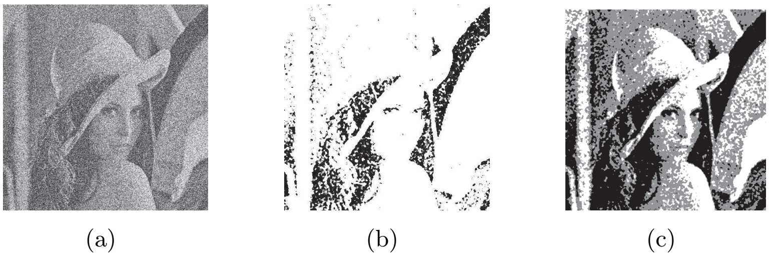


Figure 5: **Real image used for segmentation:** (a) original Lena image with gaussian noise; (b) segmentation result of Guo's method [17]; (c) segmentation result of proposed method

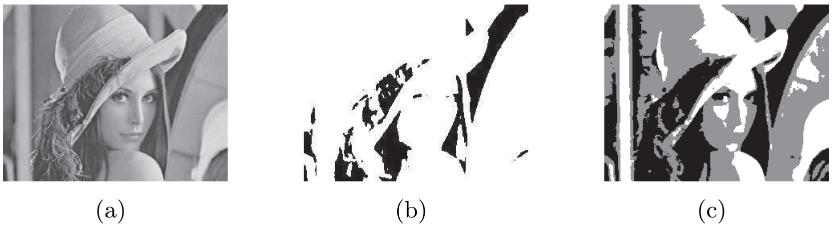


Figure 6: **Real image used for segmentation:** (a) original Lena image; (b) segmentation result of Guo's method [17]; (c) segmentation result of proposed method

5 Conclusions

In this paper, a new method with the TOPSIS method, which effectively enhances the image edges and reduces the indeterminacy with the modified α-mean operation and β-enhancement operation, is proposed. With the TOPSIS method, the NS information in the image is fully utilized. The information of T, I and F in the NS domain is made the best. The experimental results with simulated images and real images show that the proposed method is more effective in handling vague edges and in better making full use of NS information. With different noise levels and different kinds of noise, especially complex noise contained in the real medical images, the proposed method performs better and more accurately than the existing method. Image segmentation results are obtained more precisely and reasonably. At the same time, the proposed method shows that the notable validity in increasing boundary accuracy and region harmony. The comparison between the experimental results and the error segmentation rates can illustrate that our proposed method can handle the uncertain and indeterminate information better. In the further research directions, it may find more applications in diverse fields of control theory, computer vision, and artificial intelligence, especially in the image processing field.

Acknowledgments

The work is partially supported by Science and Technology on Avionics Integration Laboratory and Aviation Science Foundation (Program No. 20165553036).

Conflict of interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Bibliography

[1] Ali, M.; Smarandache, F. (2017); Complex neutrosophic set, *Neural Computing & Applications*, 28(7), 1817–1834, 2017.

[2] Ashraf, M.; Sarim, M.; Shaikh, A.B. (2017). Cellular-Cut-Interactive n-Dimensional Image Segmentation Using Cellular Automata. *International Journal of Pattern Recognition & Artificial Intelligence*, 2017.

[3] Bezdek, J.C.; Ehrlich, R.; Full, W. (1984). FCM: The fuzzy c-means clustering algorithm, *Computers & Geosciences*, 10(2-3), 191–203, 1984.

[4] Cannon, R.L.; Dave, J.V.; Bezdek, J.C. (1986). Efficient Implementation of the Fuzzy c-Means Clustering Algorithms, *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 8(2), 248–255, 1986.

[5] Chen, L.; Deng, Y. (2018). A new failure mode and effects analysis model using Dempster-Shafer evidence theory and grey relational projection method, *Engineering Applications of Artificial Intelligence*, 76, 13–20, 2018, doi:10.1016/j.engappai.2018.08.010.

[6] Dempster, A.P. (1967). Upper and Lower Probabilities Induced by a Multi-valued Mapping, *Annals of Mathematical Statistics*, 38(2),325–339, 1967.

[7] Deng, X. (2018). Analyzing the monotonicity of belief interval based uncertainty measures in belief function theory, *International Journal of Intelligent Systems*, 33(9),1869–1879, 2018.

[8] Deng, X.; Han, D.; Dezert, J.; Deng, Y.; Shyr, Y. (2016). Evidence combination from an evolutionary game theory perspective, *IEEE Transactions on Cybernetics*, 46(9), 2070–2082, 2016.

