ISSN 2278-3091
Volume 8, No.1.1, 2019
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse2681.12019.pdf
https://doi.org/10.30534/ijatcse/2019/2681.12019

Application for Position and Load Reference Generation of a Simulated Mechatronic Chain

Victor Vladareanu[1], Sergiu Boris Cononovici[1], Marcel Migdalovici[1], Hongbo Wang[2], Yongfei Feng[2]*, and Florentin Smarandache[3]

[1]Robotics and Mechatronics Dept., Institute of Solid Mechanics of the Romanian Academy
Bucharest, 010141, Romania
victor.vladareanu@vipro.edu.ro cononovici@imsar.bu.edu . migdal@imsar.bu.edu.ro
[2]Parallel Robot and Mechatronic System Laboratory of Hebei Province, Yanshan University,
Qinhuangdao, 066004, China
hongbo_w@ysu.edu.cn , yf_feng@126.com
[3]Department of Mathematics, University of New Mexico,
New Mexico, 705 Gurley Avenue, Gallup, NM 87301, USA
smarand@unm.edu

ABSTRACT

The paper presents the position and load reference generation for a motor stand simulating a mechatronic chain, in this case a three degree of freedom robot leg. The task is accomplished using three PLC controlled motors in position as the robot joint actuators coupled with three controlled in torque, simulating the load at each simulation time-step. The paper briefly discusses the mathematical model and presents the visual interface used in the simulation, which is then to be further integrated into a virtual environment robot control application.

Keywords: VIPRO platform, robot simulation, graphical user interface, reference generation.

1. INTRODUCTION

The walking robots' ability to adapt to uneven terrain makes them very useful in today's intelligent applications, from rescue operations and autonomous firefighting to elderly and disabled care. Walking locomotion ensures leg adaptation to the available environment, avoidance of unsuitable step positions and robot movement adapted to the terrain configuration [1-4].

The walking robot is physically composed of a body containing the elements capable of executing the allocated tasks and the robot legs, generally with three degrees of freedom, which ensure locomotion. The legs are either in a support phase – the phase in which the leg is in contact with the ground, the body is either stationary or moving along the walking route, or in the walking phase – in which the leg is off the ground, executing motion in relation to the robot body with the aim of contacting the ground in a new position. The legs are RRR kinematic chains with active couples [5, 6].

The design of the leg control system requires the existence of a way to simulate the walking process, virtually as well as physically, which is one of the goals of the overall project [7-10]. Numerous intelligent control interfaces based on advanced control strategies, such as neutrosophic control [4,8], extended control (Extenics) [6,9], human adaptive mechatronics [2,3], have been developed.

Uniform and periodic walking is studied on a terrain with possible irregularities. The walking is periodic if similar positions of the same leg, during the mechanical walking, appear at a given time interval. The time interval is a walking cycle and the movement done during it is an elementary step of the robot. Periodic walking allows a relatively simple control and a smooth motion of the robot body. Irregular walking appears in applications on uneven terrain.

This paper is divided as follows: Section II will describe the physical model investigated from which load reference generation is obtained, Section III will show the visual application interface and describe the immediate action of each of its elements, Section IV discusses the expected functionality of the visual interface as a whole, Section V delves into consideration regarding the code called underneath the graphical elements and its structure, and Section VI sums up the main points of the paper and the integration of the obtained results into the larger intelligent virtual environment platform.

2. LOAD REFERENCE GENERATION

Motion with an elementary step corresponds to a walking cycle and transports the characteristic point M of the robot, usually chosen as the centre of mass or the geometric centre or on a vertical axis passing through it. Point M is the origin of a referential attached to the robot. The trajectory of point M is given in relation to a referential attached to a fixed point in the workspace.

Available equipment includes a stand composed of three PLC – controlled actuators, used for simulating the kinematic chain of a robot leg. Another three actuators are physically coupled to them, simulating the resistive loads.

Taking into account that actuating the kinematic chain of the leg does not imply the appearance of new large accelerations, as well as the fact that actuating the elements is done through reducers with large transmission fractions (i.e. n = 100), the

resistive moments can be calculated using a kineto – static method. The quasi – static moments are calculated at each sample time.

