Simultaneous navigation and fault detection of legged robot

Anton Vitko[#1], Ladislav Jurišica[#3], František Duchoň[#3], Andrej Babinec[#4], Martin Dekan[#5], Dušan Kaštan[#6], Marian Kľúčik[#7], Michal Bachratý[#8]

[#]*Institute of Control and Industrial Informatics, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava*

Ilkovičova 3, Bratislava, Slovakia

[1]anton.vitko@stuba.sk

[2]ladislav.jurisica@stuba.sk

[3]frantisek.duchon@stuba.sk

[4]andrej.babinec@stuba.sk

[5]martin.dekan@stuba.sk

[6]kastan.dusan@gmail.com

[7]marian.klucik@stuba.sk

[8]michal.bachraty@stuba.sk

***Abstract*— The paper describes results obtained in the development of adaptive fuzzy-neural navigation subsystem for mobile legged robot. In order to keep the motion sufficiently smooth, free of sharp turnings and transversal swings when moving between closely located obstacles, the fuzzy rules are updated on line. To this end the fuzzy rules are expressed through a layered feed-forward neural network and parameters and parameters in two steps – rough and fine updating. That is followed by the description of the learning fault diagnosis using binary neural network based on the Carpenter and Grosbergs' adaptive resonance theory.**

***Keywords*— navigation, mobile robots, fuzzy – neural system**

I. INTRODUCTION

Defined as a process of reaching a distant goal location, the navigation is a primordial task for any mobile robot. But there are significant differences between the wheeled and legged robots. As usual, task of the legged robot is not to move in office-like environment or smooth roads on which car-like vehicles run, but in irregular and unstructured terrains which are found in the natural environment. This fact implies differences in many aspects including the kind of information to be processed.

Restricted locomotion capabilities of the wheeled robots are sufficient in structured scenarios in where the ground is sufficiently flat. Examples are straight corridors, right angle corners with marks on the floor, standard door appearance etc. That is why the wheeled robot navigation can do with simple contact sensors, sonar and infrared rangers and so on. Contrary to that the legged robots are expected to walk on irregular terrain.

In comparison with wheeled robots, the legged robot moving in a harsh natural terrain calls for flexible locomotion system and intelligent control system. Besides the robotic he system should be able to cope with uncertainties and unforeseen failures, which can occur in mechanical construction of the legs as well as faults or malfunctions of the sensor and communication system. Therefore our attention is focused on intelligent navigation using both soft computing learning strategies and fault identification in order to secure sufficient level of an autonomous operability.

The control community is familiar with the term of "intelligent control", denoting the abilities the conventional control system cannot attain. Leaving alone the general meaning of the concept, it would be useful to single out some basic features that could be used for characterizing an intelligent system. Intelligent control was linked with the features that were traditionally out of the scope of specialists in conventional control systems. These are mainly the abilities of making decisions, adapt to new and uncertain media, self-organization, planning, image recognition, and more. Intelligent systems should not be restricted to those that are based on a particular constituent of soft computing techniques (fuzzy logic, neural networks, genetic algorithms and probabilistic reasoning), as is frequently done. Soft computing techniques should be considered as mere building blocks or even "bricks" used for building up a "large house" of an intelligent system. What makes today's systems intelligent is just a synergic use of these techniques, which in time and space invoke, optimize and fuse elementary behaviors into an overall system behaviour. For instance, fuzzy inference is a computing framework based on the fuzzy reasoning. But as to the fuzzy system is not able to learn, a neural network must provide its learning ability. To this end, the fuzzy rule-set is commonly arranged into a special neural architecture like ANFIS and NEFCON with Takagi-Sugeno-Kang and Mamdani inference respectively. [1]

Intelligence of neuro-fuzzy systems springs from successive generalization of the information chunks (granules) from singular ones, through crisp granular, to fuzzy granular information. An inferential process then runs over (overlapping) information granules. Due to the information granularization a system becomes robust with respect to imprecision, uncertainties, and partial truth. Thus, the system intelligence comes from the system architecture i.e. from an inner organization of the both system elements and functionalities. To demonstrate this, let us look at the *subsumption architecture*, developed in 1986 by Brooks [2] and used also in the design of navigation algorithm of our mobile robot. The subsumption architecture was inspired by the behavior of living creatures and, it is worth saying that, it heralded a fundamentally new approach to achieving more intelligent robots. In this architecture the robot behaviour is typically broken down into a set of simpler behaviours that are loosely co-ordinated towards a final goal in a sense, that every single behaviour selectively assume the control of all subsumed behaviours. The behaviours with higher priorities are subsumed under those with lower priorities; hence a layered structure is developed. The layer (i.e. a set of behaviours of the same priority) with higher priority can inhibit or even supersede those having assigned lower priorities.

