Intelligent Trajectory Planning and Control of a Humanoid Robot using a new Eltisl-Based Selfish Gene Alorithm

Lyes Tighzert, Thafsouth Aguercif, Boubekeur Mendil, Cyril Fonlupt

Intelligent Trajectory Planning and Control of a Humanoid Robot Using a New Elitism-Based Selfish Gene Algorithm

Lyes Tighzert, Thafsouth Aguercif, Cyril Fonlupt, and B. Mendil

***Abstract*—The contemporary trend in science and society consists of searching for solutions to enhance people life, safety, economy, and health while protecting environment. In recent years, we have witnessed the arrival of complex machines with structures similar to humans known as humanoids robots. The combination of these technologies and optimization technics may result in robust, safe, reliable, and flexible machines that can substitute humans in multiple difficult tasks. In order to contribute to this topic, we propose tow new evolutionary algorithms based on the selfish gene theory and elitism strategies. Therefore, permanent elitism-based selfish gene algorithm (peSGA) and nonpermanent elitism based selfish gene algorithm (neSGA) are proposed. In order to validate and to evaluate the performance peSGA and neSGA, a numerical experiment is performed using IEEE CEC 2014 functions. the The obtained results show that the proposed algorithms are very competitive. Furthermore, evomutionary optimization of a walking robot is formulated. The proposed algorithm are applied to the generation and control of optimal motion of a humanoid robot.**

I. Introduction

Humanoid robots are articulated machines with structures similar to humans. These robots, with high degrees of freedom, are one of the most complex system designed by humans. Their control is one the most difficult engineering problems. Despite this, scientists and engineers have proposed methods, theories, and sophisticated algorithms that deal with these problems. Stable walking [1], hopping [2], running [3] and gymnastic movements [4, 5] are studied and successfully realized. The combination of humanoid robots and artificial intelligence technics, like artificial neural network, fuzzy reasoning and developmental have allowed them to learn [6], to communicate [7] and to mimic several human tasks [8].

Evolutionary computing (EC), e.g. genetic algorithm (GA), evolutionary algorithm (EA) etc., is one of the most investigated approaches for optimization [9]. Engineers and scientists have proposed several algorithms that are inspired from evolution theory. In 1999, Corno et al have proposed a new algorithm called selfish gene algorithm (SGA) [10]. This later is inspired from the modern theory of evolution, called the selfish gene theory (SGT), proposed by the biologist Richard Dawkins [11]. In SGT, the evolution is based on genes rather than individuals. Thus, the natural selection acts on genes rather than individuals. In the point of view of SGT, the breeds' bodies is made by genes only to assure their own replication through the reproduction process. Hence, Species are just vehicles that reproduce genes [11]. Unlike other evolutionary algorithms that are based on a populations of individuals, the SGA proposed by Corno et al. uses the same paradigms of Dawkins's theory of evolution. Therefore, the SGA is based on genes and the population is replaced by the concept of virtual population (VP) that consists of a pool of genes. At each generation, two individuals, denoted G1 and G2 respectively, are sampled from VP. The SGA associates for each gene a selection probability that measure the chance that it have to be selected. The selection probability is tuned by a tournament selection, i.e. a completion between G1 and G2. The winner's gene are rewarded by increasing their selection probabilities by 1/N and the loser genes are punished by decreasing their selection probability by 1/N. Note that N is the number of gene competing for each locus. One can see that in SGA, the genes are competing for loci. The pseudocode of SGA is shown in Fig.1.

Algorithm1: Selfish Gene Algorithm [10]

```
1.  VP = uniform initialization of the virtual     population
2.  P = initialize all probabilities p_ij to l/n_i
3.  B = select (VP)  / * assume the best */
4.  Repeat
5.  / * tournament  */
6.      G1 = select-individual ( VP,P) ;
7.      G2 = select-individual ( VP,P) ;
8.      if (fitness(G1) > fitness(G2)) then
9.          reward-alleles (G1) ;
10.         penalize-alleles (G2) ;
11.         / * update best */
12.         if (fitness(G1) > fitness(B))  then
a.              B = G1 ;
13.         end if
14.     else
15.         reward-alleles (G2) ;
16.         penalize-alleles (G1) ;
17.         / * update best */
18.         if (fitness(G2 > fitness(B)) then
19.             B = G2 ;
20.         end if
21.     end if
22. end while
23. return B
```

Fig. 1. The pseudo code of SGA.

Lyes TIGHZERT is with the Laboratoire de Technologie Industrielle et de l'Information (LTII), Faculté de Technologie, Université de Bejaia, 06000 Bejaia, Algérie (e-mail: ltighzert@gmail.com).

Thafsouth AGUERCIF is with the Laboratoire de Technologie Industrielle et de l'Information (LTII), Faculté de Technologie, Université de Bejaia, 06000 Bejaia, Algérie (e-mail: aguercifthafsouth@gmail.com).

