Evolutionary Gait-Optimization Using a Fitness Function Based on Proprioception*

Thomas Röfer

Center for Computing Technology (TZI), FB 3, Universität Bremen
roefer@tzi.de

Abstract. This paper presents a new approach to optimize gait parameter sets using evolutionary algorithms. It separates the crossover-step of the evolutionary algorithm into an interpolating step and an extrapolating step, which allows for solving optimization problems with a small population, which is an essential for robotics applications. In contrast to other approaches, odometry is used to assess the quality of a gait. Thereby, omni-directional gaits can be evolved. Some experiments with the Sony Aibo models ERS-210 and ERS-7 prove the performance of the approach including the fastest gait found so far for the Aibo ERS-210.

1 Introduction

In the Sony Four-Legged Robot League, two teams of four Sony Aibos each compete on a field of approximately 4.2 m × 2.7 m. For 2004, the two models allowed to be used are the ERS-210 and the ERS-7 (cf. Fig. 1a). As in all RoboCup leagues, speed is an essential factor in the game. Therefore, quite a lot of work has been done to optimize the gait of the Aibo using a variety of different approaches. In [3], a stationary state evolutionary algorithm was used to optimize the gait of the Aibo ERS-110, based on research done with Aibo's predecessor [4]. Thereby, a maximum speed of 100 mm/s was reached. On the ERS-210, Powell's method for multidimensional minimization was used to optimize the gait parameters [5]. The maximum speed reached was 270 mm/s. With an evolutionary hill climbing with line search approach, a parameter set resulting in a maximum speed of 296.5 mm/s was found [7]. In both latter works, the shapes of the foot trajectories were optimized. Using a variation of standard policy gradient reinforcement learning techniques, a maximum speed of 291 mm/s was reached, using a static, half-elliptical foot trajectory shape. In that paper it was also argued that the shapes of the foot trajectories sent to the robot significantly differ from the shapes actually performed, and therefore the adaptation of the shape of the locus is not necessarily important.

The typical training setup used in [4, 3, 5, 7, 6] was as follows: the robot walks from one side of the field the other one, then it turns on the spot and walks

* The Deutsche Forschungsgemeinschaft supports this work through the priority program "Cooperating teams of mobile robots in dynamic environments".

D. Nardi et al. (Eds.): RoboCup 2004, LNAI 3276, pp. 310–322, 2005.

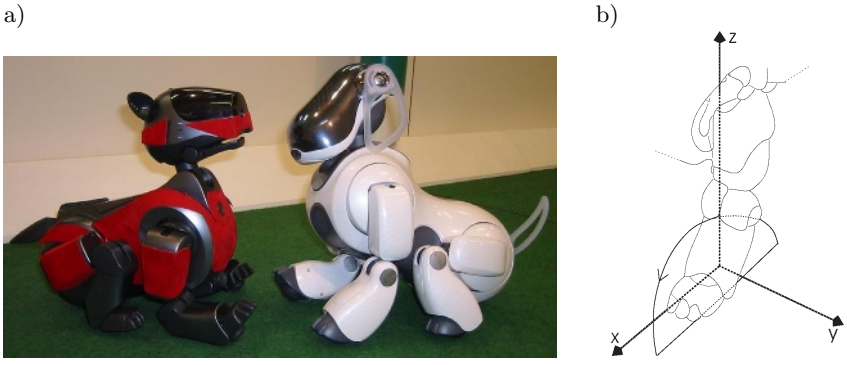

Fig. 1. a) The Aibos ERS-210 and ERS-7. b) A leg of an ERS-210 (modified from [6])

back to the first side, and so on. For orientation, it uses two beacons placed on both sides of the field (or color stripes in a pen in [4, 3]). The beacons are used to measure the speed of the walk as well as when to stop and to turn around. While in [4, 3, 5, 7] single robots are used, [6] trained three robots at once, thereby reducing the learning time by factor three. In [4, 3, 5, 7], the optimization was done on-board the robot. In contrast, [6] used an off-board computer for learning that controls the three robots used for evaluating different walk parameter sets.

