29

Evolutionary Motion Design for Humanoid Robots

Toshihiko Yanase and Hitoshi Iba
The University of Tokyo
Japan

1. Introduction

The purpose of our research is to achieve a method of intuitive motion design that could be used by people who have no specialized knowledge of robotics in order to operate humanoid robots. Currently, as a result of the research conducted by Nishiwaki et al., stable walking of humanoid robots can be generated in real time (Nishiwaki et al., 2002). This method focuses primarily on walking, but it can easily be expanded to include any type of motion which is conducted on the same plane and with a level centre of mass. Nakaoka et al., using the method of walking motion generation devised by Nishiwaki, succeeded in capturing the motions of people dancing and enabling humanoid robots to mimic them (Nakaoka et al., 2003). In order to convert the motion expression obtained by capturing those motions into movement that could be realized by humanoid robots, they analyzed the primitive aspects of the leg motions. This made it possible to specify the primitive elements and the parameters, and to dynamically reproduce the leg movements performed in dancing in a stable manner. However, designing motion using the method devised by Nakaoka et al. requires a motion capturing system. It also requires people who can express the motions to be captured. Because of that, motion generation becomes a large-scale undertaking.

On the other hand, various methods have been proposed as intuitive methods for generating movements to be executed by humanoid figures in 3D-CG (3D computer graphics). These include the UT-Poser method proposed by Yamane and Nakamura. (Yamane & Nakamura, 2002), which uses pins to fix links and drags movable links, making it possible to generate any desired pose. There is also the Interactive Evolutionary Computation (IEC) (Takagi, 1998) method by Wakaki and Iba (Wakaki & Iba, 2002), in which any desired motion can be created simply by selecting a motion displayed on the screen. In motion design for humanoid robots, however, unlike in computer graphics, movements need to be generated that are actually feasible under the physical limitations imposed by the real world. For example, there are limits to the joint angles that can be expressed by humanoid robots, and when the robots come in contact with the ground, the soles of their feet have to be horizontal (we will explain the difficulties by examples in section 2.2). In order to apply motion generated using computer graphics with humanoid robots, this method has to be modified.

Source: Frontiers in Evolutionary Robotics, Book edited by: Hitoshi Iba, ISBN 978-3-902613-19-6, pp. 596, April 2008, I-Tech Education and Publishing, Vienna, Austria

In our paper, the desired motion is designed using IEC as an intuitive method of motion design for humanoid robots and is aimed at people with no specialized knowledge of robotics and with no large-scale equipment. The aim is to achieve stable motions by following the desired ZMP (i.e., zero moment points). Because the motion generated via IEC is not necessarily the optimum motion, some form of optimization is necessary. Particularly in cases where the motion involves contact with an object that produces a reaction force, such as kicking a ball, optimization is necessary because of the reciprocity that occurs between the ball travel distance and the stability of the robot. For instance, Wolff and Nordin proposed an EC-based learning method in order to acquire stable biped walking for a simulated humanoid robot (Wolff & Nordin, 2003). However, this method requires a specialized gait controller manually developed to produce the initial population. On the other hand, in our approach, we can create the initial population from the motions generated via IEC.