Cyril Fonlupt is with Laboratoire d'Informatique Signal et Image de la Côte d'Opale (LISIC) Université du Littoral - Côte d'Opale BP 719 62228 CALAIS Cedex – France (cyril.fonlupt@univ-littoral.fr)

Boubekeur MENDIL is with the Laboratoire de Technologie Industrielle et de l'Information (LTII), Faculté de Technologie, Université de Bejaia, 06000 Bejaia, Algérie (e-mail: bmendil@yahoo.fr).

L. Tighzert and B. Mendil have proposed recently a new variants called the replacing and never penalizing selfish gene algorithm (RNPSGA) that replace the loser's genes by those of the winner rather than penalizing them [4]. The authors have demonstrated its performance under several benchmark functions and have applied it to the realization of gymnastic movements of a humanoid robot. In this paper, we propose the reconciliation of SGA with elitism strategies. Permanent elitism-based selfish gene algorithm (peSGA) and nonpermanent elitism-based selfish gene algorithm (neSGA) are proposed and analyzed. The proposed elitisms strategies make selection pressure that is good enough to allow global convergence. Thus, the SGA performance are improved. The rest of this paper is organized as follow.

In next section, we present the proposed algorithms, i.e. peSGA and neSGA. Third section presents the validation of the proposed algorithms on the IEEE CEC2014 benchmark functions. In section four, we give an application of the proposed algorithms to optimization of bipedal waking of seven links humanoid robot. Section 5 concludes this paper.

II. ELITISM-BASED SELFISH GENE ALGORITHM

In order to enhance the performance of SGA, we propose two elitism strategies.

A. Permanent elitism-based SGA

Algorithm2: permanent elitism-based SGA (peSGA)

```
1. M= optimization problem dimension, K=0;
2. VP = uniform initialization of the virtual  population
3. P = initialize all probabilities p_ij to l/N
4. N= initialize the competing gene number for each locus
5. P_OV= initialize the overwriting  probability
6. P_a = initialize adjustment rate
7. Best = select (VP, P)  / * assume the best */
8. BestFit=evaluate(Best) , K=K+1;
9. while ( termination condition )
10.     G = select-individual ( VP, P)
11.     GFit=evaluate(G) ,   K=K+1
12.     Gm=adjust G's genes with probability P_a using (1)
13.     Reduce rayon adjustment according to (2)
14.     GmFit=evaluate(Gm) ,  K=K+1
15.     if GmFit > GFit        /* ">" means better */
16.       G=Gm,     GFit=GmFit
17.     End if
18.     if (fitness(G) > fitness(Best))  /* ">" means better */
19.         reward-alleles (G) ;
20.         penalize-alleles (Best) ;
21.         Best=G, BestFit=GFit;
22.         Overwrite Best's genes by G's Genes with probability P_OV
23.     else
24.         reward-alleles (Best) ;
25.         penalize-alleles (G) ;
26.     end if
27.  End while
28.  return Best
```

Fig. 2. The pseudo code of peSGA.

In this subsection, we propose permanent elitism selfish gene algorithm (peSGA). This strategy is inspired from the works of Ahn Chang Wook and Rudrapatna S. Ramakrishna [12]. The idea is to apply a selection pressure to accelerate the convergence while avoiding local convergence. In peSGA, at each generation only one individual, denoted G, is generated from VP. Then, a tournament is done between the elite (best) individual and G. The winner's gene are reworded and the loser's gene are penalized. Furthermore, the sampling mechanism is done differently than in the classical SGA. An adjustment procedure is added. During the sampling proves, each generated gene can be adjusted with a probability P_a. The adjustment formula is given in (1).

$$G_i^{K+1} = G_i^K + AR^{K+1} \times randn(-1,1) \quad (1)$$

Where G_i^{k+1} is the value of gene of locus I at the iteration K+1, G_i^k is the value of gene of locus I at the iteration K, AR^K is the adjustment rayon at iteration K.

The Adjustment rayon is decreased using formula (2).

$$AR^K = AR_{max} + (AR_{max} - AR_{min}) * K/Maxiter. \quad (2)$$

Where AR_{max} is the maximum value AR, AR_{min} is the minimum value of AR, K is the currant iteration index and Maxiter in the maximum number of iteration. The pseudocode of the proposed peSGA is given in Fig.2.