2 Experimental Setup

The major drawback of the approaches described in the previous section is that they only optimize the gait for walking straight ahead with a more or less steady head. However, in most situations in actual RoboCup games, the robots have to perform different motions, especially when following the ball. They have to turn left and right while walking, move sideways and sometimes backwards. All these dynamic elements required for games cannot be learned when the robot only walks from beacon to beacon. Therefore, a more flexible scheme is required.

2.1 Measuring Robot Motion

So it must be possible to measure omni-directional robot motion. On the one hand, it is necessary to measure how good the motion performed matches the motion that was desired, on the other hand, the scheme must be simple enough to be performed at the actual competition site, because the carpet used there can differ from the carpet the robot was originally trained with in the laboratory.

Using a Self-Localization Method. All RoboCup teams have already implemented some kind of metric self-localization for their robots. They are either based on particle filters [10] or on extended Kalman filters [12]. However, both methods are based on an estimate of the actual motion of the robot, which is

used to bridge the time between the recognition of external references such as the beacons or the edges of the field [10]. This estimate is based on odometry. However, in contrast to wheeled motion, the four-legged robots typically use *forward-odometry*, i. e., the actual motion of the robot is not measured, but instead it is assumed that the motion requested is actually performed by the system. In fact, the walking engine is manually calibrated to execute the requested motions as precise as possible. Hence, forward-odometry just means to sum up the requested speeds for forward, sideward, and rotational motion, because it is assumed that this will also be the resulting motion. Although this assumption is too optimistic anyway, it is quite obvious that it will certainly be false if new walk parameters are tested. Therefore, using traditional self-localization methods to judge the quality of a new gait will fail.

Using an External Sensing System. Another possibility would be to use an external sensing system to measure the actual motion of the robot. This can either be an overhead camera as used in the Small-Size League, or a laser scanner, as used in [11]. However, both methods are not suitable for training at the actual competition site, because there may be no possibility to mount a camera above the Legged League field, and the fields cannot be reserved long enough to be exclusively used for training with a laser scanner that must always be able to see the robot.

Using Proprioception. The solution is to use real odometry. The Sony Aibo robot is equipped with sensors that can detect whether a foot touches the ground or not. In addition, it can measure the actual angles of all joints. Thus using forward kinematics, the actual motion of the feet can be calculated, especially the part of it, in which they have contact to the ground. If this is done for at least two feet, the actual motion of the robot can be reconstructed, as long as the feet do not skid over the ground. It is assumed that skidding feet will result in an uneven walk, which can be detected by another proprioceptive system of the Aibo: its acceleration sensors in x, y, and z direction.

2.2 Odometry

Since gaits using the typical PWalk-style originally introduced by the team from the University of New South Wales [2] are state of the art, such a kind of walk is also used for the work presented in this paper (cf. Fig. 3). This is important, because with this kind of walk, only the contact sensors of the hind legs touch the ground, and therefore, only these two sensors can be used for calculating the motion of the robot. The current position of such a ground contact sensor can easily be determined based on the measurements of the angles of the three joints the Aibo has in each leg using forward kinematics. These angles are automatically determined by the Aibo every 8 ms. The three-dimensional positions (cf. Fig. 1b) for the left and right hind feet change over time with the motion of the legs and can be computed relative to some origin in the body of the robot. However, for odometry, only their lengthwise and sideward components are required. Therefore, p_t^{left} and p_t^{right} are only two-dimensional vectors.