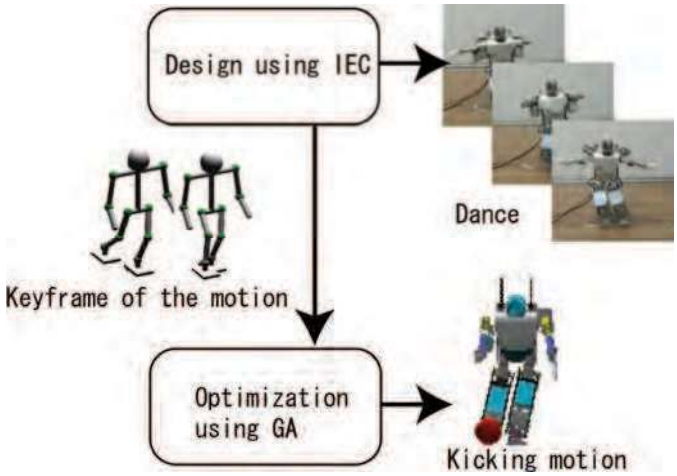


Figure 1. The flow of the experiment

In our research, we have used a dynamics simulator and carried out optimization using a genetic algorithm (GA) of the motions designed using IEC. Fig. 1 shows an overview of the experiment, which consists of two phases. In the first, IEC is used to evolve robot behaviours: users evaluate visually displayed robot motions that are generated with kinematic and stability constraints. In the second phase, GA is used to optimize the behaviour obtained from phase 1, by using the dynamics simulator.

This paper describes how successfully EC is applied to the generation of real motions for a humanoid robot. More precisely, we empirically show the following points by several experiments:

- IEC is effectively applied to designing the intuitive motions of a humanoid or a group of humanoids, e.g., kicking behaviour and cooperative dance.
- GA can be used to stabilize the motions generated by the above IEC-based method.
- The effectiveness of our approach is demonstrated by testing the generated motions with a real robot, i.e., HOAP-1.

The rest of this paper is organized as follows. In Section 2, we describe the experimental environment and the difficulties of generating humanoid's motions. IEC-based method is

proposed to avoid these difficulties. Their experimental results are shown in Section 3. Then, we explain how generated motions are optimized by means of GA in section 4. Thereafter, we discuss the results in Section 5 and give conclusion in Section 6.

2. Motion design for Humanoid Robot

2.1 Humanoid Robot and its Simulator

In our research, we have used the HOAP-1 (Humanoid for Open Architecture Platform) robot manufactured by Fujitsu Automation, as shown in Fig. 2. Motions were controlled by specifying the joint angles of the 20 joints of the entire body every 0.002 seconds. The characteristics of the HOAP-1 are noted below: (1) Height: 483 [mm], Weight: 5.9 [kg]. (2) The internal interface between the hardware and software is available for public use. (3) For movable parts, each leg had six degrees of freedom and each arm had four degrees of freedom, for a total of 20 degrees of freedom on the left and right sides. (4) The robot had the following functionalities: a joint angle sensor, a 3-axis acceleration sensor, a 3-axis gyro sensor and two foot load sensors.

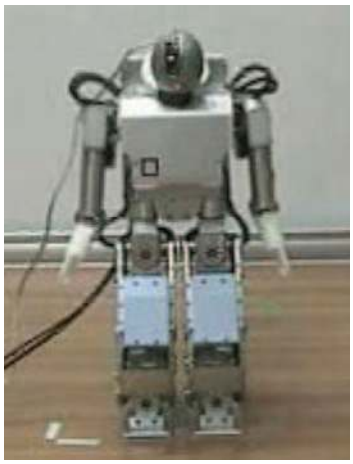


Figure 2. The HOAP-1

OpenHRP (Kanehiro et al., 2004) is used as the dynamics simulator. It is a software platform for humanoid robotics, and consists of a dynamics simulator, view (camera) simulator, motion controllers and motion planners of humanoid robots.

2.2 Difficulties of generating motions by using 3D-CG

When designing motions for a humanoid, it is essential to consider the contact with other objects and the external forces such as gravity so that the robots can move in a real environment. On the other hand, in case of 3D-CG humanoid figure, the motion design is more highly flexible, because the real-world constraints do not necessarily apply. Fig. 3 shows the typical design result of a kicking motion by means of 3D-CG, in which IEC was applied to generating keyframes of humanoid animations according to (Wakaki & Iba, 2002). We determined the motion of an avatar used in H-Anim in such a way that the amount of rotation of the joints could be obtained from the genes from IEC. In this design

example, we ignored gravity or forces of floor repulsion and fixed the position of the waist links. In the figures, (a) shows the keyframes of the designed kick motion. (b) gives the motion sequence between the first and second key frames, whereas (c) is the same motion sequence simulated considering the gravities and the repulsion force. As can be seen from the figures, the robot fell down when he raised his foot. This is because of the ignorance of external forces such as gravity or repulsion forces when designing motion by simulation. However, it is usually very difficult to include all the influences of these external forces beforehand for simple simulation. Therefore, the traditional 3D-CG technique has serious limitation to the application of designing robot motions in a real environment. The next section describes how IEC can generate stable motions by using ZMP calculation.