[9] Deng, X. Jiang, W. (2018). Dependence assessment in human reliability analysis using an evidential network approach extended by belief rules and uncertainty measures, *Annals of Nuclear Energy*, 117, 183–193, 2018.

[10] Deng, X.; Jiang, W.; Wang, Z. (2019). Zero-sum polymatrix games with link uncertainty: A Dempster-Shafer theory solution, *Applied Mathematics and Computation*, 340, 101–112, 2019.

[11] Deng X.; Xiao, F.; Deng, Y. (2017). An improved distance-based total uncertainty measure in belief function theory, *Applied Intelligence*, 46(4), 898–915, 2017.

[12] Deng, Y. (2012). D numbers: theory and applications, *Journal of Information & Computational Science*, 9(9), 2421–2428, 2012.

[13] Dou, R.; Nan, G. (2018). Optimizing sensor network coverage and regional connectivity in industrial IoT systems, *IEEE Systems Journal*, 11(3), 1351–1360, 2018.

[14] Eklund, A.; Dufort, P.; Forsberg, D.; Laconte, S.M. (2013). Medical image processing on the GPU: Past, present and future, *Medical Image Analysis*, 17(17), 1073–1094, 2013.

[15] Fei, L.; Deng, Y.; Hu, Y. (2018). DS-VIKOR: A New Multi-criteria Decision-Making Method for Supplier Selection, *International Journal of Fuzzy Systems*, 2018, p. 10.1007/s40815–018–0543–y.

[16] Greco, S.; Matarazzo, B.; Slowinski, R. (2001). Rough sets theory for multicriteria decision analysis, *European Journal of Operational Research*, 129(1), 1–47, 2001.

[17] Guo, Y.; Cheng, H.D. (2009). New neutrosophic approach to image segmentation, *Pattern Recognition*, 42(5), 587–595, 2009.

[18] Guo, Y.; Sengur, A. (2015). NECM: Neutrosophic evidential c -means clustering algorithm, *Neural Computing & Applications*, 26(3), 561–571, 2015.

[19] Guo, Y.; Sengur, A. (2013). A novel color image segmentation approach based on neutrosophic set and modified fuzzy c -means, *Circuits, Systems, and Signal Processing*, 32(4), 1699–1723, 2013.

[20] Han, Y.; Deng, Y. (2018). An enhanced fuzzy evidential DEMATEL method with its application to identify critical success factors, *Soft computing*, 22(15), 5073–5090, 2018.

[21] He, Z.; Jiang, W. (2018). An evidential Markov decision making model, *Information Sciences*, 467, 357–372, 2018.

[22] He, Z.; Jiang, W. (2018). An evidential dynamical model to predict the interference effect of categorization on decision making, *Knowledge-Based Systems*, 150, 139–149, 2018.

[23] Hong, C.; Zhang, J.; Cao, X.B.; Du, W.B. (2016). Structural properties of the Chinese air transportation multilayer network, *CHAOS SOLITONS & FRACTALS*, 86, 28–34, 2016.

[24] Hu, K.; Ye, J.; Fan, E.; Shen, S.; Huang, L.; Pi, J. (2017). A novel object tracking algorithm by fusing color and depth information based on single valued neutrosophic cross-entropy, *Journal of Intelligent & Fuzzy Systems*, 32(3), 1775–1786, 2017.

[25] Jiang, W. (2018). A correlation coefficient for belief functions, *International Journal of Approximate Reasoning*, 2018, p. Published on line, Doi: 10.1016/j.ijar.2018.09.001.

[26] Jiang, W.; Hu, W. (2018). An improved soft likelihood function for Dempster-Shafer belief structures, *International Journal of Intelligent Systems*, 33(6), 1264–1282, 2018.

[27] Jiang, W.; Huang, C. (2018). A Multi-criteria Decision-making Model for Evaluating Suppliers in Green SCM, *International Journal of Computers Communications & Control*, 13(3), 337–352, 2018.

[28] Jiang, W.; Wang, S. (2017). An Uncertainty Measure for Interval-valued Evidences, *International Journal of Computers Communications & Control*, 12(5), 631–644, 2017.

[29] Kang, B.; Deng, Y.; Hewage, K.; Sadiq, R. (2018). A method of measuring uncertainty for Z-number, *IEEE Transactions on Fuzzy Systems*, 2018, p. DOI:10.1109/TFUZZ.2018.2868496.