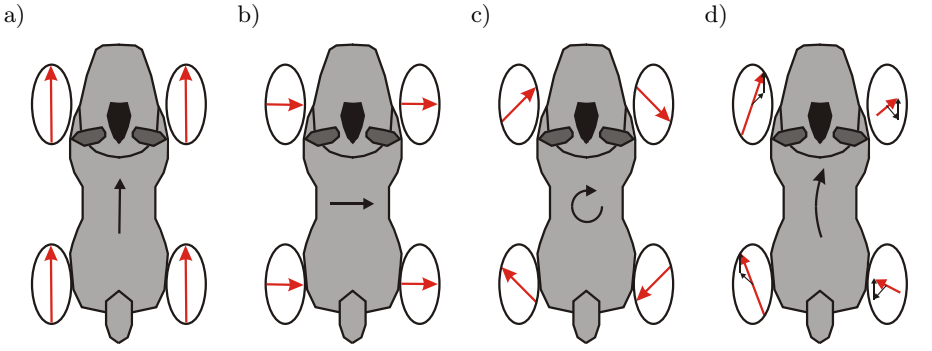

Fig. 2. Principle of treating legs as wheels [9]. Walking a) forwards, b) sideways. c) Turning. d) Turning while walking forward

Fig. 3. The Aibo ERS-210 walking with 311 mm/s

The overall offset a hind leg h has walked during a period of time $t_a \ldots t_b$ can be determined by summing up all the 2-D offsets between successive positions at which the foot had contact to the ground:

$$d^h_{t_a,t_b} = \begin{pmatrix} dx^h_{t_a,t_b} \\ dy^h_{t_a,t_b} \end{pmatrix} = \sum_{t=t_a+1}^{t_b} \begin{cases} p^h_t - p^h_{t-1} & \text{, if } g^h_t \wedge g^h_{t-1} \\ 0, & \text{, otherwise} \end{cases} \quad (1)$$

g just states whether the ground contact sensor of leg h was active at time t or not. Since each leg has sometimes contact to the ground and sometimes not while walking, it is important to always determine the distance with ground contact for a complete step phase. Otherwise, the distances measured by the left and the right feet are not comparable.

For quadruped walking, the three requested motion speeds x^r, y^r, and θ^r are overlaid to result in the actual x^h and y^h amplitudes of the legs. The walking engine of the GermanTeam [9] uses the following equations for the two hind legs. Please note that the feet move always in the opposite direction of the robot's body, so everything is negated (r is the radius to the body center):

$$x^{left} = -x^r + r\theta^r \tag{2}$$
$$y^{left} = -y^r + r\theta^r \tag{3}$$
$$x^{right} = -x^r - r\theta^r \tag{4}$$
$$y^{right} = -y^r + r\theta^r \tag{5}$$

For odometry, everything is the other way round. The foot motion is known, and the speed components have to be calculated. By transforming the motion equations, the measured walking speeds x^m, y^m, and θ^m can be calculated from the measured ground contact distances d^{left} and d^{right}. Please note that dx and dy measure only half of a step phase (while the corresponding foot touches the ground), so everything measured is multiplied by 2, i. e. all divisions by 2 are missing:

$$x^m_{t_a,t_b} = -\frac{dx^{right}_{t_a,t_b} + dx^{left}_{t_a,t_b}}{t_b - t_a} \tag{6}$$
$$y^m_{t_a,t_b} = -\frac{dy^{right}_{t_a,t_b} + dy^{left}_{t_a,t_b} + dx^{right}_{t_a,t_b} - dx^{left}_{t_a,t_b}}{t_b - t_a} \tag{7}$$
$$\theta^m_{t_a,t_b} = -\frac{dx^{right}_{t_a,t_b} - dx^{left}_{t_a,t_b}}{r(t_b - t_a)} \tag{8}$$

3 Evolutionary Algorithm

In evolutionary algorithms, a *population* of *individuals* is used to find a solution for a given problem. Each individual has a number of *genes* g_i that represent possible values for the variables of the problem. In each evolution step, the *fitness* f is determined for each individual, i. e., the current values of its genes are used to solve the problem and the solution is assessed by a so-called fitness function. Then, based on the current population and the fitness of its individuals, a new generation is created that replaces the current one. There are two major methods to generate individuals for the next population that can be combined in a lot of different ways: *mutation* and *crossover*. Mutation just adds noise to the genes of an individual and thereby creates a new one. In the crossover, the genes of two individuals are combined two a new one. There exist quite a lot of methods to select individuals for mutation or crossover [1].