B. Nonpermanent elitism-based SGA

Algorithm 3: nonpermanent elitism-based SGA (neSGA)

```
1.  M= optimization problem dimension, K=0; τ=0;
2.  VP = uniform initialization of the virtual  population
3.  P = initialize all probabilities p_ij to l/N
4.  N= initialize the competing gene number for each locus
5.  P_OV= initialize the overwriting  probability
6.  P_a = initialize adjustment rate
7.  η= N/2,  / * the predefined inherence length */
8.  Best = select (VP)  / * assume the best */
9.  BestFit=evaluate(Best) , K=K+1;
10. Elite = select (VP, P)  / * assume an elite  */
11. ElitetFit=evaluate(Elite) , K=K+1;
12. while ( termination condition )
13.   if  τ< η
14.      G = select-individual ( VP, P)
15.      GFit=evaluate(G) ,   K=K+1
16.      Gm=adjust G's genes with probability P_a using (1)
17.      Reduce rayon adjustment according to (2)
18.      GmFit=evaluate(Gm) ,  K=K+1
19.      if GmFit > GFit        /* ">" means better */
20.         G=Gm,     GFit=GmFit
21.      end if
22.      if (fitness(G) > fitness(elite))  /* ">" means better */
23.         reward-alleles (G) ;
24.         penalize-alleles (elite) ;
25.         Elite=G, EliteFit=GFit;
26.         Overwrite elite's genes by Elite's Genes with probability P_OV
29.         τ= 0
30.         if  (fitness(G) > fitness(Best))  /* ">" means better */
31.            Best=G, BestFit=GFit;
27.         end
28.      else
29.         reward-alleles (Elite) ; penalize-alleles (G) ;
30.         τ= τ+1
31.      end if
32.   else
33.      Elite = select (VP, P)  / * assume an new elite  */
34.      ElitetFit=evaluate(Elite) , K=K+1;
35.      τ= 0
36.   End if
37. end while
38. return Best
```

Fig. 3. The pseudo code of neSGA.

One of the drawbacks of peSGA may that the applied selection pressure made by keeping the elite as long as possible. This high selection pressure may result in premature convergence. In nonpermanent elitism based selfish gene algorithm (neSGA), the elite is guarded for a given number

of cycles in which is not surpassed. This means that the elite is updated after a predefined inherence length denoted η. For the rest, peSGA is similar to neSGA. The pseudocode of neSGA is given in Fig.3.

TABLE I. EXPERIMENTAL RESULTS OF SGA, RNPSGA, PESGA, AND NESGA OVER 25 INDEPENDENT RUNS ON 30 TEST FUNCTIONS OF 30 VARIABLES WITH 300,000 FES. "MEAN " AND "STD"INDICATE THE AVERAGE AND STANDARD DEVIATION OF THE FUNCTION ERROR VALUES OBTAINED IN 25 RUNS, RESPECTIVELY.

