Transactions of the TSME (2016) Vol. 4, No. 1, 46-57
Journal of Research and Applications in Mechanical Engineering

ISSN 2229-2152 print
DOI:

Research Article

PSO-BASED LEG-LOSS IDENTIFICATION METHOD FOR LEGGED ROBOTS

S. Chattunyakit[1,*]
Y. Kobayashi[2]
T. Emaru[2]
[1] *Division of Human Mechanical Systems and Design, Graduate School of Engineering, Hokkaido University, N13W8, Kita-ku, Sapporo, Hokkaido, 060-8628, Japan*
[2] *Division of Human Mechanical Systems and Design, Faculty of Engineering, Hokkaido University, N13W8, Kita-ku, Sapporo, Hokkaido, 060-8628, Japan*

ABSTRACT:
Legged robots have been widely developed and utilized in various applications since they are more flexible than the conventional wheel-based robots which cannot perform effectively in bumpy areas. Although the legged robot has no difficulty operating in uneven terrain, the broken parts of legged robot can lead to the task failure. In case of damages, the legged robots cannot operate properly with prior control strategies due to transformed models. This paper proposes the new method to detect the broken legs by employing only internal sensors. The lengths of robot legs will be estimated using the comparison between damaged robot and candidate models constructed in the simulation. Particle Swarm Optimization (PSO) is operated to discover the best candidate model that provides the highest fitness value. The similarity of orientation of robot body between actual damaged robot and candidate models is set as the fitness function calculated using normalized cross-correlation algorithm. The efficiency of this method is verified using numerical simulations and experiments, which shows that the proposed method can detect the lengths of robot legs more accurate than the existing method.

Keywords: *Fault detection, self-identification, normalized cross-correlation*

1. INTRODUCTION

Legged robots can be applied in several applications (e.g. being in action on uneven terrain and outdoor environment) on account of flexibility in which regular wheel-based robots are not attainable. However, intricate control manners and sophisticated components are necessary to achieve the desired behaviors. In general, legged robots can function successfully with predesigned controllers. However, there are some failures occurred when some parts of robots are not working, such as broken legs and joints lock. Damaged legs can lead to the lack of success while operating with the prior-designed controller because of changed models. Recently, self-damage recovery algorithms have been proposed so far to overcome this drawback. It makes robot more flexible as allowed the robots to create the new-alternative behaviors to deal with broken robots automatically without human control. Human and certain animals, in like manner, will be able to learn the new gait when their legs get injured. A biped dog can create a distinctive self-recovery behavior with only two hind legs as it can do balancing, standing and even walking. Thus, it can be recapitulated that applying self-damage recovery behavior can open doors of possibility for damaged-legged robots to restore themselves.

The insect-inspired central pattern generators are employed in a robot with leg malfunction successfully [1]. The controlling frequency of each leg are changed to compensate the leg malfunction. In the experiment, legs of the

* Corresponding author: S. Chattunyakit
E-mail address: mr.sarun.ch@gmail.com

robot are disable in order to mimic the leg malfunction. However, the broken legs, joint and body have not been considered yet in [1]. In [2], the six-legged robot will be able to walk after getting damaged; still, it cannot cope some imperfections owing to the non-updated model. Leg-loss identification is a significant process to assist the damaged robots to discover the new, present models. The camera equipped on the robot is employed to detect the artificial markers mounted on robot's legs [3]. This method works suitably to detect the abnormal actions of leg loss; yet, there are some conditions of using camera and image processing that lead to task failures. For example, intensity of light or dusty environment can affect a detection system. Although the artificial markers aid in recognizing position of legs faster and easier, predesigned markers are needed to attach on robot's legs in advance. Some researchers also detect the abnormal joints of legged robot by measuring the electrical current spent while operating [4]. The advantage of this method is that external sensing measurement is not employed which make it more flexible in practical situation. However, the errors might occur with low battery as measuring electrical current incorrectly. Another method, named as virtual joint sensor, can detect the joint sensor fault on humanoid robot [5]. This method uses the linear invert pendulum model and the leg kinematic models cooperating with the Kalman filter to detect and recovery the fault of joint sensors. Although the simulation results prove that this method can detect the fault of joint sensor effectively, it can detect only the fault of joint sensor. Acceleration data is applied to identify the fault as well [6]. It is used to measure the orientation of robot and set as the input of the state machine. This method can detect only the abnormal behaviors of the robot.