In the work presented here, the genes represent the parameters of a gait. The fitness cannot directly be calculated, instead it has to be determined based on measurements, i. e. the gait parameter set has to be tested by letting the

robot actually walk with these parameters. Thus, the fitness for a certain set of parameters has to be assumed to be noisy, so each time it will be measured it will be slightly different. This is especially true for learning omni-directional walking, because it is impossible to test all possible combinations of walking directions and rotation speeds. Therefore, the selection of individuals is based on a probabilistic approach. The ratio of an individual fitness to the overall fitness of the whole population is interpreted as the probability that the parameter set represented by this individual is the best one. So based on this interpretation, individuals are *drawn* from a population to form a new one. For each individual in the new population, two individuals are drawn from the previous one, and their genes are crossed. If a new population consists to more than 50% of a single individual from the previous population, every second individual is mutated.

The initial population is generated from a single individual (a hand-coded gait parameter set) by mutation. In addition, whenever the battery of the robot is empty, the best individual found so far is used to form the basis for the next evolution run.

Mutation. The mutation performed is rather simple. For each gene (i. e. gait parameter) a mutation radius is defined that has a reasonable size with respect to the possible range of that parameter. The genes are then mutated by adding random values within their mutation radius to them.

$$g_i^{new} = g_i^{old} random_{-r_i \ldots r_i} \tag{9}$$

Crossover. As a new approach to evolutionary algorithms, there are two different methods for crossover: the *interpolation* and the *extrapolation*. They are used alternately, so one generation is created by interpolating between the individuals of the previous one, and the next one results from an extrapolation between the individuals of its predecessor. For the interpolating step, two individuals a and b are selected based on their fitness f and a new one is created by a random interpolation between their genes:

$$\begin{aligned} g_i^{new} &= \beta g_i^a + (1-\beta) g_i^b \\ &\text{where } \alpha = random_{0\ldots1} \\ &\beta = \tfrac{\alpha f_a}{\alpha f_a + (1-\alpha) f_b} \end{aligned} \tag{10}$$

While the idea of the interpolating step is to find a solution that is between two individuals, the extrapolating step shall find a solution that is outside. The difference between the genes of two individuals is used to extrapolate to one or the other side. The amount and direction of the extrapolation is again biased by the fitness of the two individuals selected:

$$\begin{aligned} g_i^{new} &= \begin{cases} g_i^b + (1-\beta)(g_i^a - g_i^b) \text{ , if } \beta < 1 \\ g_i^a + (\beta - 1)(g_i^b - g_i^a) \text{ , otherwise} \end{cases} \\ &\text{where } \alpha = random_{0\ldots1} \\ &\beta = \tfrac{2\alpha f_a}{\alpha f_a + (1-\alpha) f_b} \end{aligned} \tag{11}$$

That way, a solution can be found by alternately interpolating and extrapolating—without disturbing mutations. However, if a single individual is drawn too often, neither interpolation nor extrapolation can generate different sets of genes. Therefore, mutation is applied in such a case.

3.1 Fitness Function

The fitness function assessing the individual gait parameter sets is the sum of two values. On the one hand, it is determined, how good the motion performed matches the motion requested, and on the other hand, the measurements of the acceleration sensors are used to judge the smoothness of the gait.