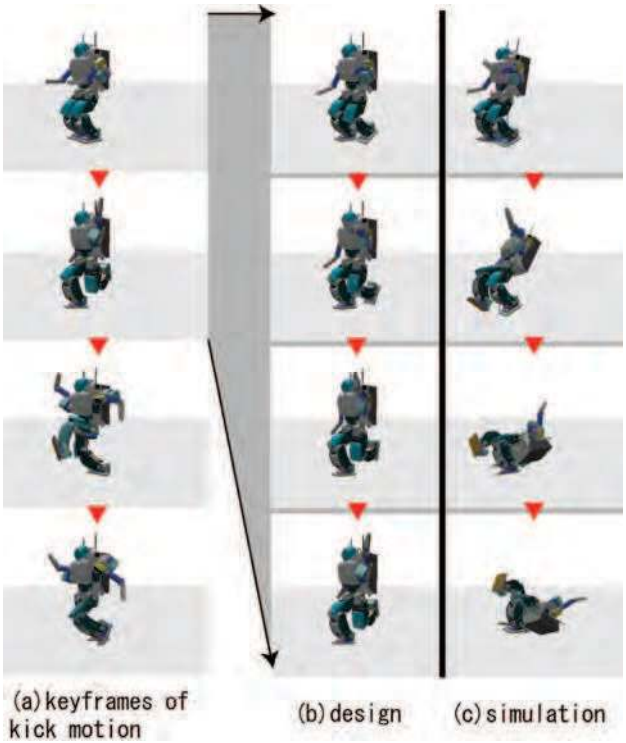


Figure 3. Applying 3D-CG method

3. Motion design using IEC

3.1 IEC-based motion design for a humanoid robot

Fig. 4 shows an overview of the motion design using IEC. The system provides the user with the motions generated using IEC. The design is created by the user looking at these motions and evaluating them. Everything that requires any specialized knowledge about humanoid robots, such as kinematics calculation, is done internally by the system, so the motions can be designed simply by having the user look at the screen and evaluate the motions.

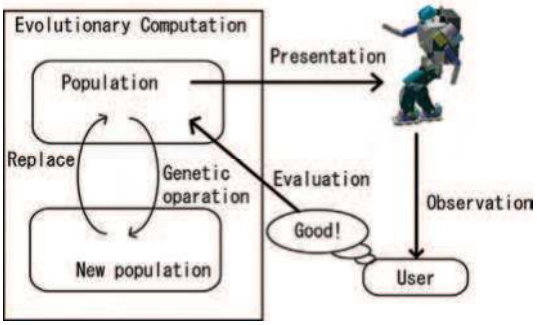


Figure 4. The IEC algorithm

3.2 Expressing motion

In our experiment, we have used the keyframe animation technique, which is often used with 3D-CG as a method of expressing motion. With keyframe animation, motion is expressed as the combination of a pose and a timeframe, and interpolation of the time between the poses is carried out to produce the animation. In (Wakaki & Iba, 2002), Wakaki et al. created the animation by creating the joint angle values for the entire body of the humanoid figure using interactive GA or interactive GP. With humanoid robots, if the joint angle values are set without taking the conditions of the support leg into consideration, there is very little possibility of generating an individual robot that will not fall down. With that in mind, we propose a method for specifying poses and an interpolation method, which together satisfy the conditions such that the humanoid robot will not fall down.