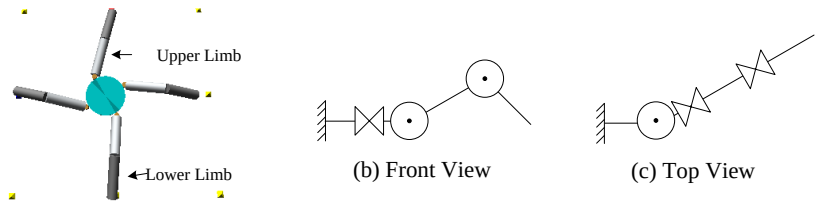
In spite of the fact that aforementioned methods can correctly detect the broken parts or abnormal behavior, the new-model of robot is not provided. Since the method proposed in [3] can identify only the status of each leg (usable or unusable) and the method proposed in [4] can detect the joint fault, they cannot provide actual updated model after damaged. Bongard and et al. published the self-modeling method that lets the robot be able to reconstruct an exact model [7]. Robot can discover the current model using collation of responses between the actual robot and candidate models in the simulation. The candidate models in the search space will evolve their body in which makes their behaviors similarly to real-damaged robot. The optimization method, genetic algorithm (GA), is utilized for this process. The responses of damaged robot are observed using external sensory device, the Wii infrared remote. After applying this approach, the new proper model and alternative behaviors have been found. However, using an external sensor limits the ability of legged robot to perform in other different environments. To eliminate this problem, the algorithm proposed in [8] employs only built-in sensors, i.e., tilt sensor and switches, to observe its behavior, called as self-identification method based on biological evolutionary mechanisms. The damaged robot can obtain the new model successfully with this method. In addition, this method is applied to get alternative gaits in [9]. However, GA can provide resolution errors as encoding and decoding processes are required as dealing with binary chromosomes. Moreover, due to a limited number of sensors, some candidate models with different leg's length can yield the duplicate final postures at the end of process so that the candidate models in search space have to perform twice to overcome this problem.

The PSO-based Leg-loss Identification method (PLI) is proposed in this paper. The PLI method uses only on-board sensors that lets robot become more versatile. Particle swarm optimization is utilized to optimize the fitness function that is set as the resemblance of candidate models and actual damaged robot. The advantage of PSO is that parameters can be set as the real number, meaning that decoding and encoding process (using in general Genetic algorithm) are not essential. Since using only final posture can cause the confusion of model identification, it is better to observe the behavior of robot during a period of time. The comparison between actual-damaged robot and candidate models in the simulation is done using normalized cross-correlation algorithm. The performance of PLI is evaluated using the Open Dynamics Engine™ (ODE) simulator. In the benchmark, the efficiency of PLI is compared with existing GA-based identification method [8]. The results show that the proposed method can assist legged robots to discover the new leg's lengths better than GA-based method in term of accuracy. Moreover, two experiments are conducted with 2-DOFs quadruped robot to apply this method in practical situation. According to the result, the PLI algorithm can provide the acceptable new-updated model so that the proposed method can employ with actual robot as well. However, the gap between real world and simulation slightly limit the performance of algorithm.

2. ROBOT MODEL AND SIMULATION

The quadruped robot, consisting of twelve servo motors (three joints for each leg), is employed to evaluate the proposed algorithm in this paper. The attended mode of servo motors is shown in Fig. 1. There are two limbs for one leg, upper limb and lower limb. The lower limb and upper limb are connected together with the hinge joint, but

the body and the upper limb are connected with two hinge join, as shown in Fig. 1(b) and (c). Both upper and lower limbs are constructed as the cylinders with 2 cm in radius and 20 cm in length. The shape of the robots' body is also in the cylinder shape with 30 cm diameter. To measure the orientation, used as the main cue for the PLI, the Inertial Measurement Unit (IMU) is mounted on the robot body. The Open Dynamic Engine™ (ODE) is utilized in order to simulate and evaluate the performance of proposed method.



(a) Robot Model (b) Front View (c) Top View

Fig. 1. The robot model used in the simulation: (a) reference quadruped robot, (b) joints construction in front view and (c) joints construction in top view.

3. LEG-LOSS IDENTIFICATION ALGORITHM

Leg-loss identification method is beneficial for legged robots in order to deal with the damaged legs. This paper mainly focuses on identifying the length of broken legs. In the PLI method, the normalized cross correlation and PSO are employed to evaluate the differences between damaged robot and candidate model and to discover the candidate model providing similar behavior to actual damaged robot, respectively.