[30] Kannan, S.R.; Ramathilagam, S.; Devi R.; Hines, E. (2012). Strong fuzzy c-means in medical image data analysis, *Journal of Systems and Software*, 85(11), 2425–2438, 2012, doi:10.1016/j.jss.2011.12.020.

[31] Kittaneh, O.A.; Khan, M.A.U.; Akbar, M.; Bayoud H.A. (2016). Average Entropy: A New Uncertainty Measure with Application to Image Segmentation, *American Statistician*, 70, 18–24, 2016.

[32] Kuo, T. (2016). A modified TOPSIS with a different ranking index, *European Journal of Operational Research*, 260, 2016.

[33] Li, P.; Chen, Z.; Yang, L.T.; Zhao, L.; Zhang, Q. (2017). A privacy-preserving high-order neuro-fuzzy c-means algorithm with cloud computing, *Neurocomputing*, 2017.

[34] Li, Y.; Deng, Y. (2018). Generalized Ordered Propositions Fusion Based on Belief Entropy, *International Journal of Computers Communications & Control*, 13(5), 792–807, 2018.

[35] Li, Z.; Chen, L.; Nan, G. (2018). Small-scale Renewable Energy Source Trading: A Contract Theory Approach, *IEEE Transactions on Industrial Informatics*, 14(4), 1491–1500, 2018.

[36] Liang, W.; He, J.; Wang, S.; Yang, L.; Chen, F. (2018). Improved cluster collaboration algorithm based on wolf pack behavior, *Cluster Computing*, 2018, p. Published on line, doi: 10.3390/s17040922.

[37] Lourenzutti, R.; Krohling, R.A.; Reformat, M.Z. (2017). Choquet based TOPSIS and TODIM for dynamic and heterogeneous decision making with criteria interaction, *Information Sciences*, 408, 41–69, 2017.

[38] Mahela, O.P.; Shaik, A.G. (2017). Power quality recognition in distribution system with solar energy penetration using S -transform and Fuzzy C-means clustering, *Renewable Energy*, 106, 37–51, 2017.

[39] Mohan, J.; Krishnaveni, V.; Guo, Y. (2013). MRI denoising using nonlocal neutrosophic set approach of wiener filtering, *Biomedical Signal Processing & Control*, 8(6), 779–791, 2013.

[40] Muller, H.; Michoux, N.; Bandon, D.; Geissbuhler, A. (2004). A review of content based image retrieval systems in medical applications clinical benefits and future directions, *International Journal of Medical Informatics*, 73(1), 1–23, 2004.

[41] Nădăban, S.; Dzitac, S. (2016). Neutrosophic TOPSIS: A general view, *2016 6th International Conference on Computers Communications and Control, IEEE*, 250–253, 2016.

[42] Nădăban, S.; Dzitac, S.; Dzitac, I. (2016). Fuzzy TOPSIS: A general view, *Procedia Computer Science*, 91, 823–831, 2016.

[43] Nayak, J.; Naik, B.; Behera, H.S.; Abraham, A.(2017); Hybrid Chemical Reaction based Metaheuristic with Fuzzy c-means Algorithm for Optimal Cluster Analysis, *Expert Systems with Applications*, 79, 282–295, 2017.

[44] Onu, U.P.; Xie, Q.; Xu, L. (2017). A Fuzzy TOPSIS model Framework for Ranking Sustainable Water Supply Alternatives, *Water Resources Management An International Journal Published for the European Water Resources Association*, 1–15, 2017.

[45] Peng, J.; Wang, J.; Wu, X.; Wang, J.; Chen, X. (2015). Multi-valued Neutrosophic Sets and Power Aggregation Operators with Their Applications in Multi-criteria Group Decision-making Problems, *International Journal of Computational Intelligence Systems*, 8(2), 345–363, 2015.

[46] Qian, P.; Zhao, K.; Jiang, Y.; Su, K.H.; Deng, Z.; Wang, S.; et al. (2017). Knowledge-leveraged transfer fuzzy C -Means for texture image segmentation with self-adaptive cluster prototype matching, *Knowledge-Based Systems*, 2017.

[47] Reyes-Galaviz, O.F.; Pedrycz, W. (2017). Enhancement of The Classification and Reconstruction Performance of Fuzzy C-Means with Refinements of Prototypes, *Fuzzy Sets & Systems*, 318, 80–99, 2017.

[48] Shafer, G. (1976). A Mathematical Theory of Evidence, *New Jersey, Princetion University Press*, 1976.

[49] Shan, J.; Cheng, H.D.; Wang, Y. (2012). A novel segmentation method for breast ultrasound images based on neutrosophic l-means clustering, *Medical Physics*, 39(9), 5669–5682, 2012.