Function		SGA	RNPSGA	peSGA	neSGA
1	Mean	8.8391e+006	1.0998e+005	1.6953e+006	2.0293e+006
	STD	5.9489e+006	7.2818e+004	1.4220e+006	1.4482e+006
2	Mean	3.0021e+007	1.8399e+003	2.6658e+003	4.1337e+003
	STD	3.5475e+007	2.1664e+003	2.8912e+003	3.4341e+003
3	Mean	6.1453e+003	5.3290e+003	5.6906e+003	7.4229e+003
	STD	5.5397e+003	5.4175e+003	4.6509e+003	4.8457e+003
4	Mean	5.1538e+001	3.5803e+000	1.5174e+001	1.8634e+001
	STD	1.5675e+001	2.1576e+000	1.8683e+001	2.1433e+001
5	Mean	2.0093e+001	1.9201e+001	1.9999e+001	2.0000e+001
	STD	5.3231e-002	3.9164e+000	1.6783e-003	6.8063e-004
6	Mean	4.7183e+000	3.8869e+000	4.5043e+000	5.1598e+000
	STD	1.3987e+000	1.5154e+000	1.4108e+000	1.5977e+000
7	Mean	1.2749e+000	1.6336e-001	1.6520e-001	1.3037e-001
	STD	3.3043e-001	8.0134e-002	1.0558e-001	6.8026e-002
8	Mean	5.4338e+000	1.0825e+001	1.0546e+001	1.2178e+001
	STD	2.1343e+000	4.7535e+000	4.3505e+000	4.7615e+000
9	Mean	1.6187e+001	2.0098e+001	1.9248e+001	2.3839e+001
	STD	4.7513e+000	9.1924e+000	9.1792e+000	8.8700e+000
10	Mean	3.4264e+001	3.1268e+002	2.8065e+002	3.7919e+002
	STD	4.0662e+001	1.1674e+002	1.0261e+002	1.4329e+002
11	Mean	6.6327e+002	6.8359e+002	6.6865e+002	8.5289e+002
	STD	2.5548e+002	2.2289e+002	2.6422e+002	2.7343e+002
12	Mean	5.6671e-001	1.0878e-001	1.1815e-001	1.5126e-001
	STD	1.8193e-001	6.0947e-002	8.6073e-002	1.1456e-001
13	Mean	2.2917e-001	5.6372e-001	4.5187e-001	5.6223e-001
	STD	6.8267e-002	1.7779e-001	1.6304e-001	1.9712e-001
14	Mean	3.3928e-001	4.4715e-001	5.6759e-001	4.5673e-001
	STD	6.3867e-002	2.1789e-001	2.8138e-001	2.6867e-001
15	Mean	4.2601e+000	1.5945e+000	3.1908e+000	2.7588e+000
	STD	2.3018e+000	1.0458e+000	1.9813e+000	1.1566e+000
16	Mean	3.1788e+00	2.9497e+000	2.8185e+000	2.8998e+000
	STD	3.0139e-001	5.1414e-001	4.6450e-001	4.3076e-001
17	Mean	1.0812e+006	3.9386e+004	6.1347e+005	1.0057e+006
	STD	1.1697e+006	4.3241e+004	5.1031e+005	6.0329e+005
18	Mean	1.6340e+005	6.2094e+003	6.2715e+003	7.3055e+003
	STD	6.1673e+005	6.2300e+003	7.0412e+003	8.1281e+003
19	Mean	2.9898e+000	1.6216e+000	1.6717e+000	1.9974e+000
	STD	1.2102e+000	9.4023e-001	1.0542e+000	8.1092e-001
20	Mean	5.6601e+003	3.7832e+003	3.1857e+003	2.5409e+003
	STD	5.9977e+003	4.6321e+003	5.8837e+003	2.8518e+003
21	Mean	2.7184e+005	1.7304e+004	5.6421e+005	6.0261e+005
	STD	3.7984e+005	1.0660e+004	7.3073e+005	7.6800e+005
22	Mean	8.4679e+001	1.7066e+002	1.8277e+002	1.9536e+002
	STD	6.4120e+001	5.0939e+001	1.0888e+002	1.0623e+002
23	Mean	3.3494e+002	3.2945e+002	3.2945e+002	3.2945e+002
	STD	6.8668e+000	7.6936e-005	6.4674e-005	2.7065e-005
24	Mean	1.5606e+002	1.4621e+002	1.4306e+002	1.3960e+002
	STD	2.8575e+001	2.4652e+001	2.7629e+001	1.4609e+001
25	Mean	1.8789e+002	2.02426e+02	2.0266e+002	2.0305e+002
	STD	2.1699e+001	4.06186e+00	1.2439e+000	1.1586e+000
26	Mean	1.0140e+002	1.1627e+002	1.0833e+002	1.2428e+002
	STD	2.9798e+000	3.6552e+001	2.7038e+001	4.2572e+001
27	Mean	4.0278e+002	4.0532e+002	4.0654e+002	4.1369e+002
	STD	3.7082e+001	9.2968e+001	3.9069e+001	4.6694e+001
28	Mean	4.9379e+002	6.8643e+002	6.7349e+002	6.7465e+002
	STD	6.6058e+001	1.4203e+002	1.2378e+002	2.0022e+002
29	Mean	4.7815e+004	2.1025e+002	7.8504e+005	4.2333e+005
	STD	1.2608e+005	9.4819e+000	1.0499e+006	8.4798e+005
30	Mean	2.3169e+003	1.1709e+003	1.3910e+003	1.6133e+003
	STD	1.1451e+003	2.7562e+002	3.7766e+002	2.9799e+002

We note that the VP and the probability selection in the proposed peSGA and neSGA are matrix of size (N×M), where N in the number of genes competing for each locus and M is the dimension of the optimization problem. Hence, each gene contains one allele and represents an optimization parameter. The main difference between peSGA and neSGA consists of the fact that peSGA finds a near optimal solution (i.e., a winner) that is maintained as long as other solutions generated from the virtual population (the pool of genes) are not better. In contrast, neSGA further improves the performance of the neSGA by avoiding strong elitism that may lead to premature convergence. The nonpermanent SGA comes with all the benefits of the peSGA. In addition, it maintains genetic diversity as a bonus. In the next section we evaluate the performance of peSGA and neSGA under IEEE CEC2014 functions.

III. VALIDATION

In order to evaluate the performance of the proposed algorithms, we propose a benchmarking under the IEEE CEC2014 benchmark functions [13]. This black box contains 30 functions including multimodal, unimodal, separable and non-separable functions. The dimension is fixed to D=30, the maximum number of fitness evaluation is set to 300000 as recommended in [13]. Each test is repeated 25 times. The averaged errors and the standard deviations are reported in Table I.

The proposed peSGA and neSGA are compared to the original SGA [14, 10] and to the most recent SGA variant known as the replace and never penalize selfish gene algorithm (RNPSGA) [4]. The parameters setting of the compared algorithms are as follow.

$AR_{max}=5$; $AR_{min}=10^{-10}$; $P_{ov}=0.15$; $P_a=0.01$; N=50; η=N/2.

The mean and standard deviation of the function error value ($f(x) - f(x^*)$) were calculated over 25 independent runs for each test function, where x is the best solution returned by the algorithm after 300000 fitness evaluations and x^* is the global optimal solution. As the compared algorithms are stochastic and their results differ freom one run to another, a nonparametric test should be used to compare the results. Hence, Wilcoxon's rank sum test, recommended by [15], at a 0.05 significance level was performed to test the statistical significance of the experimental results between two algorithms. The obtained results are reported in Table II. For each test function the proposed peSGA and neSGA are compared both to peSGA and PSGA. "–", "+", and "≈" denote that the performance of the corresponding proposed algorithm, e.g. peSGA or neSGA, is worse than, better than, and similar to that of the corresponding algorithm, respectively.