3.1 Normalized Cross-Correlation (NCC)

The cross correlation is utilized to compare the responses of damaged robot and candidate model. It is generally used in signal processing with the concept of convolution technique to compare between an input signal and a reference signal [10]. In PLI, the normalized cross-correlation is implemented as its result is allocated between +1 and -1. In case that the result is close to +1, it means that two signals are completely similar. On the other hands, if the result is close to -1, it means that two signal are exceedingly different. The cross-correlation between two real continuous signals x and y can be computed using Eq. (1),

$$C_{xy}(\tau) \equiv \int_{-\infty}^{\infty} x(t)y(t-\tau)dt \tag{1}$$

where the time shift τ is the *lag*. The discrete normalized cross-correlation of signal x and y over the period of time N is given as follows:

$$C_{norm} \equiv \frac{\sum_{n=0}^{N} x[n] \cdot y[n]}{\sqrt{\sum_{n=0}^{N} x^2[n] \cdot \sum_{n=0}^{N} y^2[n]}} \tag{2}$$

In PLI, as orientation is employed, there are three signals, i.e., pitch, roll and yaw angles of robot's body, to be considered. The NCC result is combined and averaged in one value as following equation:

$$cw = C_{norm-ave} = \frac{C_{pitch} + C_{roll} + C_{yaw}}{3} \tag{3}$$

where C_{pitch}, C_{roll} and C_{yaw} are the normalized cross-correlation of pitch, roll and yaw angles of robot's body, respectively. The value of cw can be varied from -1 to +1; hence, on condition that it allocates at 1, it means that the candidate model and broken robot have similar model (same behaviors). On the contrary, it indicates that candidate

model and broken robot have different model when *cw* is close to -1. In addition, the value *cw* will be manipulated in PSO algorithm in the next process.

3.2 Particle Swarm Optimization (PSO)

The PSO, inspired by swarm behavior of living creatures, is generally used in optimization problems. The solution of the problem is called as 'particle'. The concept of PSO can be described that all particle in the search space will follower the leader (highest fitness value) to find optimum point. In the search space, there are three main parameters, which are current position (*x*), best previous position (*p*) and current velocity (*v*). For *i*th particle in a D-dimension space, it can be set as $X_i = (x_{i1}, x_{i2}, ..., x_{iD})$, $P_i = (p_{i1}, p_{i2}, ..., p_{iD})$ and $V_i = (v_{i1}, v_{i2}, ..., v_{iD})$. The best prior positions of any particles in population are stored for updating their speeds. The evolution of each particle is varied by the following equations:

$$v_{id} = w \cdot v_{id} + c_1 \cdot Rand() \cdot (p_{id} - x_{id}) + c_2 \cdot Rand() \cdot (p_{gd} - x_{id}) \tag{4}$$

and

$$x_{id} = x_{id} + v_{id} \tag{5}$$

where *w* is the weight to balance between global search and local search, c_1 and c_2 are the positive constant weights, and $Rand()$ is a random function range 0 to 1. The velocity of each particle is updated by combining best local solution (p_{id}) and best global solution (p_{gd}). Eq. (4) and Eq. (5) are originally published in [11]. In PLI, the Eq. (5) is modified to improve the robustness as parameter *cw* is added into the equation as follows:

$$x_{id} = x_{id} + (1 - cw) \cdot v_{id} \tag{6}$$

The term $(1 - cw)$ is contributed the system to become convergent rapidly. The term $(1 - cw)$ become 0 when *cw* is +1, meaning that the updated model of damage robot is achieved (a candidate model can produce same behaviors as broken robot). Thus, it is not in need of updating the position (*x*).

3.3 PSO-based Leg-loss Identification (PLI)

The PLI algorithm can be applied to assist the legged robot to identify the new updated model after damaged. This method is able to get the length of robot's legs by means of comparing the responses of an actual robot and candidate models in the simulation. The NCC and PSO are further employed in PLI algorithm to obtain the similarity of behaviors and to evolve the candidate robots in the search space, consequently. The advantage of this algorithm is that only internal sensors are utilized. Thus, it leads the legged robot become feasible to operate in unfamiliar environments.

The procedure of PLI algorithm can be seen in Fig. 2. The algorithm begins with creating the *N* random particles in the population. Each particle contains eight real values – four legs (two parts per leg) – as the robot model illustrated in Fig. 1 (a). The value of particle (*p*), so-called as candidate model, of *n* can be represented as follows:

$$p_n = \{L_{n,1}, L_{n,2}, L_{n,3}, L_{n,4}, L_{n,5}, L_{n,6}, L_{n,7}, L_{n,8}\} \tag{7}$$

where *L* is the length of robot's legs and $n = 1, 2, 3, \ldots, N$. In primary state, it is assumed that only lower parts of legs are broken. The lower lengths of legs will be generated randomly with range of 0 to 60 cm, but the upper lengths will be set as *x* cm identically to the length of actual robot – the lower lengths should be constant since they have not been broken yet. After initializing the particles, all of candidate models and actual robot will perform similar action that is generated randomly beforehand. Later, the evaluation process begins. All of candidate models will be compared with damaged robot following Eqs. (1) and (2). If the best *cw* value of the candidate model reaches the desired value *m* (it is set as 0.95 in this study), the process of leg-loss identification will be completed, and select the best model as the new model for damaged robot. Moreover, if the iteration reaches the time *t*, this process also ends. In case that the model is not satisfied, the new set of candidate model will be generated according to the PSO algorithm – all of *L* in p_n will be updated following Eqs. (4)-(6), and then continue the same process as shown in

Fig. 2. This routine will be operated repeatedly until reaching the desired *cw* or set time *t*. Fig. 3 illustrates the best candidate model in search space that can adapt itself to provide same behavior as damaged robot (the lengths of upper limbs and lower limbs are 20 cm but the length of broken limb is set as 10 cm). By performing for 25 iterations, the robot can discover the new-updated model. Additionally, in evolution process, some of L values in p_n will be varied randomly with probability *prob* = 0.10, and the upper length of legs will not be changed if the lower length of legs are not zero.