Error Function. For each step, the motion actually performed is determined based on the calculations described in Section 2.2. However, instead of just using the odometry offset, a special kind of walk is forced by extending equation (1). In addition to determining the offset a foot moved on the ground, also the offset is calculated that the foot has moved while the *other foot* was on the ground. This second offset is negated and the average of both is used as a replacement for the original equation for the foot motion. This kind of fitness function enforces that when one foot is on the ground, the other one will swing in the opposite direction. Otherwise, the feet may waste time during their air phase by moving in the wrong direction.

$$d'^{h}_{t_a \ldots t_b} = \sum_{t=t_{a+1}}^{t_b} \frac{d_t^{h,h} - d_t^{h,other(h)}}{2} \quad \text{where } d_t^{h,h'} = \begin{cases} p_t^h - p_{t-1}^h & \text{, if } g_t^{h'} \wedge g_{t-1}^{h'} \\ 0 & \text{, otherwise} \end{cases} \tag{12}$$

d' is used instead of d in the equations (6) to (8), resulting in the measured walking speeds x'^m, y'^m, and θ'^m. They are compared to the requested walking speeds x^r, y^r, and θ^r. These are determined by summing up all the requested speeds for the same whole step. The error w is determined as follows:

$$w_{t_a \ldots t_b} = |x'^m_{t_a \ldots t_b} - x^r_{t_a \ldots t_b}| + |y'^m_{t_a \ldots t_b} - y^r_{t_a \ldots t_b}| + \alpha |\theta'^m_{t_a \ldots t_b} - \theta^r_{t_a \ldots t_b}| \tag{13}$$

α is a weight that relates errors made in translation to the deviations in rotation. If more than one step is used for the assessment (which is normally the case), the average error is used.

Vibration. The acceleration sensors a^x, a^y, and a^z of the Aibo measure accelerations along the x, y, and z axes of the robot. The amount of vibration during a walk can easily be determined by calculating the standard deviation of the measurements:

$$v_{t_a \ldots t_b} = stdv(a^x_{t_a \ldots t_b}) + stdv(a^y_{t_a \ldots t_b}) + stdv(a^z_{t_a \ldots t_b}) \tag{14}$$

Fitness of an Individual. The overall fitness of an individual can then be computed as a sum of error and vibration. However, since small errors and few vibrations are desired, both values have to be negated. In addition, they have to be scaled to relate them to each other (by ϕ, β, and $gamma$):

$$f_{t_a \ldots t_b} = e^{\phi - \beta w_{t_a \ldots t_b} - \gamma v_{t_a \ldots t_b}} \tag{15}$$

4 Results

Experiments were conducted both with the ERS-210 and the ERS-7. As the ERS-210 was used as test bed in some recent work [5, 7, 6], a similar experiment is described here in detail. However, as another walking engine was used, the 23 parameters evolved are different. In fact, 26 parameters were evolved, but three of them are only relevant if the robot is also moving sideways or it is turning. In the final experiment conducted with the ERS-7, they were also used.

As in [6], a fixed half-ellipsoidal locus shape was used. The parameters are:

- The relative positions of the fore/hind feet as (x, y, z) offsets
- The height of the steps performed by the fore and hind legs
- The tilt of the locus (cf. Fig. 1b) of the fore and hind feet
- The timing of a step. The overall duration of a step is separated into a ground phase, a lift phase, an air phase, and a lowering phase. Since all phases sum up to 1, one of them can be calculated from the others and is left out. Please note that for the half-ellipsoidal shape of the locus used, the lifting and lowering phases are just waiting positions for the feet. As it will pointed out later, this does not mean that the feet actually stop during these phases, because the actual motion of the feet is different from the motion that they were requested for.
- The step size determines the amplitude of a step (here: only in x direction).
- The duration of a step (in seconds)
- The difference between the speed of the forefeet and the hind feed. In fact, this value modifies the amplitudes of the motion of the hind feet, so it is possible that the hind feet perform smaller steps that the front feet (here: only for x speed).
- A phase shift between the fore and the hind feet
- A body shift in x and y direction. This shifting is synchronous to the phases of walking, but also a phase shift can be defined. This allows the body to swing back and forth and left to right while walking.