In order to rank the compared algorithm, the statistical Freedman test was performed. This test gives a global ranking of the compared algorithms. The results are presented in Table III. From this experimental study, we can conclude that the proposed algorithms are very competitive. They present performance enhancement for both unimodal and multimodal problems. According to Freedman test presented in Table 3, the proposed neSGA is the best algorithm for this experimentation. Thus, the proposed peSGA and neSGA can be used for real word applications.

TABLE II. WILCOXON'S RANK SUM TEST RESULTS AT A 0.05 SIGNIFICANCE LEVEL BETWEEN BOTH PESGA AND NESGA AND EACH OF SGA AND RNPSGA.

Algorithm	peSGA		neSGA	
Compared to	SGA	RNPSGA	SGA	RNPSGA
Function	Comparison Results			
1	+	-	+	-
2	+	=	+	=
3	=	=	=	=
4	+	=	+	=
5	+	+	+	-
6	=	=	=	-
7	+	=	+	=
8	-	=	-	=
9	=	=	-	=
10	-	=	-	=
11	=	=	-	-
12	+	=	+	=
13	-	=	-	=
14	-	-	=	=
15	+	-	+	-
16	+	=	+	=
17	=	-	=	-
18	+	=	=	=
19	+	=	+	=
20	=	=	=	=
21	=	-	=	-
22	-	=	-	=
23	+	-	+	-
24	=	+	=	+
25	=	=	-	=
26	-	=	-	=
27	=	=	=	=
28	-	=	-	=
29	=	-	-	-
30	+	=	+	-
+	12	2	11	1
-	7	7	10	10
=	11	21	9	19

"–", "+", and "=" denote that the performance of the corresponding proposed algorithm is worse than, better than, and similar to that of the corresponding RNPSGA and SGA, respectively.

IV. APPLICATION TO HUMANOID ROBOT

A. Robot Model

The humanoid robot considered in this study is the *HYDROID* robot [16]. The dynamic parameters of the robot are given in Table IV. The length of each robot link except the feet is denoted l_i; the distance between the each link mass center is denoted lc_i; each link mass is denoted m_i, the inertia moment of each link is denoted I_i. The robot foot structure is represented in Fig. 5. The vector F_1 represent the ground reaction, x_{ZMP} represent the position of the zero moment point (ZMP) [17]. The hydroid robot is represented in Fig.5. We consider here the model in the sagittal plane. This seven links robot has 9 degrees of freedom (9 DOF) which are left foot angle (α1), left knee angle (α2), left leg angle (α3), body angle (α4), right foot angle (α5), right knee angle (α6), right leg angle (α7), x body position and the z body position. The geometric, cinematic and the dynamic model of this robot are not reported here in details because of the limited paper pages. *HYDROID* is described in details in the literature (the lectors can see also [18]).For simulation, we have created a virtual model of HYDROID robot in SimMechanics on Simscape/Matlab2014. Fig.5 gives the obtained 3D model. It shows the robot links, their center of mass and their associated frames. The ground contact is also modeled using an approximate model known as spring-mass-dumper model. Other methods can be found in the literature [19]. The relationship between the foot penetration, foot velocity and the ground reaction force (F_1) is given in (3)

$$F = K\Delta z + C\Delta \dot{z} \quad (3)$$

Where F represent the reaction force, K and C are constants describing the nature of impact. is the hypothetic penetration of the foot in the ground.

TABLE III. ALGORITHMS' RANKING ACCORDINF TO FREEDMAN TEST. THE BEST RANKS ARE GIVEN IN BOLD STYLE

Function	SGA	RNPSGA	peSGA	neSGA
1	2.0400	3.0800	3.2400	**1.6400**
2	2.6800	3.1600	3.1600	**1.0000**
3	**2.2000**	2.4400	2.7200	2.6400
4	2.9600	2.8800	2.5600	**1.6000**
5	2.5200	2.8400	3.5600	**1.0800**
6	**2.2800**	2.4800	2.3200	2.9200
7	3.2000	2.9600	2.8400	**1.0000**
8	**1.5600**	2.5200	2.8400	3.0800
9	**2.2000**	**2.2000**	2.6400	2.9600
10	**1.0400**	2.8400	2.9200	3.2000
11	2.3200	2.4000	**2.1200**	3.1600
12	2.8400	3.0400	3.1200	**1.0000**
13	**1.1600**	2.8000	3.0000	3.0400
14	2.4000	**2.1600**	2.6400	2.8000
15	2.5600	2.8800	2.4000	**2.1600**
16	2.8800	2.4400	2.7200	**1.9600**
17	2.4800	**2.0400**	3.0800	2.4000
18	2.7200	2.7600	2.5600	**1.9600**
19	2.7200	2.8400	2.4800	**1.9600**
20	2.4000	3.1600	2.2800	**2.1600**
21	2.3200	**2.1200**	2.7600	2.8000
22	**1.9200**	2.6000	2.4800	3.0000
23	2.0000	3.6400	3.3600	**1.0000**
24	2.6000	3.0800	2.3200	**2.0000**
25	**1.7600**	2.9600	2.8000	2.4800
26	**2.0800**	2.3200	2.8000	2.8000
27	**2.2800**	2.6400	2.5200	2.5600
28	**1.3600**	2.5600	2.8800	3.2000
29	**2.1600**	3.0800	2.2800	2.4800
30	2.3600	2.7200	3.2800	**1.6400**
Rank	2.2667	2.7213	2.7560	**2.2560**