II. THE NAVIGATION

Within the development of the navigation algorithm it was supposed that the robot is equipped with an ultrasonic ranger which rotates and scans the environment around, providing information about the distance and azimuthal angle of the nearest obstacle. The output signals (angle and size step) control the robot to turn left or right and to modify its speed. The navigation is exclusively of reactive character. It doesn't need any environmental map.

In order to keep the motion sufficiently smooth, free of sharp turnings and transversal swings when moving between obstacles, the parameters of fuzzy rules are updated on-line. It is done periodically in two steps for each period. Within the first step takes place the tuning of rectangular membership functions (MF). To this end the fuzzy rules are updated using algorithms of unsupervised learning within which a cost function is evaluated. The cost function is chosen in such a way that its minimal value should prevent the robot from possible overthrowing due to high speed along a bend path. That conception allows us to flexibly change the radius of the curved path and thus to account for instantaneous dynamic conditions during motion (this aspect has not been included in this paper).

The fine tuning of MF's takes place within the second step. To be more specific, normally straight walls of the trapezoidal MF's are deformed into appropriate irregular shapes. This is done with the aim to reach yet smaller value of the cost function. To avoid the unnecessary reduction of the robot speed the two steps should be repeated with high frequency, which is derived from actual speed of the robot. Simulation results have showed that the described learning philosophy conception is feasible. Besides, it also prevents the robot from intensive transversal swings which are natural in a purely reactive navigation. The fuzzy system can mimic the human reasoning, and possesses human tolerance for incompleteness, uncertainty, imprecision. As a means of modelling the decision a fuzzy model, comprising 24 fuzzy rules was used. Typical structure of the fuzzy rules used is:

IF (obstacle is middle) AND (distance is near) AND (target is right) THEN (turn is right)

The premise parts are connected by AND function of the three input variables, namely:

-"obstacle", means the azimuthal angle of the nearest obstacle

-"distance", means the distance from the robot to the nearest obstacle

-"target", means the azimuthal angle of the target

Outputs of the neuro-fuzzy engine are:

-"turn"- turning angle by which the mobile platform is requested to turn in order to avoid the nearest obstacle

-"step"- size of the step to be done in requested direction. Fig.1

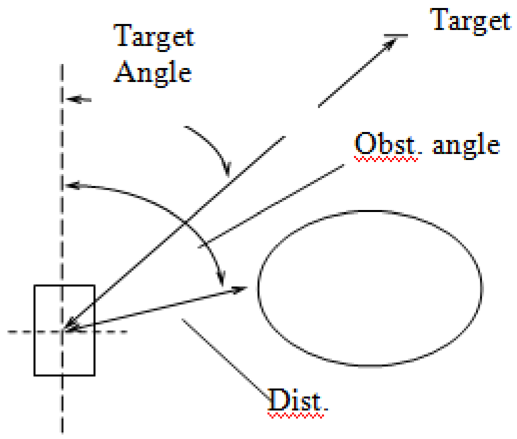

Fig. 1 Definition of input variables

The size of the step is reduced as the robot approaches either the obstacle or the target. The antecedent parts are evaluated through fuzzy reasoning which is based on Min-Max composition rule for fuzzy AND and OR operators respectively. For conversion of the fuzzy set outputs to corresponding crisp was used the bisector method .