3.3 Pose Definition

Let us consider the process of assigning numeric values to poses in terms of keyframes. The weight of the robot has to be supported, and also important is the question of how to express the positions and attitudes of legs, because they might bump into or interfere with each other. If the poses likely to be taken by the humanoid robot are grouped based on the relationship of the feet to the floor, there are three cases to be considered: when both feet are in contact with the floor, and when one foot (right or left) is on the floor. If the position of the ankle of the support leg is determined, the state positions and attitudes that will keep the humanoid robot from falling over are limited, so the position and orientation of the ankle of the support leg should be set as the reference for poses of the entire body.

The position for the landing point of the support leg, as shown in Fig. 5, is decided based on the polar coordinates (r,θ) that use the support leg ankle position from the prior keyframe as a reference. The orientation is determined by the parameter ϕ. In order to prevent interference between the feet, the origin of the coordinates is offset from the ankle position of the support leg by the amount of the waist link, perpendicular to the orientation of the ankle. However, because the size of r that can be realized by means of θ is different, r is defined within a range solved by inverse kinematics. The ankle attitude is specified such that the sole of the foot is horizontal. This parameter is effective in cases when the support leg is changing, such as when shifting from one leg on the ground to both feet on the ground, or from the right foot on the ground to the left foot on the ground.

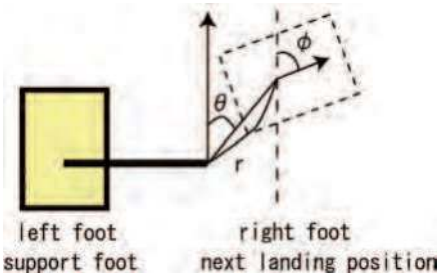


Figure 5. Specifying the ankle position of the support leg

The ankle of the swing leg is specified as shown in Fig. 6. The position of the ankle is specified using a cylindrical coordinate system (r, θ, h). In this coordinate system, the origin is the position offset from the support leg ankle position by the amount of the waist link, perpendicular to the orientation of the ankle. Additionally, the ankle attitude is specified based on the roll angle, pitch angle and yaw angle. If the two feet are close together, or if the position of the swing leg ankle is near the floor, the feet may bump into each other in some cases, depending on the attitude of the ankle. In a case such as this, we restricted the attitude of the ankle to avoid collision. If both feet are in contact with the ground, this parameter is invalid.

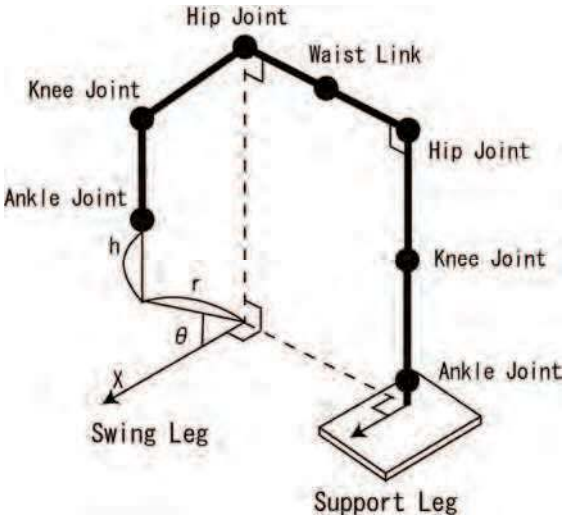


Figure 6. Specifying the position of the swing leg ankle

For the arms, which do not have to support the weight of the humanoid robot, we specified an arm joint angle with a total of eight degrees of freedom for the left and right arms together. For the waist link, we specified the height and the attitude.

The attitude was provided by the roll angle, pitch angle and yaw angle, using the support leg ankle link as a reference. When both feet were in contact with the ground, the ankle link of the right foot was used as a reference.