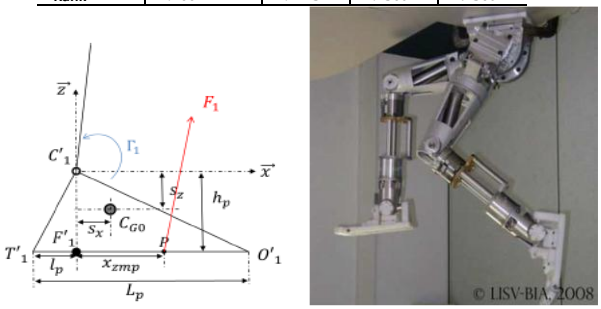

Fig. 4. Robot foot structure and the hydroid robot [16]

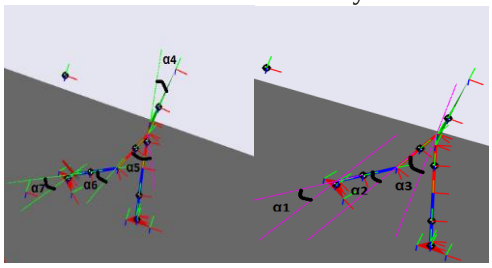

Fig.5. HYDROID robot model in Simscape/Matlab

TABLE IV. HYDROID ROBOT PARAMETERS

Links	Length [m]	Mass [Kg]	Inertia moment [$kg.m^2$]	center of gravity [m]
Foot	L_p=0.207 l_p=0.072 h_p=0.064	0.678	0.001	s_x=0.135 s_y=0.321
Tibia	l_1=0.392	2.188	0.028	0.1685
Shin	l_2=0.392	5.025	0.068	0.1685
Trunk	l_3=0.543	29.27	0.815	0.01921

B. Dynamic Robot Model

The dynamic model can be obtained by Newton-Euler formalism or by Euler-Lagrange equation [20]. In this paper, the Euler-Lagrange equation is used (4).

$$\Xi_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \left(\frac{\partial L}{\partial q_i}\right) \quad (4)$$

Where Ξ_i Corresponds to the force or torque resulting from the dynamics of the i[th] link. L in the langrangian of the i[th] link. The obtained dynamic model if given in (5).

$$D(X)\ddot{X} + H(X,\dot{X})\dot{X} + Q(X) = B\cdot\Gamma + A_1F_1 + A_2F_2 \quad (5)$$

Where X is the state variable vector composed from the articulations angles and the x and z body position $X = [\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, x, y]^T$; D(X) is the (9×9) inertia matrix; H(X, $\dot{X}$) is the centrifuge and Coriolis matrix of dimension (9×1); Q(X) is the gravitational matrix of dimension (9×1); B is a ponderation matrix; Γ is the joint actuators torques at the articulations; F_1 is the ground reaction on the left foot; F_2 is the ground reaction on the right foot; A1 is the Jacobian matrix of the left foot contact with the ground; and A_2 is the Jacobian matrix of the right foot contact with the ground.

The dynamic walking can be considered as a hybrid process constituted of 3 phases. The double support phase (two feet on the ground), the simple support phase (one foot on the ground) and the impact phase (when the flying foot bumps the ground surface). The model given in (5) describe the double support phase. In the case of simple support phase, F_1 or F_2 is equal to zero.