4.1 ERS-210

The ERS-210 was trained to walk forward as fast as possible. As in [6], a population of 15 individuals was used. However, each parameter set was only tested for five seconds, the last three seconds of which were used to assess the performance of the resulting walk. The first two seconds were ignored to eliminate any influence of (switching from) the previous gait parameter set. The robot walked

Table 1. The parameters evolved for the ERS-210

Parameter	Initial Value	Best Value
Front locus		
z offset	76.85	58.97
y offset	78.1	67.24
x offset	55.88	63.26
step height	5.0	6.14
tilt	-0.25	-0.48
ground phase	0.5	0.64
lift phase	0.06	0.04
lowering phase	0.06	0.12
Step		
size	76.0	89.77
duration	0.64	0.512
Rear to Front offsets		
x speed ratio	1.1	1.0
phase shift	0	-0.012
Rear locus		
z offset	108.72	111.54
y offset	76.95	58.09
x offset	-45.216	-35.96
step height	24.0	27.27
tilt	0.05	0.12
ground phase	0.5	0.34
lift phase	0.06	0.19
lowering phase	0.06	-0.02
Body shift		
x ratio	0	1.12
y ratio	0	-0.10
phase offset	0	0.35

from goal to goal and was turned manually. To avoid assessing the time when the robot was picked up, learning was interrupted when the back switch of the robot was pressed, and the assessment of the current individual was restarted when the button was released. To match the conditions in the related work, the robot's head did not move.

Learning was started with a variation of the hand-tuned gait of the German-Team 2003 [9]. The middle position of the hind feet was manually moved to the front to let the foot sensors have contact to the ground. The initial values of all parameters are given in Table 1. From this initial setup, the gait was learned automatically. After about three hours, the fastest gait found by the robot had a speed of 311±3 mm/s (13.5 s for 4.2 m from goal line to goal line on a Sony Legged League field), which is also the fastest gait found so far for the Aibo ERS-210. In addition, the gait is quite smooth, i. e. the head of the robot does not shake significantly. Under [8] two videos can be downloaded showing the Aibo walking from the front and from the side. Six images from the latter video are depicted in Figure 3, showing a single step phase.

In addition, Figure 4 shows the trajectories of the feet and the knees of the robot. Please note that for the front legs (on the right in both subfigures), the robot is walking on its knees. Thus the feet never have contact to the ground. In [5, 7] it is analyzed, which shape of a foot trajectory is optimal. However, the resulting trajectories of the feet were never investigated in that work. Figure 4 shows that there is a severe difference between the shape of the locus that was desired (half-elliptical), and the resulting shape. While the knees follow the given trajectory quite well (in fact they never reach the maximum front point and overshoot the maximum back point), the feet have very significant deviations. The resulting trajectory of the hind feet can be explained by over- and

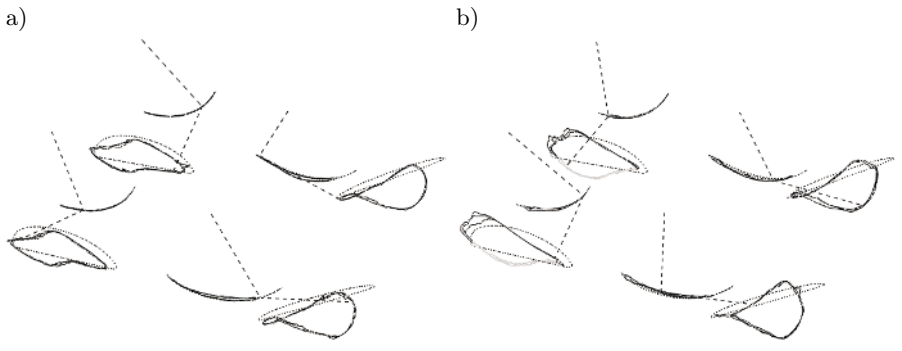

Fig. 4. The motion of the legs when moving with 311 mm/s (the orientation of the robot is the same as in Fig. 3). The dashed lines show a single snapshot of the positions of the legs. The dotted curves depict the trajectory of the feet and the knees as sent to the robot. The solid curves visualize the motion of the knees and the feet as measured by the joint sensors. a) Trajectories of a picked-up robot. b) Trajectories when walking on the ground. When ground contact was detected, the solid curve is drawn in gray