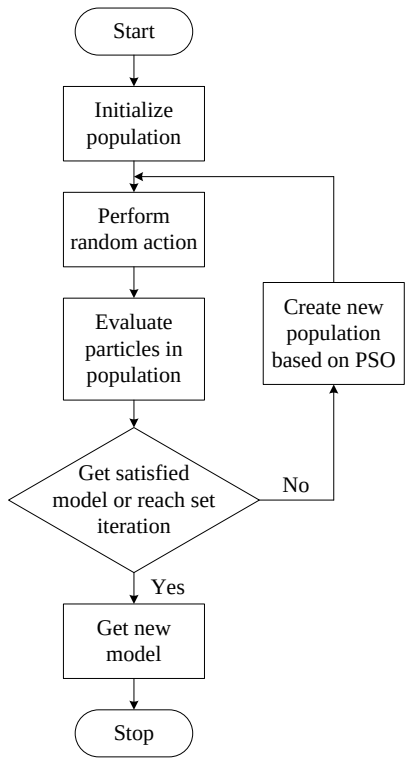


Fig. 2. Process of PLI algorithm.

4. SIMULATION RESULTS

The performance of proposed algorithm (PLI) is certified by numerical simulation, and compared with the GA-based method as it uses only internal sensor, likewise [7]. There are three of broken legs validated in the experiments, which are one leg-loss by half, one leg loss and two leg-loss, illustrated in Fig. 4. Two algorithm, PLI and GA-based method, are programmed to perform with the same sets of parameter as population of 80 particles and 25 iterations with 5 ms sampling rate. Each algorithm will perform 10 times, and the average of root mean square (RMS) error will be measured. The RMS error can be calculated as follows:

$$\text{RMS error} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2} \tag{8}$$

where $i = 1,2,\ldots,n$ and e is the error of lengths between the actual damaged robot and the new updated model obtained from the simulation. The performance of PLI is also measured by varying the number of particles in the population, and this experiment collects the number of iteration that the best models are discovered. In addition, the noise and sampling rate tests are conducted to evaluate the performance of proposed method.

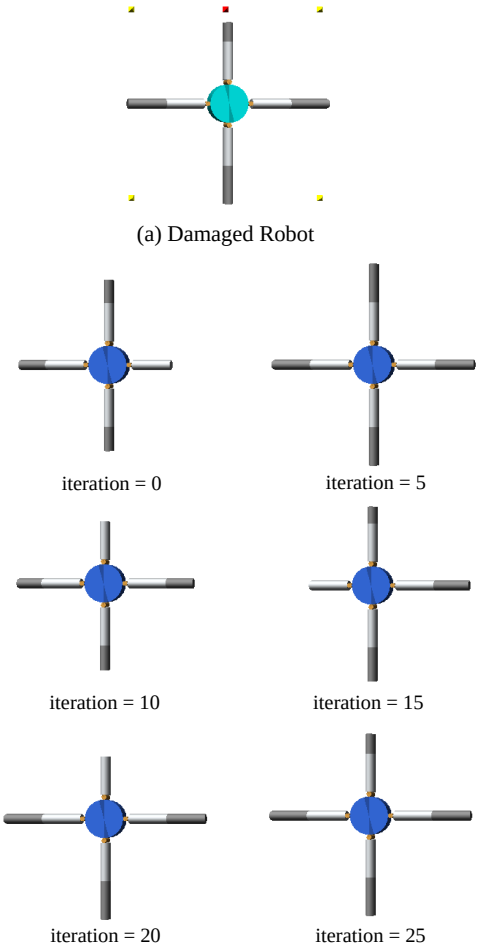


Fig. 3. The evolution of the best candidate robot in search space: (a) damaged robot and (b) the best candidate model captured every 5 iterations – shown that the candidate model can evolve itself in order to create the behaviour as same as damaged robot. In 20 iterations, it cannot find the proper model, but it can identify the length of legs correctly within 25 iterations.