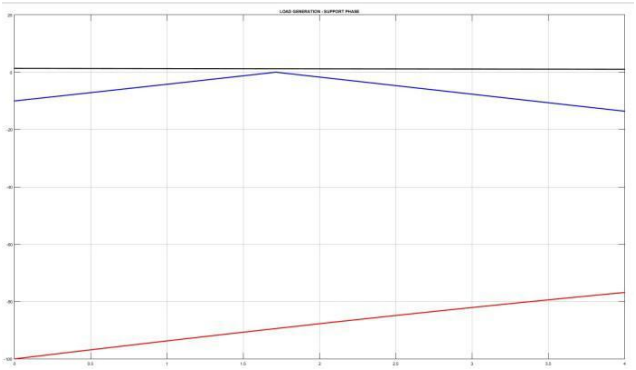

Figure 1:. Load profiles for the support phase

In order to control the motors on the simulation stand for the joints of a walking hexapod robot, a calculation and transfer model is developed for the joint control references. The application simulating the robot leg on the motor stand requires knowledge of the position of each joint, as well as the loads they are sustaining. The dataset calculated as the robotic actuator load, simulated by a torque-controlled coupled motor at each simulation time-step, constitutes the control references for the secondary motor set, which replace the load during simulation.

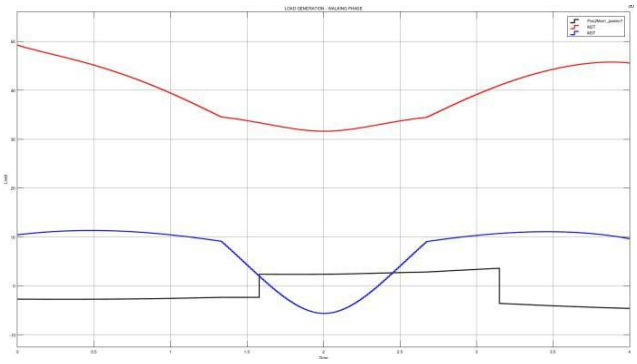

Figure 2:. Load profiles for the walking phase

To properly test the mechatronic configuration within the virtual environment interface (thus allowing an unspecialized user to test various assumptions about their model), the position reference and load reference of a specific joint need to be given exactly at each time-step. Therefore, the two reference datasets are sent jointly to the PLC control structure, which synchronizes the application parameters.

For the considered example (an RRR kinematic chain representing the leg of a walking hexapod robot), the results of the mathematical model are shown in the next section. Further details are available in [11]. Using these mathematical equations and a standard set of input values (see discussion on "*Default*" setting), load profiles were obtained for the three joints. These are shown in Figures 1 and 2, for the support phase and walking phase, respectively. The simulation was run for 4 seconds, which would be the time it takes for a complete leg movement. For both figures, the black line represents the load value of the first joint (M_{B1}), the blue line the second joint (M_{B2}) and the red line, the third (M_C).

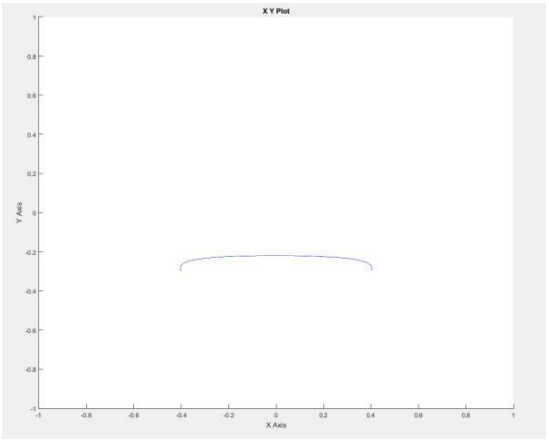

Figure 3. Leg motion during walking phase

The equivalent motion made by the robot leg during the walking phase is shown in Figure 3. It is ellipsoidal in the plane perpendicular to the second and third joints (MB2 and MC).

The calculation of a specific load set, dependant on the input values fed into the calculations previously presented, can then be further sent to the PLC – motor system for actual hardware testing. The entire process can be achieved through a graphical user interface, which allows setting the initial inputs, specifying the simulation parameters and visualizing the obtained results, before feeding them into the hardware actuator system. Under the hood, this graphical user interface calls routines and functions which implement the equations previously discussed in the paper. The final aim is to additionally provide remote access and control capabilities, which is to be achieved in a future version of this implementation.