undershooting the given trajectory (it overshoots less when the robot is picked up, cf. Fig. 4a), but the control system seems to have real problems in following the desired loci of the forefeet.

4.2 ERS-7

All other experiments were conducted with the new Aibo model ERS-7, because it will be used in RoboCup 2004. However, the results are preliminary, because the experiments were only conducted with the beta version of the control software at a stage in which also the porting of the code from the ERS-210 to the ERS-7 was not completely finished. A major problem with the ERS-7 was that it switched itself off if there was too much load on one of its legs, and that this condition could easily be reached. These problems were only solved in the second release of the final version of the operating system, which came to late to repeat the experiments for this paper. So training the ERS-7 was a little bit complicated, because sometimes it had to be restarted after a few minutes. However, quite impressive results were achieved.

Walking Straight Ahead. The experiment conducted with the ERS-210 was repeated with the ERS-7. The robot is much more powerful, so it was able to reach 400±3 mm/s using a gait with 2.3 steps/s. The general appearance of the gait is quite similar to the gait of the ERS-210.

Walking Straight Backward. Then, the ERS-7 was trained to walk straight backward using the same experimental setup as before. The resulting gait performs less steps per second (1.9) and the robot is significantly slower (294 mm/s). In this walk, the robot still walks on the knees of the fore legs and the hind feet

(which was more or less predefined by the hand-tuned walk that was used as a start). The result shows that walking on the knees in walking direction is superior to using the knees as hind legs (with respect to the walking direction). However, the speed of a gait only walking on the four feet still has to be tested, but the training may be complicated because such a gait it is less stable and the robot may often fall down.

Following the Ball. As the main goal of this work was to be able to train a gait under more realistic conditions, the ERS-7 was also trained while following the ball. In this setup, the ball was moved manually and the normal *go-to-ball* basic behavior from the GermanTeam's code was used to follow the ball. Again, each walking parameter set was switched every five seconds, and again, only the last three seconds were assessed. Learning was only interrupted when the motion requested by *go-to-ball* was too slow, because in such cases any kind of walk parameter set would get a good assessment. The ball was moved in a way that resulted in a good mixture between walking straight (less often) and turning for the ball (more often, due to the small field). The gait parameters were initialized with the ones from the fastest straight ahead gait of the ERS-7. After about of three hours of training, the Aibo had learned a gait with 3.1 steps/s, a maximum forward speed of 331 mm/s, and a rotation speed of 195°/s. The sideward and backward speeds were low, because the *go-to-ball* behavior seems not the use these motion directions. Due to the very high step frequency, the gait is extremely reactive.

5 Conclusions and Future Work

This paper presented a new approach to optimize gait parameter sets using evolutionary algorithms. On the one hand, it is the first method that is potentially suitable to learn appropriate parameter sets for any motion required in RoboCup, on the other hand, separating the crossover-step of the evolutionary algorithm into interpolation and extrapolating proved to be a good means to optimize the parameter sets. As a result, the fastest gait of the Aibo ERS-210 known so far has been found, outperforming the runner-up by 5%. In addition, preliminary experiments with the new Aibo ERS-7 also showed promising results.