Table 1 shows a summary of the parameters used to decide the poses. Maximum and minimum values were determined for each parameter in advance, and these were handled by being normalized at [0, 1]. Because stabilization was the highest priority, the maximum and minimum values of the elements were set lower than the limit values.

	Element name	Degrees of freedom
(a)	Type of pose	1
(b)	Landing position and direction of support leg	3
(c)	Swing leg ankle-link position	3
(d)	Swing leg ankle-link attitude	3
(e)	Arm joint angle	8
(f)	Waist-link height	1
(g)	Waist-link attitude	3

Table 1. Pose parameters

3.4 Converting from a pose to motion

For the arms, which do not need to support the weight of the robot, the joint angles were smoothly interpolated using the natural cubic spline method. In converting the poses to motion, the interpolation has to be carried out under the condition that the calculation ZMP is not outside the actual support polygon, as explained below.

The centre of pressure of the floor reaction force in that state is called the ZMP (Vukobratovic & Stepanenko, 1972). A "support polygon", as shown in Fig. 7, can be defined as a convex closure that is the convex hull formed by the set of contact points of the robot and the floor. The left part of Fig. 7 shows the case in which both feet are on the floor, and the right part shows one foot in contact with the floor.

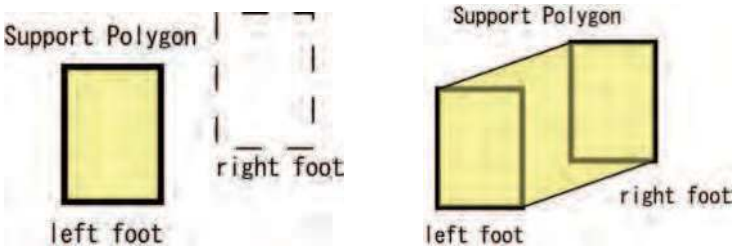


Figure 7. Support polygon

If the ZMP ends up being outside the actual support polygon, however, the robot will fall down. With that in mind, in order to prevent the robot from falling down in our experiment, we can correct the provided poses when interpolating them so that the ZMP resulting from the calculation of the physical model would not be outside the support polygon. The leg motions are derived according to the following sequence:

- Decide the landing position
- Derive the centre of mass position by the desired ZMP
- Calculate the support leg joint angles

First, the landing position of the support leg is decided. This derivation is based on (a) Type of pose and (b) Landing position and direction of the support leg. When the landing position is decided, the support polygon is also calculated. When one foot is on the floor

(Figure 8), the desired ZMP is taken as the centre of mass position of the support polygon. The waist-link height is the same as the one in the previous keyframe, and the waist-link attitude is upright. Next, the ankle position and attitude of the swing leg are decided based on (c) Swing leg ankle-link position and (d) Swing leg ankle-link attitude.

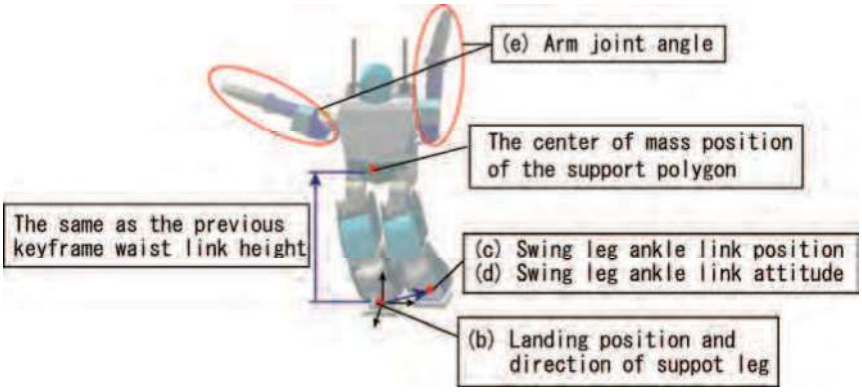


Figure 8. Converting from a pose to motion: when one foot is on the floor

When both feet are in contact with the floor (Fig. 9), the desired ZMP is decided based on (f) Waist-link height. Next, the waist-link attitude is derived from (g) Waist-link attitude.