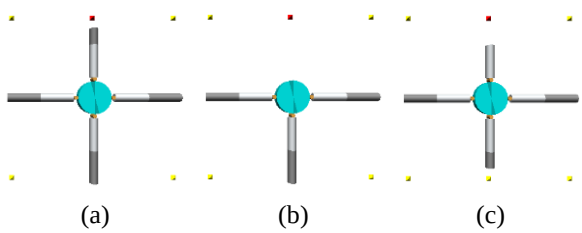


Fig. 4. The damaged robot tested in the experiment: (a) first lower limb-loss by half, (b) first leg-loss and (c) first lower limb-loss and third lower limb-loss by half.

4.1 One-leg loss by half (Case A)

In this test case, two algorithms are conducted to verify the broken robot with lower limb-loss by half, and the average RMS errors are measured. The example of result is shown in Table 1. The PLI can provide more accurate

results, compared to GA-based method, and the average RMS error is also lower as the RMS error of PLI and GA-based method are 0.094 and 9.157, subsequently. The main reason why GA-based method product more error is that it employs the switches mounted at the end of robot's legs. In case of damaged leg, the responses of the switch cannot be measured correctly.

Table 1: The length of robot's legs (in cm) from PLI and GA-based Method

Leg Number		Upper Limbs				Lower Limbs				RMS Error
		1st	2nd	3rd	4th	1st	2nd	3rd	4th	
Case A	*Actual*	40.000	40.000	40.000	40.000	20.000	40.000	40.000	40.000	-
	PLI	40.000	40.000	40.000	40.000	19.745	39.991	39.991	39.921	0.095
	GA	33.647	40.000	32.941	40.000	0.000	33.647	48.941	32.235	9.157
Case B	*Actual*	0.000	40.000	40.000	40.000	0.000	40.000	40.000	40.000	-
	PLI	0.000	40.000	40.000	40.000	0.4152	40.000	40.000	40.000	0.147
	GA	33.647	40.000	29.176	40.000	27.058	40.000	29.412	32.235	16.408
Case C	*Actual*	40.000	40.000	40.000	40.000	0.000	40.000	20.000	40.000	-
	PLI	39.720	40.000	40.000	40.000	0.079	39.609	19.739	39.940	0.196
	GA	40.000	41.882	32.941	40.000	1.647	40.000	24.235	37.882	3.133

4.2 One leg loss (Case B)

The PLI still performs well and provides good result in term of error, as illustrated in Table 1. Due to the same reason as aforementioned in 4.1, the GA-based method, further, provide big error while operating. Since this benchmark is simpler than the first one – as no shorten leg to make an impact movement, the error of GA-based method is less than the result in *Case A*, as 16.408. Still, the PLI can make a better result as RMS error is 0.147.

4.3 Two leg loss (Case C)

This is the most difficult test compared to others because there are two broken legs, one leg-loss by half and one lower leg-loss by half. The error of PLI algorithm is still lower than GA-based method because of the same reason. However, the GA-based method can provide feasible result as illustrated in Table 1. The RMS errors of PLI and GA-based Method are 0.196 and 3.133 by order.

4.4 Number of particles

To ensure that PLI algorithm can discover the new model correctly, the number of particles is varied to test the stability. The result of this test can be seen in Fig. 5. The number of particles is set at 40, 60 and 80 particles. The experiments are conducted for 30 times per each case, and the average numbers of iteration are shown in the graph. In *Case A*, the result shows that increasing number of particle leads the robot to find the new model rapidly. However, in *Case B*, a low number of particles can perform faster because this situation is simpler than other so that more number of possible solution can affect the system failure. In the most complex experiment, *Case C*, the results are therefore same as *Case A* since it is difficult to detect the correct model, a large number of feasible solutions will make the robot to get the correct model faster. Therefore, the number of particles can affect the performance of leg-loss identification in term of time spent. On the other hands, in non-complicated situation, a low number of particles can be employed to achieve better solution.

4.5 Noise Analysis

Since the PLI algorithm is experimented only in simulation, the noise effect is considered as well. This experiment is done with *Case A* with 5 ms sampling rate and 80 particles, and the Gaussian noise is added into the signal to simulate the noise of the accelerometer and gyroscope which are used in the experiment. However, the noise in the real system is not similar as simulated, but the aim of this test is to ensure that the PLI can perform with the noisy condition. The result shows that, with noise, the error of the system is bigger than that without noise situation; however, the error at the end of the simulation becomes smaller (acceptable result with 2 cm error) as shown in Fig.

6. It means than the PLI algorithm is feasible to apply in noisy systems. In additional, the PLI will be demonstrated with an actual damaged robot in the next section.