As a measure of the closeness of the actual radius to the desired one was used the function

$$E = (r^d - r)^2 = \left(r^d - \frac{\sqrt{a^2 + c^2 + 2ac\cos\alpha}}{2\sin\alpha} \right)^2 \qquad (1)$$

where r is an actual radius of the robot's path curvature a and c are two consecutive step seizes with α being an angle between their directions. Finally r^d denotes a desired (meaning maximum allowable) radius of the robot yawing. Such arrangement allows for optimization of the radius *r* with respect to the step sizes *a* or *c* and turning angle α. The error signal for the NN output node can be computed directly. For the particular angle α^* obtained the adaptation error ε is computed in accord with by (2)

$$\varepsilon = \left(\frac{\partial \ E}{\partial \alpha} \right)_{\alpha^*} \qquad (2)$$

The developed navigation system was verified by both simulation and real experiment. Results of the real verification are depicted in Fig 2. Crosses represent the borders of the obstacles as identified by sonar sensor. The path of the robot movement is represented by circles.

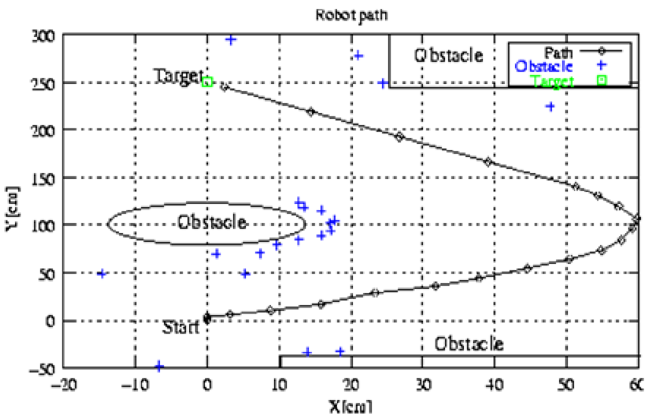

Fig. 2 The map constructed by the mobile robot

III. Neural Clustering and Classification (A Case Study)

Due to the extensive use of complex mechanical components like arms, legs, actuators, gears, clutches, grippers etc., the robot's mechanical parts suffer from significantly higher fault rates than pure electric and electronic circuitry. Potential faults should be detected sufficiently soon so as not to avoid a fatal failure. In other words, the system should anticipate possible faults on the basis of their pathologic behaviours. Therefore a novelty detection mechanism is necessary. Imminent failures are often manifested through the declined values of system parameters and variables or their fused complexes. An idea is to identify any deviation from normal behaviour. The component degradation, like wear, increased friction, stiction due to contamination, corrosion etc., is related to an observable effect on the system performance (higher vibrations, increased friction, decreased positioning precision etc). Such relationships may change as the degradation progresses.

Neural based classifiers are today the most powerful means due to their learning ability. They can classify even noisy and sparsely populated sets of measured values. In essence, practically any kind of neural network can be used for fault classification. The NN classifiers make weaker assumptions concerning the shape of statistical distribution of the input patterns in comparison with e.g Kalman filter. Another motivation is the need to detect new and unexpected faults (problem of novelty detection). This can be achieved by unsupervised learning. A serious problem with NN classification is that, in real situations, the problem domain does not always behave well. For instance, if some unexpected and strongly different input patterns appear, in the most NN there is no built-in mechanism for recognizing the novelty. Simply said, the NN should preserve previously learned patterns (stability) while keeping its ability to learn new patterns (plasticity). This phenomenon is known as *stability-plasticity dilemma*.

An elegant solution to the problem of stability-plasticity provides a family of the neural networks based on the "*adaptive resonance theory"* (ART), developed by Grossberg and Carpenter [3]. The ART family of self-organizing networks with competitive learning comprises network architectures, which are able to cluster input patterns based on a given measure of similarity. In particular, the ART1 network which was used in the experiment, allows for incremental learning of prototypes, rather then instantaneous input exemplars. This is because the whole cluster of similar inputs is updated using information from input patterns and therefore preserves main features of already accepted input patterns.

IV. Results of Classification

Efficiency of the developed neural classifier was verified by simulation as well as by experimentation with the developed legged robot. The simplest and most evident faults like those related to control sequences controlling the movement of joints and legs or the faults appearing during switching between robot gaits were easily detected and classified by using the deterministic final-state machine, developed for this purpose. It was possible due to the fact that such faults manifest themselves through the total fallouts of particular sensor signals.