However, as has already pointed out in [5], the more general a gait is, the less fast it will be. Therefore, several parameter sets have to be combined, each for a certain range of walking directions and rotations. This way, the gaits evolved for the ERS-7 were combined and used by the *Bremen Byters* at the RoboCup German Open 2004. They turned out to be the fastest gaits for forward motion, backward motion, and rotation in the whole competition, although other teams had also experimented with evolutionary gait optimization. In addition, the odometry calculation presented in this paper replaced the original forward odometry, because it scales better with different walking speeds.

However, since they were trained independently from each other, the transition between them is not guaranteed to be smooth, i. e. the robot may stumble. For instance, although they have a different step frequency, switching between the fast forward gait and the ball follower gait of the ERS-7 seems to work quite fine, while the transition between the ball follower (forward) gait and the backward gait is uneven. Therefore, several parameter sets for different directions/rotations have to be learned at once with interpolations between neighboring sets. Thus the transitions between the sets should be smooth, but it is still possible to have specialized sets for certain motions such as walking straight ahead.

Acknowledgments

The author thanks all members of the GermanTeam for providing the basis for this research, especially Uwe Düffert and Max Risler for writing the walking engine this work is based on.

References

1. J. E. Baker. Adaptive selection methods for genetic algorithms. In J. J. Grefenstette, editor, *Proceedings of an International Conference on Genetic Algorithms and their Application*, pages 101–111, Hillsdale, New Jersey, USA, 1985. Lawrence Erlbaum Associates.
2. B. Hengst, D. Ibbotson, S. B. Pham, , and C. Sammut. Omnidirectional motion for quadruped robots. In A. Birk, S. Coradeschi, and S. Tadokoro, editors, *RoboCup 2001: Robot Soccer World Cup V*, number 2377 in Lecture Notes in Artificial Intelligence, pages 368–373. Springer, 2001.
3. G. Hornby, S. Takamura J. Yokono, O. Hanagata, T. Yamamoto, and M. Fujita. Evolving robust gaits with Aibo. In *IEEE International Conference on Robotics and Automation 2000 (ICRA-2000)*, pages 3040–3045, 2000.
4. G. S. Hornby, M. Fujita, S. Takamura, T. Yamamoto, and O. Hanagata. Autonomous evolution of gaits with the sony quadruped robot. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, Orlando, Florida, USA.
5. M. S. Kim and W. Uther. Automatic gait optimisation for quadruped robots. In J. Roberts and G. Wyeth, editors, *Proceedings of the 2003 Australasian Conference on Robotics and Automation*, Brisbane, Australia, 2003.
6. N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation*, May 2004. to appear.
7. M. J. Quinland, S. K. Chalup, and R. H. Middleton. Techniques for improving vision and locomotion on the sony aibo robot. In J. Roberts and G. Wyeth, editors, *Proceedings of the 2003 Australasian Conference on Robotics and Automation*, Brisbane, Australia, 2003.
8. T. Röfer. Videos of Aibo ERS-210. Only available under “Images and Videos” of http://www.robocup.de/bremenbyters.

9. T. Röfer, H.-D. Burkhard, U. Düffert, J. Hoffmann, D. Göhring, M. Jüngel, M. Lötzsch, O. von Stryk, R. Brunn, M. Kallnik, M. Kunz, S. Petters, M. Risler, M. Stelzer, I. Dahm, M. Wachter, K. Engel, A. Osterhues, C. Schumann, and J. Ziegler. GermanTeam, 2003. Technical report, only availabe at http://www.robocup.de/germanteam.
10. T. Röfer and M. Jüngel. Vision-based fast and reactive Monte-Carlo localization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA-2003)*, pages 856–861, Taipei, Taiwan, 2003. IEEE.
11. T. Röfer and M. Jüngel. Fast and robust edge-based localization in the Sony four-legged robot league. In *International Workshop on RoboCup 2003 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Padova, Italy, 2004. Springer. to appear.
12. C. Sammut, W. Uther, and B. Hengst. rUNSWift 2003. In *International Workshop on RoboCup 2003 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence, Padova, Italy, 2004. Springer. to appear.