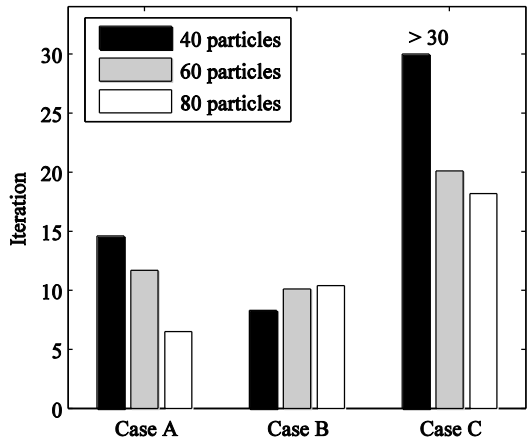


Fig. 5. Results of number of particles test (average iteration that robot discovered the best solution).

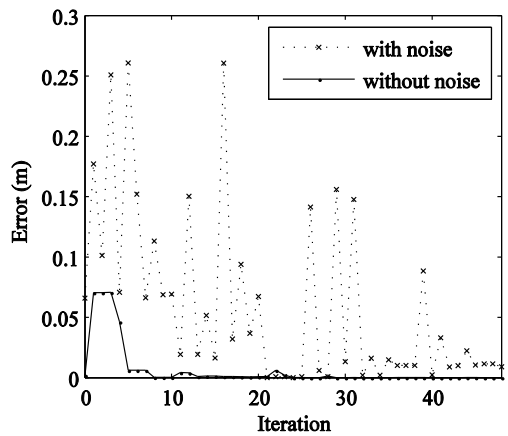


Fig. 6. Results of number of particles test (average iteration that robot discovered the best solution).

4.5 Sampling rate

The sampling rate of the signals are also tested in the experiment, and it has been evaluated with case A, without noise and 80 particles. As shown in Table 2, the result shows that varied sampling rates, 5, 10, 20 and 100 ms, do not cause the major errors. Although the large sampling rates have not been tested in the experiments, in general robotics system, the 100 ms of sampling rate is possible for robot to operate properly.

Table 2: The length of robot's legs (in cm) of sampling rate test

Sampling Rate		5 ms	10 ms	20 ms	100 ms
1st leg	Upper Limb	40.000	40.000	40.000	40.000
	Lower Limb	20.009	19.999	20.017	20.021
2nd leg	Upper Limb	40.000	40.000	40.000	40.000
	Lower Limb	39.972	39.999	40.000	40.000
3rd leg	Upper Limb	40.000	40.000	40.000	40.000
	Lower Limb	39.982	39.879	40.000	39.999
4th leg	Upper Limb	40.000	40.000	40.000	40.000
	Lower Limb	39.980	39.999	40.000	40.000
RMS error		0.019	0.181	0.003	0.005

5. EXPERIMENTAL RESULTS

To be certain that the proposed method is feasible to employ with practical robots, two experiments are conducted in this study. The PLI algorithm is programmed and tested with 2-DOFs-quadruped robot with external Inertial Measurement Unit (IMU) – which is used to measure the orientation of robot's body. Two cases of broken legs, one leg ripped out and three limbs ripped out, are conducted in this test.

5.1 Experimental Setup

In the experiment, a robot is made up with four legs consisting of two degrees of freedom per each leg. Eight Dynamixel servo motors are employed as actuators for each joint, illustrated in Fig.7. The size of robot's body is set as a square with 16 cm width (l_b). The lengths of lower limb (l_l) and upper limb (l_u) are 6 cm and 8 cm, respectively. The quadruped robot is controlled through the computer with FDIII-HC board and USB cable. The orientation of robot is measured via x-IMU, high-performance and well-calibrated IMU, which is connected to computer wirelessly with Bluetooth technology.

For simulation side, the parameters, w, c_1 and c_2, are set as 0.99, 0.5 and 0.5, in order. The weight balance is set as 0.99 since it is expected to be minimized over time – additionally, the solutions of evolutionary algorithm (like PSO) will be getting optimized as time passes. The values of parameters, c_1 and c_2, are identical because of leading the particles to follow both global and local solutions. Even the proposed algorithm can operate successfully in the simulation, there is a minor contrast between simulation and real-world environment. To avoid such problem, the high-performance sensor is selected to obtain the orientation's values in this study; however, there is still a gap between simulation and a practical robot in the experiments as shown in Fig. 8. The values of orientation achieved by x-IMU are rather diverse from the values calculated in the simulation. Even if the values are not completely similar, the tendency goes somewhat in the same direction.

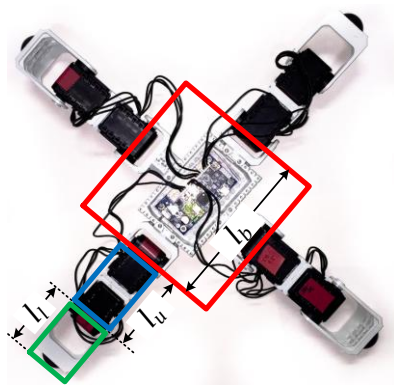


Fig. 7. The quadruped robot utilized in the experiments.