Contrary to this, more complex faults may be caused by increased friction in bearings, slipping or dragging clutches, lack of lubrication or a partial loss of energy delivery to a particular joint. Finally, there could be the faults caused by incorrect coordination of legs due to improper timing (fall out of phase or fall out of step and the like). Malfunctions of this kind may remain hidden for longer time and may gradually lead to fatal failures, like the total destruction of bearings or drives, lagging legs movement, which could jeopardize the walking stability or even cause instability of the robot. Such faults are commonly manifested through abnormal trajectories of the joint torques or forces. Therefore, the learning neural classifier was designed just for the task of detection and classification of any abnormal joint torques. In order to teach the neural network to classify abnormal torques, the leg dynamics were simulated in Toolbox SIMMECHANICS (a part of Simulink toolbox in MATLAB, oriented towards simulation of mechanical systems, including actuators and sensors).

The leg can be either in a stance state, when it supports the robot body or in a swing state, when it moves in air to the position where it can begin a new stance. A time-course of the normal (faultless) torque exerted in a femur joint is shown in Fig. 3. One complete step cycle is performed in three phases, each lasting one second. As seen from the figure, these three phases can be easily observed from a torque-time dependence. Particular phases are supplemented with a simple imbedded sub-figure depicting the leg configuration what corresponds to the phase. During the first phase the leg remains in a flexed configuration in the stance. The femur joint exerts the torque value about 30 Nm, which maintains an attitude of the robot body.

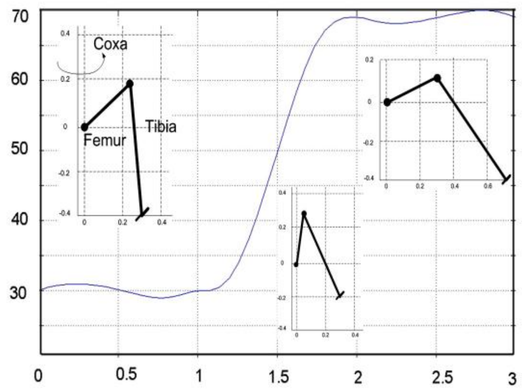

Fig. 3 Normal torque in the femur joint

The second phase starts at one second. The leg is uncoupled from the ground and starts its swing movement in a direction of walking. While the torque exerted in the femur joint causes raising the leg, the coxa joint is rotating the leg about the vertical axis and the tibia joint is extending the leg. When reaching the highest position and maximum extension the leg ends its second phase. At this time instant the femur joint exerts maximum torque of about 70Nm. Just after the third second the femur torque slightly decreases so as to make the foot go down until it reaches the ground. At this moment (at about the fourth second) the leg is entering into its stance state again, and supports the robot body.

During learning, the neural network ART1 is first taught to learn the normal torque. As a result, the neural network appoints the normal torque course as the centre of a receptive field of the cluster of all "approximately normal"torque courses (torque patterns). This is done by adaptation of the bottom-up weights leading to most left neuron in the layer F_2. From this time on the unit value of this neuron will indicated that the current input belongs to the From this time on the unit value of this neuron will indicated that the current input belongs to the cluster of "approximately normal" torque courses and this cluster will represent a class of normal torque courses. Then a training list, i.e. a series of faulty torque patterns, generated by Simmechanics Toolbox, is repeatedly presented. The experimental results have shown that the learning task could be considered accomplished (the weights reach their steady values), after presentation about 5 or 6 epochs. After learning the neural network becomes able to classify abnormal torques. Results of classification shown feasibility of the described design will be presented at the conference.

ACKNOWLEDGMENT

Support of the grant VEGA 1/0177/11 is duly acknowledged.

REFERENCES

[1] J. S. R. Jang, E. C. T. Sun and E. Mituzami, *Neuro-fuzzy and soft Computing*, 1st ed., Prentice Hall, Inc. Upper Saddle River, 1997.

[2] R. A. Brooks, *Achieving Artificial Intelligence Through Building Robots*, AI Memo 899, MIT, Art. Intelligence Lab., 1996.

[3] G. A. Carpenter and S. Grossberg, *A massively parallel Architecture for a Self-organizing Neural Pattern Recognition Machine*, Comp. Vision, Graphics and Image Proc., vol. 37, pp. 54-115, 1987.