C. The Impact Model

We have chosen to consider impulse impacts. This choice simplify the resolve of contact between the ground and the foot. The detailed impact model is given in (6).

$$D(X)(\dot{X}^+ - \dot{X}^-) = A_1F_1 + A_2F_2 \quad (6)$$

Where $\dot{X}^+$ and $\dot{X}^-$ are the velocities after and before the impact, respectively. The constraints associated to impact are given in (7) and (8). These latter gives the constraints of non-slip and the non-detachment of the feet.

$$A_1(X)\dot{X}^+ = 0 \quad (7)$$

$$A_2(X)\dot{X}^+ = 0 \quad (8)$$

D. Stability and ZMP

The stability of humanoid robot is one of the most topics studied. The stability of these articulated machines is assured if and only if the zero moment point (ZMP) is in the sustention polygon. This principle was proposed by Miomir Vukobratović [17]. In our case, the ZMP position is given in (9). When this position is in the sustention polygon the robot is stable.

$$x_{ZMP} = \frac{1}{F_{1z}}(\Gamma_1 + s_x m_0 g - h_p F_{1x}) \quad (9)$$

E. Trajectory generation

The trajectories of each articulation angle is assumed to be *spline cubic function*. This type of function has been used in several works. The walking period is denoted T. We decompose each robot step into two phases. The mathematical definition of each spline cubic function during each robot forward step is given in (10) and (11).

$$\alpha_i(t) = \sum_{k=1}^{3} a_k^i t^k \ , \qquad 0 < t < T/2 \quad (10)$$

$$\alpha_i(t) = \sum_{k=1}^{3} b_k^i (1-t)^k \ , \quad T/2 < t < T \quad (11)$$

Additional constraints can be imposed to assure on the continuity of the movement, i.e. the velocity and the acceleration at t=T/2. The parameters of the spline functions are calculated depending on the robot kinematic, dynamic and technologic constraints of the movement and an energetic criterion.

F. Kinematic and dynamic constraints

Robot walking consists on constrained movements. During footsteps, the stability must be assured. This leads to two constraints given in (12) and (13). During the movements, the position in the z-axis of toe and heel of the mobile foot must be positive which leads to (14) and (15). Anthropomorphic constraints on the knees' angles are given in (16) and (17). The bottom foot, i.e. toe and heel, z-axis velocity must be positive to assure that the fixed foot take off the ground, i.e. obtain a transition to the next footstep. This leads to constraints (18) and (19). The ground reactions are also positive which leads to (20) and (21).

$$C_1 = x_{ZMP} + l_p > 0 \quad (12)$$

$$C_2 = -x_{ZMP} + l_p + L_P > 0 \quad (13)$$

$$C_3 = z_t > 0 \quad (14)$$

$$C_4 = z_h > 0 \quad (15)$$

$$C_5 = -\alpha_2 > 0 \quad (16)$$

$$C_6 = -\alpha_6 > 0 \quad (17)$$

$$C_7 = z_t > 0 \quad (18)$$

$$C_8 = z_h > 0 \quad (19)$$

$$C_9 = F_{1z} > 0 \quad (20)$$

$$C_{10} = F_{2z} > 0 \quad (21)$$

The optimization criterion is given in (22)

$$C = \int_0^T \Gamma^T P\ \Gamma dt \quad (22)$$

Where P is an identity matrix. T is the footstep period.

G. Evolutionary optimization

In order to realize optimal stable walking that minimize (22) and respect the constraints presented above, evolutionary optimization of the walking can be formulated. According to the results exposed in previous section, neSGA is the most competitive algorithm compared to RNPSA, SGA and peSGA. The evolutionary optimization algorithm is summarized in Fig. 6.

Algorithm 4: neSGA applied to optimal walking problem

1. Initialize VP, P, N, P_{OV} , P_a, η, K.
2. Best = select (VP), Elite = select (VP, P) / * assume the best and elite */
3. BestFit=evaluate(Best) , ElitetFit=evaluate(Elite) K=K+2;
4. **while (** termination condition **)**
5. **if** $\tau < \eta$
6. G = select-individual from VP that verify the movement constraints
7. GFit=evaluate(G) , K=K+1 Gm=adjust G's
8. Reduce rayon adjustment according to (2)
9. GmFit=evaluate(Gm) , K=K+1
10. **if** GmFit < GFit G=Gm, GFit=GmFit **end if**
11. **if** (fitness(G) < fitness(elite)) /* "<" means better */
12. reward-alleles (G) ; penalize-alleles (elite) ; Elite=G, EliteFit=GFit; $\tau=0$
13. **if** (fitness(G) < fitness(Best)) **then** Best=G, BestFit=GFit; **end if**
14. **else** reward-alleles (Elite) ; penalize-alleles (G) ; $\tau = \tau+1$ **end if**
15. **else** Elite = select (VP, P), ElitetFit=evaluate(Elite) , K=K+1; $\tau = 0$ **end if**
16. **end while**
17. **return** Best

Fig. 6. Application of neSGA to Humanoid robot.

H. Results and discussions

We present in Fig.7 the obtained optimal trajectory using neSGA. A proportional derivative control law is used to control articulations' trajectories. Despite the high complexity of this task, the humanoid robots are hybrid highly nonlinear under actuated machine with nine DOF, a satisfying results are obtained. neSGA assures stability and minimal energy consumption. Hence, the proposed algorithms including neSGA, peSGA can be used in real-world applications. In the future works we can focus on the parameters adaptation of neSGA and peSGA.