3. APPLICATION INTERFACE

The mathematical model, functioning and detailed calculations are geared towards implementation as a standalone application, as well as integration into a massive robot control and virtualization platform, such as the VIPRO project [8, 10].

For facilitating the implementation of the proposed model in testing the mechatronic chain, a graphical interface is designed and implemented, aimed at visualizing the working environment of the data sent to the PLC stand. Figure 4 shows the graphical elements and the control buttons.

The upper side of the interface contains three main areas which specify or select the options necessary for simulation and modelling. The mathematical model is described by the

following general formulae for the support phase and walking phase, respectively.

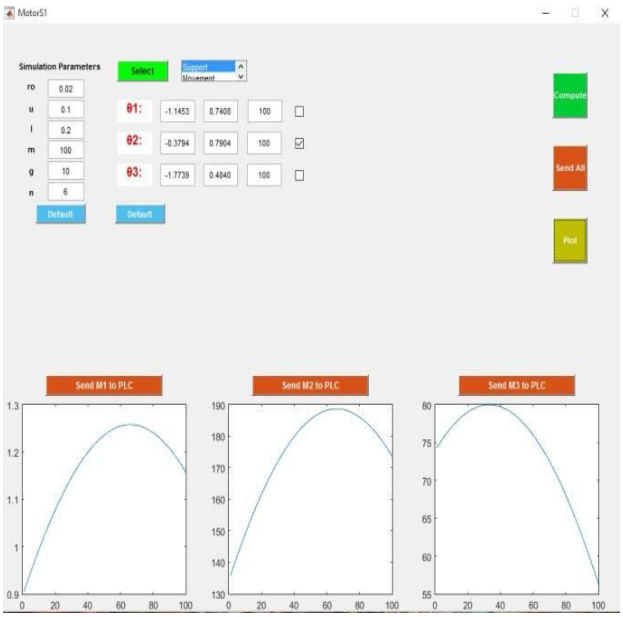

Figure 4: Load modelling interface

For the support phase:

$$M_{B1} = M_1 = \frac{\rho * \mu * q * G^*}{h}$$

$$M_{B2} = M_2 = q * G^*$$

$$M_C = M_3 = l_1 * \cos(\theta_2) * G^*$$

For the walking phase:

$$M_{B1} = M_1 = \frac{q_{CM}}{q} * a * \sin(\theta_1^*) * (m_1 + m_2)$$

$$M_{B2} = M_2 = G_1 * \frac{l_1}{2} * c_2 + G_2(l_1 * c_2 + \frac{l_2}{2} * c_{23})$$

$$M_C = M_3 = G_2 * c_{23}$$

The upper left corner contains the simulation parameters, based on which the mathematical model is composed. These are:

ρ – (ro) – radius of the friction couple to axis

B1 μ – (u) – the friction coefficient

l – the robot leg characteristic length, based on which all

length elements are calculate:

$_1 = 2 *$; $_2 = 3 *$;$h = 1.5 *$

m – robot total mass, assumed to be equally dispersed on the number of legs

g – gravitational constant

n – number of robot legs.

All other parameters present in the calculations are obtained based on those described above, or are simulation constants. These simulation constants are exceptions, however, as the model and application are strongly parameterized, in order to allow approaching situations and equipment that is as varied as possible.

For a quick run-through of the application or for check runs, a standard set of values can be loaded by using the „*Default*" button. This will load the following data vector: ρ=0.02; μ=0.1; l=0.2; m=100; g=10; n=6. This button only modifies the existing values in the simulation parameter input fields. As will be mentioned when discussing the application code, the values can be modified until pressing the „*Select*" button, without a model being loaded for processing.

The central part of the application deals with the movement of each joint, for which there can be input the movement limits of the actuated angle (in radians), in the first two fields. The third field is used to specify the resolution of the walking space by inputting the number of steps of the loaded linear space. For each of the angles, there is the possibility of specifying whether its reference point changes, or it is maintained constant, by selecting the marking field to the right of each. For an actively actuated angle, the model will load a linear space of the number of points selected, between the specified limits. For a static angle, the mean of the two limits will be used, as a constant value. This group also allows a quick load of standard values by using its respective „*Default*" button.