The proposed algorithm, PLI, is modified to overcome the problems of dissimilar orientation. Firstly, the velocity (v), used to update the particle, is changed from Eq. (4) as follows:

$$v_{id} = (1-cw)\cdot[w\cdot v_{id} + c_1\cdot rand()\cdot(p_{id} - x_{id}) + c_2\cdot rand()\cdot(p_{gd} - x_{id})] \tag{9}$$

Since there is a difference between simulation and actual robot – making this term $(1-cw)$ become 0 gradually, transferring Eq. (4) to Eq. (9) leads the velocity to become convergent rapidly. The position (x) is updated by using PSO original equation Eq. (5). Secondly, the probability of mutation (*prob*) is adjusted to 0.50 to avoid particles to allocate in local optimal. Thirdly, the selection of new updated model is done by the current and previous fitness values (f_c and f_p). The final solution will be achieved if f_c and f_p are greater than the desired value, set as 0.95 in this study. In additional, the sampling rate, to read the sensor values, is defined as 20 ms, and the candidate models in the search space are programmed to perform for 250 ms. The population of particles is set as 80 particles.

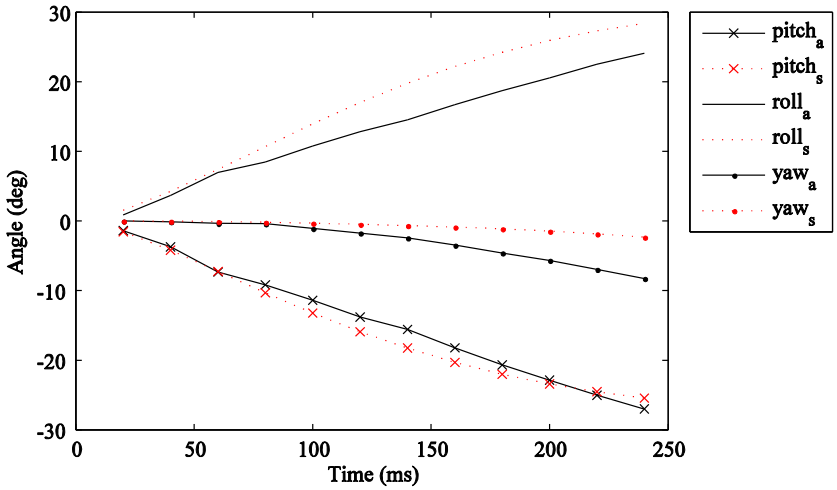
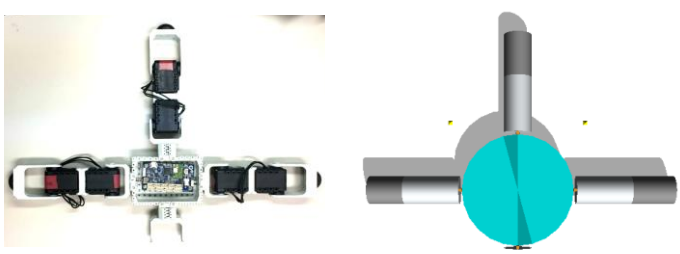


Fig. 8. The orientation of robot's body between actual robot (a) and simulation (s).

5.2 One leg ripped out (Case D)

According to the result written in Table 3, the PLI algorithm provide the acceptable result with 3.392 RMS error. The finding process is done with 6 iterations, and the maximum fitness value is 0.953. A main obstacle which causes an error can be noted as the difference between the orientations of real damaged robot and candidate models in the simulation. Therefore, it can be concluded that the proposed method, PLI, can assist the broken robot to discover a new updated model. Fig. 9 shows the actual robot and the new model found by PLI algorithm.

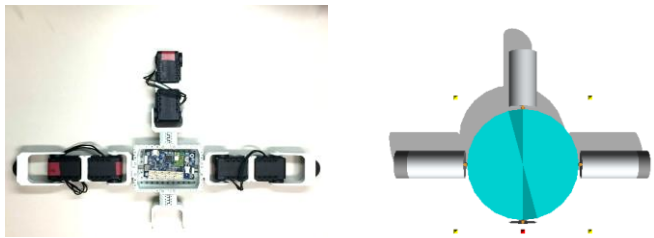


Actual Robot Simulation

Fig. 9. The actual damaged robot and the updated model provided by PLI of *Case D*.

5.3 One leg and one limb ripped out (Case E)

This test case is similar to *Case C* in previous section, and it is hard to comprehend the final solution. The PLI cannot discover the new model efficiently as shown in Table 3. Due to the same reason causing to Case C, the RMS error is quite high as 21.133. The main problem providing the error is not only the difficulty but also the gap between simulation and real world; nevertheless, the PLI algorithm can give a rough structure of new model in this experiment as illustrated in Fig. 10. It is worth noting that if the gap between simulation and actual robot can be minimized, the successful rate can be increased, following the simulation of *Case C*.