[50] Wang, B.; Xiong, H.; Jiang, X.; Zheng, Y.F. (2014). Data-Driven Hierarchical Structure Kernel for Multiscale Part-Based Object Recognition, *IEEE Transactions on Image Processing*, 23(4), 1765–1778, 2014, doi:10.1109/TIP.2014.2307480.

[51] Wang, H.; Smarandache, F.; Sunderraman, R.; Zhang, Y.Q. (2005). Interval Neutrosophic Sets and Logic: Theory and Applications in Computing: Theory and Applications in Computing. vol. 5, *Infinite Study*, 2005.

[52] Wang, H.; Smarandache, F.; Zhang, Y.; Sunderraman, R. (2010). Single valued neutrosophic sets, *Rev Air Force Acad*, 17, 4–10, 2010.

[53] Wang, P.; Hu, X.; Li, Y.; Liu, Q.; Zhu, X. (2016). Automatic cell nuclei segmentation and classification of breast cancer histopathology images, *Signal Processing* 122, 1 – 13, 2016.

[54] Xiao, F. (2019). Multi-sensor data fusion based on the belief divergence measure of evidences and the belief entropy, *Information Fusion*, 46(2019), 23–32, 2019;.

[55] Xin, Z.; Shitong, W. (2012). Neutrosophic image segmentation approach based on similarity, *Application Research of Computers*, 29(6), 2371–2374, 2012.

[56] Xiong, H.; Zheng, D.; Zhu, Q.; Wang, B.; Zheng, Y.F. (2013). A Structured Learning-Based Graph Matching Method for Tracking Dynamic Multiple Objects, *IEEE Transactions on Circuits and Systems for Video Technology*, 23(3), 534–548, 2013, doi:10.1109/TCSVT.2012.2210801.

[57] Xu, S.; Jiang, W.; Deng, X.; Shou, Y. (2018). A modified Physarum-inspired model for the user equilibrium traffic assignment problem, *Applied Mathematical Modelling*, 55, 340–353, 2018.

[58] Xu, Z.; Hu, C.H.; Yang, F.; Kuo, S.H.; Goh, C.K.; Gupta, A.; et al. (2017). Data-driven Inter-Turn Short Circuit Fault Detection in Induction Machines, *IEEE Access*, 5(1), 25055–25068, 2017.

[59] Ye, J. (2013). Multicriteria decision-making method using the correlation coefficient under single-valued neutrosophic environment, *International Journal of General Systems*, 42(4), 386–394, 2013.

[60] Ye, J. (2014). A multicriteria decision-making method using aggregation operators for simplified neutrosophic sets, *Journal of Intelligent* & *Fuzzy Systems*, 26(5), 2459–2466, 2014.

[61] Yin, L.; Deng, Y. (2018). Toward uncertainty of weighted networks: An entropy-based model, *Physica A: Statistical Mechanics and its Applications*, 508, 176–186, 2018, doi:http://doi.org/10.1016/j.physa.2018.05.067.

[62] Zadeh, L.A. (2011). A Note on Z-numbers, *Information Sciences*, 181(14), 2923–2932, 2011.

[63] Zhang, G.; Wang, D. (2014). Neutrosophic image segmentation approach integrated LPG & PCA, *Journal of Image* & *Graphics*, 19(5), 693–700, 2014.

[64] Zhang, H.; Ji, P.; Wang, J.; Chen, X. (2015). An Improved Weighted Correlation Coefficient Based on Integrated Weight for Interval Neutrosophic Sets and its Application in Multi-criteria Decision-making Problems, *International Journal of Computational Intelligence Systems*, 8(6), 1027–1043, 2015.

[65] Zhang, M.; Zhang, L.; Cheng, H.D. (2010). A neutrosophic approach to image segmentation based on watershed method, *Signal Processing*, 90(5), 1510–1517, 2010.

[66] Zhang, X.; Mahadevan, S. (2017). Aircraft re-routing optimization and performance assessment under uncertainty, *Decision Support Systems*, 96, 67–82, 2017.

[67] Zhang, X.; Mahadevan, S.; Deng, X. (2017). Reliability analysis with linguistic data: An evidential network approach, *Reliability Engineering* & *System Safety*, 162, 111–121, 2017.

[68] Zhao, X.; Wang, S.T.; Juna, W.U. (2011). Neutrosophic image segmentation approach based on thermal balance, *Computer Engineering*, 37(19), 210–212, 220, 2011.

Author index