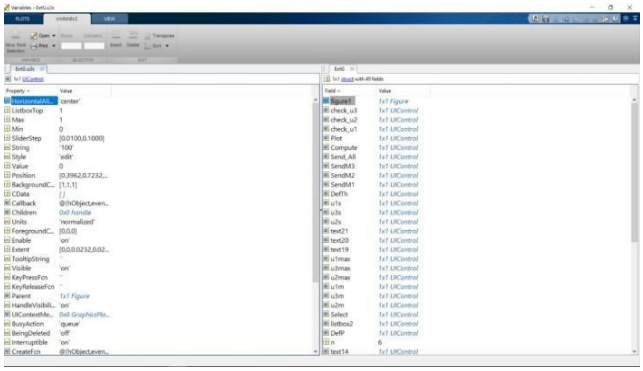

Figure 5: Variables in the workspace

Before running the application, there still needs to be specified the phase that the robot leg is in. The user can select one of the two options, „*Support*" and „*Movement*". As can be seen from the robot's mathematical model, the two situations are completely disjoint, and different calculations are made for each of them. After specifying the input parameters, the model is loaded into the runtime memory by pressing the „*Select*" button. If the application is run with the Matlab environment in the background, a new variable of type structure can be observed in the workspace, which contains all the described parameters. An example of such an occurrence is shown in Figure 5.

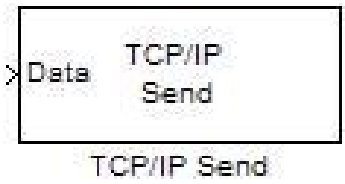

Figure 6: TCP/IP Simulink block

The three buttons above each axis system handle the communication between the load profiles obtained with the interface and another application which controls the PLC behaviour. The integration of the visual interface for load generation into the larger intelligent virtual environment platform is achieved through data transmission on a local area network. The desired outcome is that the various applications would reside on different computers, thereby increasing the available processing power and modularity of the platform. The communication is done through the TCP/IP protocol on an Ethernet connection. Each of the three buttons, as well as the initial test diagrams (see code structure for details), fire off intents to a TCP/IP transmission object in Matlab. The diagrams, running concurrently, simply replaced the final sink for each load profile (usually a Scope block) with a TCP/IP send block, like the one shown in Figure 6 [12].

4. APPLICATION DIAGRAM

The application diagram visually describes the user interaction and is shown in Figure 7. After loading the model at the previous step, the „*Compute*" button starts the calculation of the mathematical model discussed before. The user is notified through a dialog box when the numerical processing is done, at which time the processed data is made available.

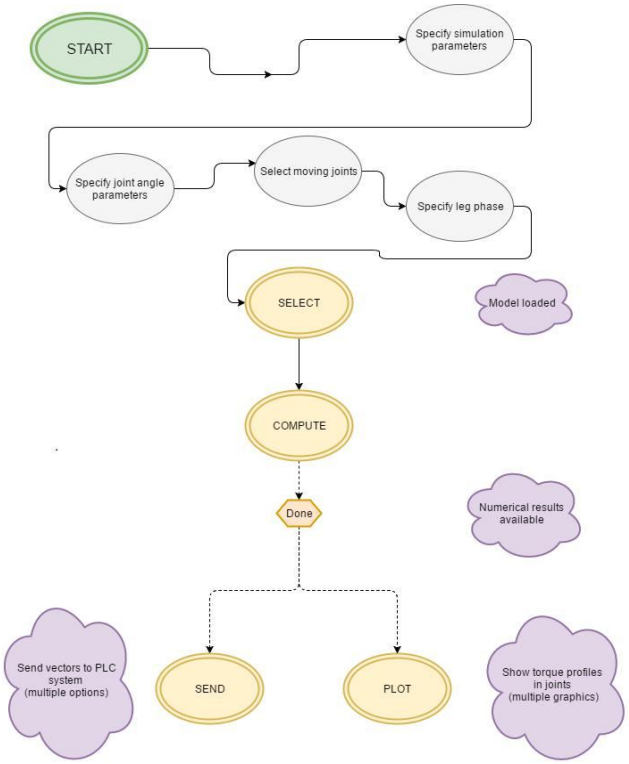

Figure 7: Application usage diagram

The interface now allows the visualization of the obtained data for each of the motor loads by pressing the „*Plot*" button, or sending these to the PLC control system by using the „*Send All*" button or the individual send commands, as desired.