Actual Robot Simulation

Fig. 10. The actual damaged robot and the updated model provided by PLI of *Case E*.

Table 3: The length of robot's legs (in mm) from PLI in the experiments

Leg Number		Upper Limbs				Lower Limbs				RMS Error
		1st	2nd	3rd	4th	1st	2nd	3rd	4th	
Case D	*Actual*	0.000	80.000	80.000	80.000	0.000	60.000	60.000	60.000	-
	PLI	0.000	80.000	80.000	80.000	0.000	61.132	61.609	50.610	3.392
Case E	*Actual*	0.000	80.000	80.000	80.000	0.000	60.000	0.000	60.000	-
	PLI	0.000	80.000	80.000	80.000	0.000	16.600	0.000	18.900	21.133

6. CONCLUSIONS

This paper has proposed to develop the leg-loss identification algorithm based on particle swarm optimization method and normalized cross-correlation algorithm, called as PSO-based Leg-loss Identification (PLI). The procedure of detecting the broken legs is done by comparing the actual damaged robot and a number of candidate robots in the search space. The responses of actual damaged robot and candidate model are compared by the cross-correlation method, which is also utilized as the fitness function in PSO. The PSO algorithm is employed to evolve the candidate models to provide the same behavior as the damaged robot. In the experiment, the results of PLI are compared with GA-based method, and they show that PLI is better in term of efficiency. In addition, the number of particles is varied to evaluate the performance of the proposed algorithm, and it can be summarized that, in complex situation, a large number of particle can assist the robot to detect a failure part faster, but in case of simple problem, a limited number of particle can provide more robustness. However, in the future, it is planned to test the performance of the algorithm in various problems. To be certain that the PLI can handle the noisy system (represented the practical situation), the noise analysis is also conducted, and the result shows that even if noisy signals provide more error than non-noisy signals, the error become small at the end of process. The changing of sampling rate is experimented as well. The error between best candidate model and actual damaged robot is very small even the sampling rate increased ten times. Moreover, the experiments also show that the PLI can discover the new updated model with actual damaged robot. However, the difference between simulation and practical situation makes an error to the final model but it is acceptable as the structure of the model is similar to the damaged robot. It can be concluded that the proposed method (PLI) is feasible to detect the broken leg by discovering the new model, which will be beneficial for legged robot to create the alternative walking behavior after damaged. Additionally, the performance of the PLI can be improved by adding the transferring module between simulation and real world, which decreases the gap and errors.

REFERENCES

[1] Ren, G., Chen, W., Dasgupta, S., Kolodziejski, C., Worgotter, F. and Manoonpong, P. Multiple chaotic central pattern generators with learning for legged locomotion and malfunction compensation, Information Sciences, Vol. 294, 2015, pp. 666-682.

[2] Koos, S., Cully, A. and Mouret, J.B. Fast damage recovery in robotics with the T-resilience algorithm, International Journal of Robotics Research, Vol. 32(14), 2013, pp. 1700-1723.

[3] Seljanko, F. Fault detection algorithm for legged walking robot, IEEE International Conference on Mechatronics and Automation, 2013, Takamatsu, Japan.
[4] Roennau, A., Heppner, G., Kerscher, T. and Dillmann, R. Fault diagnosis and system status monitoring for a six-legged walking robot, IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2011, Budapest, Hungary.
[5] Hashlamon, I. and Erbatur, K. Joint sensor fault detection and recovery based on virtual sensor for walking legged robots, IEEE 23rd International Symposium on Industrial Electronics (ISIE), 2014, Istanbul, Turkey.
[6] Johnson, A.M., Haynes, G.C. and Koditschek, D.E. Disturbance detection, identification, and recovery by gait transition in legged robots, IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010, Taipei, Taiwan.
[7] Bongrad, J., Zykov, V. and Lipson, H. Resilient machines through continuous self-modeling, SICENCE, Vol. 314, 2006, pp. 1118-1121.
[8] Liang, J. and Xue, C. Self identification and control of four-leg robot based on biological evolutionary mechanisms, IEEE Conference on Industrial Electronics and Applications, 2010, Taichung, Taiwan.
[9] Gao, H., Wang, T., Liang, J. and Zhou, Y. Model adaptive gait scheme based on evolutionary algorithm, IEEE Conference on Industrial Electronics and Applications, 2013, Melbourne, Australian.
[10] Stein J.Y. Digital Signal Processing: A Computer Science Perspective, ISBN 0-471-29546-9, 2000, John Wiley & Sons, New York.
[11] Shi, Y. and Eberhart, R. A modified particle swarm optimizer, IEEE World Congress on Computational Intelligence, 1998, Anchorage, Alaska, USA.