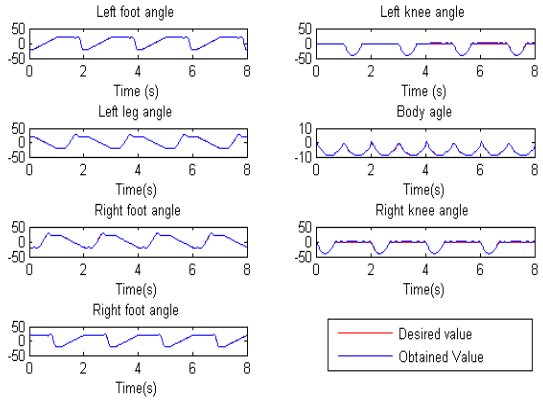

Fig.7. Optimal Trajectory of the articulations during four footsteps

V. CONCLUSION

In this paper two new evolutionary algorithms, neSGA and peSGA, are developed and presented. peSGA and neSGA are based on the selfish gene theory an use elitism strategies to assure selection pressure. The proposed algorithms are benchmarked on IEEE CEC2014 functions. Their performance are compared to the SGA stat-of-art variants. The obtained results show that the proposed approaches leads to performance enhancement. The proposed neSGA is than applied to a constrained multimodal nonlinear real word problem. The objective is to realize optimal walking of HYDROID robot. The obtained results are very satisfying.

REFERENCES

[1] Plestan F, Grizzle JW, Westervelt ER, Abba G. Stable walking of a 7-DOF biped robot. IEEE Transactions on Robotics and Automation. 2003 Aug;19(4):653-68.

[2] Hong YD, Lee B. Evolutionary Optimization for Optimal Hopping of Humanoid Robots. IEEE Transactions on Industrial Electronics. 2016 Sep 28.

[3] Wensing PM, Orin DE. High-speed humanoid running through control with a 3D-SLIP model. InIntelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on 2013 Nov 3 (pp. 5134-5140). IEEE.

[4] Tighzert L, Mendil B. Realization of Gymnastic Movements on the Bar by Humanoid Robot Using a New Selfish Gene Algorithm. In Modelling and Implementation of Complex Systems 2016 (pp. 49-65). Springer International Publishing.

[5] Xin X, Kaneda M. Swing-up control for a 3-DOF gymnastic robot with passive first joint: design and analysis. IEEE transactions on robotics. 2007 Dec;23(6):1277-85.

[6] Calinon S, Guenter F, Billard A. On learning, representing, and generalizing a task in a humanoid robot. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). 2007 Apr;37(2):286-98.

[7] Goodrich MA, Schultz AC. Human-robot interaction: a survey. Foundations and trends in human-computer interaction. 2007 Feb 1;1(3):203-75.

[8] Calinon S, Billard A. Incremental learning of gestures by imitation in a humanoid robot. InProceedings of the ACM/IEEE international conference on Human-robot interaction 2007 Mar 10 (pp. 255-262). ACM.

[9] Eiben AE, Smith JE. Introduction to evolutionary computing. Heidelberg: springer; 2003 Nov 1.

[10] Corno F, Reorda MS, Squillero G. The selfish gene algorithm: a new evolutionary optimization strategy. InProceedings of the 1998 ACM symposium on Applied Computing 1998 Feb 27 (pp. 349-355). ACM.

[11] Dawkins R. The selfish gene. Oxford university press; 2016 Jun 8.

[12] Ahn CW, Ramakrishna RS. Elitism-based compact genetic algorithms. IEEE Transactions on Evolutionary Computation. 2003 Aug;7(4):367-85.

[13] Liang JJ, Qu BY, Suganthan PN. Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore. 2013 Dec.

[14] Ariff NM, Khalid NE, Hashim R, Noor NM. Selfish Gene Algorithm Vs Genetic Algorithm: A Review. InIOP Conference Series: Materials Science and Engineering 2016 Nov (Vol. 160, No. 1, p. 012098). IOP Publishing.

[15] Derrac J, García S, Molina D, Herrera F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation. 2011 Mar 31;1(1):3-18.

[16] Ting WA. Contribution à la commande de robots marcheurs bipèdes (Doctoral dissertation, Université de Nantes).

[17] Vukobratović M, Borovac B. Zero-moment point—thirty five years of its life. International Journal of Humanoid Robotics. 2004 Mar;1(01):157-73.

[18] Mousavi PN, Bagheri A. Mathematical simulation of a seven link biped robot on various surfaces and ZMP considerations. Applied Mathematical Modelling. 2007 Jan 31;31(1):18-37.

[19] Jung Y, Jung M, Ryu J, Yoon S, Park SK, Koo S. Dynamically adjustable foot-ground contact model to estimate ground reaction force during walking and running. Gait & posture. 2016 Mar 31;45:62-8.

[20] Braun DJ, Mitchell JE, Goldfarb M. Actuated dynamic walking in a seven-link biped robot. IEEE/ASME Transactions on Mechatronics. 2012 Feb;17(1):147-56.