The application diagram shows the expected one-time run-through for an untrained user. The process can be repeated and the parameters changed to obtain different sets of results. However, the application receiving the data from the interface will not be expecting a new set of incoming load or position profiles until its current run simulation is ended, so there is no option to send new data during the PLC runtime. This is by design as a safety feature, setting the destination listening port in the PLC application to only refresh between hardware simulations.

5. CODE STRUCTURE

In reviewing the code used in the application, the main standout are the Simulink diagrams used for visualization. The application actually uses coded functions written directly in the programming language, which execute the same operations, but obviously run much faster than a visual model.

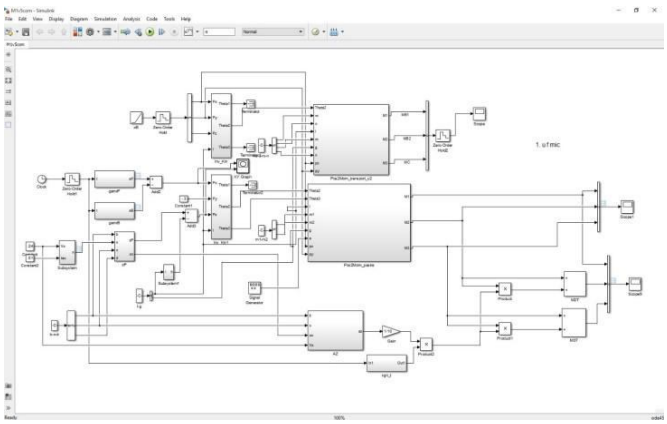

Figure 8: General structure of the load reference application

A diagram containing all of the application components is found in Figure 8. A noteworthy aspect is the multitude of inputs and pre-processing blocks, which help with model parameterization.

As can be seen from the figure, different sets of data are calculated, from different model components, depending on the selected state of the robot leg. In the visual model, these are shown in parallel, but in the application code they are only done once, for the selected state.

A snippet of code is shown in Figure 9. The functions within the graphical user interface are independent, using local variables, according to standard practice. Therefore, the „*handles*" structure is used to pass the saved or generated values between various functions inside the application.

Figures 10 and 11 shows the visual disposition of operations for the support and walking phase, respectively. There can be noted the equations used to calculate the model. The two diagrams are contained in the main subsystems visible in Figure 8.

```
u1=handles.u1;
u2=handles.u2;
u3=handles.u3;

ro=handles.ro;
u=handles.u;

l1=handles.l*2;
l2=handles.l*3;
h=handles.l*1.5;

Gstar=handles.m*handles.g/(handles.n-1);
px=cos(u1)*cos(u2+u3)*l2+cos(u1)*cos(u2)*l1;
py=sin(u1)*cos(u2+u3)*l1+sin(u1)*cos(u2)*l1;
q=sqrt(px.^2+py.^2);

handles.M1=ro*u*q*Gstar/h;
handles.M2=q*Gstar;
handles.M3=l1*cos(u2)*Gstar;

 % Save the handles structure.
guidata(hObject,handles)
msgbox('Done! ;)')
```

Figure 9: Example code in the callback function of the „Compute" button

The third main subsystem present in Figure 8 is responsible for adjusting the load value due to present acceleration on the Z axis, for a leg in the walking phase. This is done as follows:

$$A_Z = -\frac{b}{c * u} * \frac{\chi^2}{\sqrt{u}} * V_x$$

$$\text{where } u = c^2 - \chi^2.$$

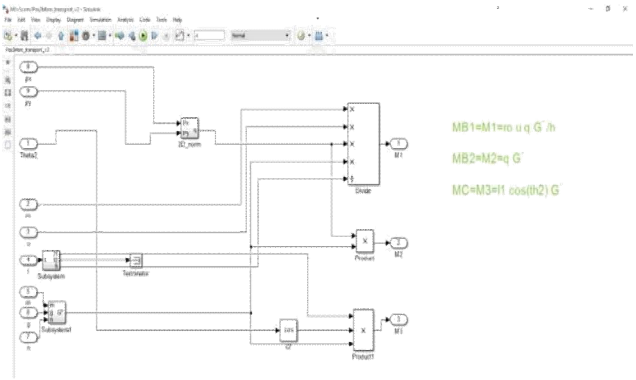

Figure 10: Diagram structure for load calculation in support phase

These are further used in:

$$M_2^{nou} = M_2 + M_2 * \frac{A_Z}{10} * sgn(t)$$

$$M_3^{nou} = M_3 + M_3 * \frac{A_Z}{10} * sgn(t)$$

$$\text{where: } sgn(t) = \begin{cases} 1, 0 \le t \le \frac{T_{sim}}{2} \\ -1, \frac{T_{sim}}{2} \le t \le T_{sim} \end{cases}$$

With this final adjustment step, the load profiles can be returned to the interface for plotting or sending to the PLC control component of the platform.

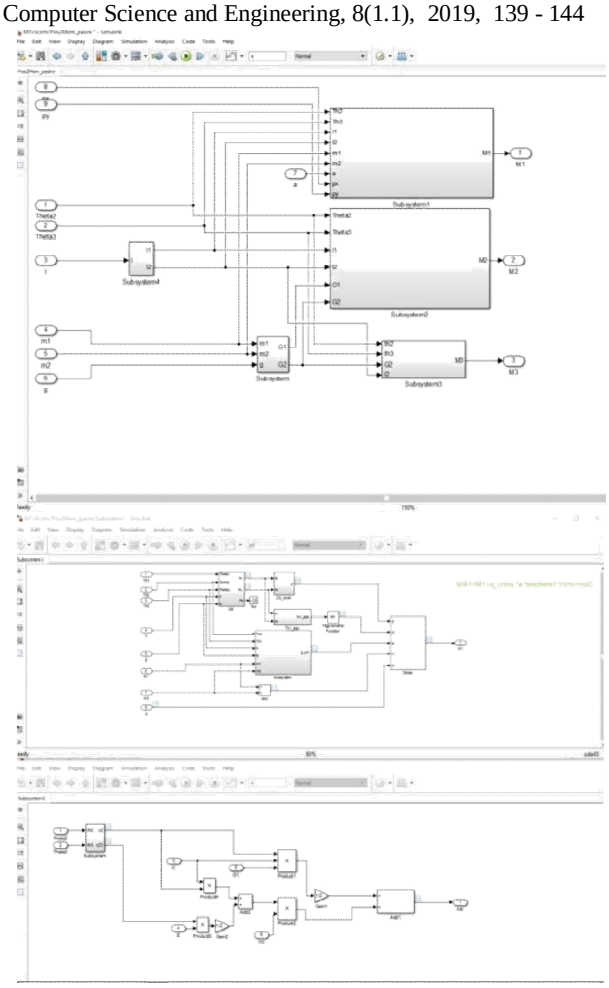

Figure 11: Diagram structure for load calculation in walking phase

The final results look similar to those shown in Section II, for the default dataset. As regards the transmission coding, they are n-dimensional vectors, with the timestamp being implicitly calculated from the simulation time-step, which is a known variable set by the user. The diagrams also include Zero-order_Hold blocks to illustrate the concept.

6. CONCLUSION

The paper presents the design and strategies for modelling a robotic kinematic chain using coupled actuators driven from PLCs on a testing stand. This entails the calculation of dynamic resistive loads within the proposed virtual environment, which are duplicated in coupled actuators controlled in torque. Once this model is fully operational, it can be used to safely test any type of control and motion strategies on a virtual robot system, while maintaining a solid base in concrete hardware implementation.

As can be seen from the described process of carrying out the modelling and mathematical simulations, the end-result is well parameterized and can be scaled to a wide variety of robotic

applications, as well as various environment and kinematic chain situations.

This is part of a larger effort to completely define a virtual environment for the simulation and testing of mechatronic systems on a remote virtual platform, encompassing all the usual and innovative aspects in the field of Robotics research, from low-level actuator control and mechanism design to intelligent operational strategies [10] and environment configuration modelling. It has the advantage of allowing virtually all manner of testing to be made remotely, with little or no extra configuration cost, while reducing the risk of equipment damage and maintaining the realism and end-result application value that can only come with actual hardware testing. This approach combines the best features of both scientific lines of enquiry, software simulation and direct hardware implementation.

ACKNOWLEDGMENT

"Thiswork was developed with the support of MEN-UEFISCDI, PN-II-PT-PCCA-2013-4, VIPRO project No.009/2014, Romanian Academy, "Joint Laboratory of Intelligent Rehabilitation Robot" collaborative research agreement between Yanshan University, China and Romanian Academy by IMSAR, RO by China Science and Technical Assistance Project for Developing Countries (KY201501009) and National Construction High Level University Government-Sponsored Graduate Student Project (Certificate No. 201608130106)."

REFERENCES

[1] Vladareanu, L Curaj, A ; Munteanu, RI, Complex Walking Robot Kinematics Analysis and PLC Multi-Tasking Control, Revue Roumaine Des Sciences Techniques-Serie Electrotechnique et Energetique, Volume: 57 Issue: 1 Pages: 90-99, WOS:000303096800010, ISSN: 0035-4066, (2012).

[2] Vladareanu, Luige; Sandru, Ovidiu I.; Velea, Lucian M.; Yu,Yu, Hongnian), Actuator Control In Continuous Flux Using Winer Filters, Proceedings of The Romanian Academy Series A-Mathematics Physics Technical Sciences Information Science, Volume:10 Issue: 1 Pages: 81-90 Published: JAN-APR 2009, WOS:000264992200011, ISSN: 1454-9069.

[3] Robin R. Murphy, "Human-Robot Interaction in Rescue Robotics", IEEE Trans on Systems, Man, and Cybernetics, Part C, vol.34, no. 2, pp.138-158.

[4] Victor Vladareanua, Radu I. Munteanub, Ali Mumtazc, Florentin Smarandached and Luige Vladareanua , "The optimization of intelligent control interfaces using Versatile Intelligent Portable Robot Platform", Procedia Computer Science 65 (2015): 225 – 232, ELSEVIER, www.sciencedirect.com, doi:10.1016/j.procs.2015.09.115.

[5] Vladareanu, L., Tont, G., Ion, I., Munteanu, M. S., Mitroi, D., Walking Robots Dynamic Control Systems on an Uneven Terrain, Advances in Electrical and Computer Engineering, Volume: 10 Issue: 2 Pages: 145-152 DOI: 10.4316/AECE.2010.02026 (2010), ISSN: 1582-7445.

[6] Wang HB, Zhang D & Lu H, Active training research of a lower limb rehabilitation robot based on constrained trajectory. International Conference on Advanced Mechatronic Systems, PP.24-29, 2015.

[7] Y.F. Feng, H.B. Wang, T.T. Lu, et al. "Teaching training method of a lower limb rehabilitation robot." Int J Adv Robot Syst, vol. 13, pp. 1-11, February 2016.

[8] Smarandache F., Vladareanu L., "Applications of Neutrosophic Logic to Robotics - An Introduction", The 2011 IEEE International Conference on Granular Computing Kaohsiung, Taiwan, Nov. 8-10, 2011, pp. 607-612, ISBN 978-1-4577-0370-6

[9] Şandru Ovidiu Ilie, LuigeVladareanu, Paul Şchiopu, Victor Vlădareanu, Alexandra Şandru, "Multidimensional Extenics Theory", U.P.B. Sci. Bull., Series A, Vol. 75 Iss. 1, 2013, pg.3-12, ISSN 1223-7027

[10] Nicolae Pop, Luige Vladareanu, Ileana Nicoleta Popescu, Constantin Ghiţă, Ionel Alexandru Gal, Shuang Cang, Hongnian Yu, Vasile Bratu, Mingcong Deng, A numerical dynamic behaviour model for 3D contact problems with friction, Computational Materials Science, Vol. 94, pp. 285-291, DOI: 10.1016/j.commatsci.2014.05.072 Published: NOV 2014, WOS:000342360000034, ISSN: 0927-0256, eISSN: 1879-0801

[11] Cononovici, S.B., Vlădăreanu, V., et al. "Control Strategies for Synchronous Generation of Resistive Loads Correlated to the Robot Motion Environment", 5th International Workshop on Cyber Physical Systems (IWoCPS-2016), Romanian Academy, Bucharest.

[12] https://www.mathworks.com/help/instrument/tcpipsend.html