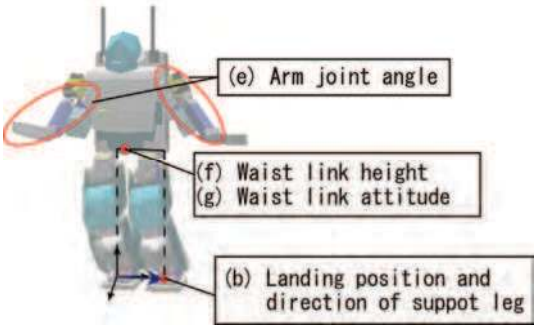


Figure 9. Converting from a pose to motion: when both feet are on the floor

3.5 Dynamic Balance

In order to determine the centre of mass trajectory that will satisfy the desired ZMP trajectory, as in the motion generation method used by Nakaoka et al. (Nakaoka et al., 2003), our method uses the fast generation method of motion pattern that follows the desired ZMP proposed by Nishiwaki et al. (Nishiwaki et al., 2002).

On a discrete system, supposing all the segments are restricted to be translated horizontally in the same distance, the following equation is acquired:

$$x_{\mathrm{zmp}}(t_i) = \frac{-hx(t_{i+1}) + (2h + g\Delta t^2)x(t_i) - hx(t_{i-1})}{g\Delta t^2}, \tag{1}$$

where x_{zmp} is a difference between a calculated ZMP and a desired ZMP, x is a translation distance of centre of mass to realize the desired ZMP, t_i is time at frame i, h is height of centre of mass, Δt is time per one frame. This equation is about x-axis, and similar equation applies to y-axis. This equation is expressed by information from 3 consecutive frames. These kinds of equations are solved as tridiagonal simultaneous linear equations. This method cannot figure out a result which completely follows the desired ZMP trajectory in one calculation because the constraint that all the segments translate parallel in the same distance is actually impossible. However, by iterating the calculation, a converged result is acquired.
After that, the waist-link position is calculated from the centre of mass position. The support leg joint angles are decided using inverse kinematics so that the relationship between the waist-link position and ankle position is satisfied.

3.6 Setting the evolution calculation

When carrying out motion design using IEC, it is conceivable that the poses and times for all of the keyframes could be searched for at one time, but the large number of degrees of freedom in the joints of humanoid robots may make the search range too huge. For this reason, we progressively generate each pose and time by IEC, according to the following method.

	Element name	Degrees of freedom
(c)	Swing leg ankle-link position	3
(d)	Swing leg ankle-link attitude	3
(e)	Arm joint angle	8
(f)	Waist-link height	1
(g)	Waist-link attitude	3

Table 2. Elements of GTYPE: IEC

The real-valued GA is used to optimize the 18 parameters (shown in Table 2) of the pose of the final time. Thus, GTYPE consists of 18 real values. In most cases, the landing position can be determined from the travelling direction of the robot. Deciding the landing position by IEC prevents the convergence of the other parameters. We indicate the parameters of the landing position by using a simple GUI instead of IEC. The position (r, θ) can be specified by clicking in the predetermined feasible area, and the direction (ϕ) can be changed by dragging the footstep.
The motion calculation is carried out as follows:

1. A new keyframe is added to the motion, and the pose is randomly generated.
2. The poses are converted to the motion.
3. The motion is shown to the user.
4. The user evaluates the individuals. The design is terminated, if a sufficient pose is obtained.
5. The next-generation is created. Then, return to step 2.

Figure 10 shows a snapshot of our IEC-based motion design system. The user looks at the motions displayed on the screen and uses the slider under each pose to provide an evaluation value. In the displayed motions, the calculated joint angles have been played back using forward kinematics, and are not the result of a dynamics simulation.

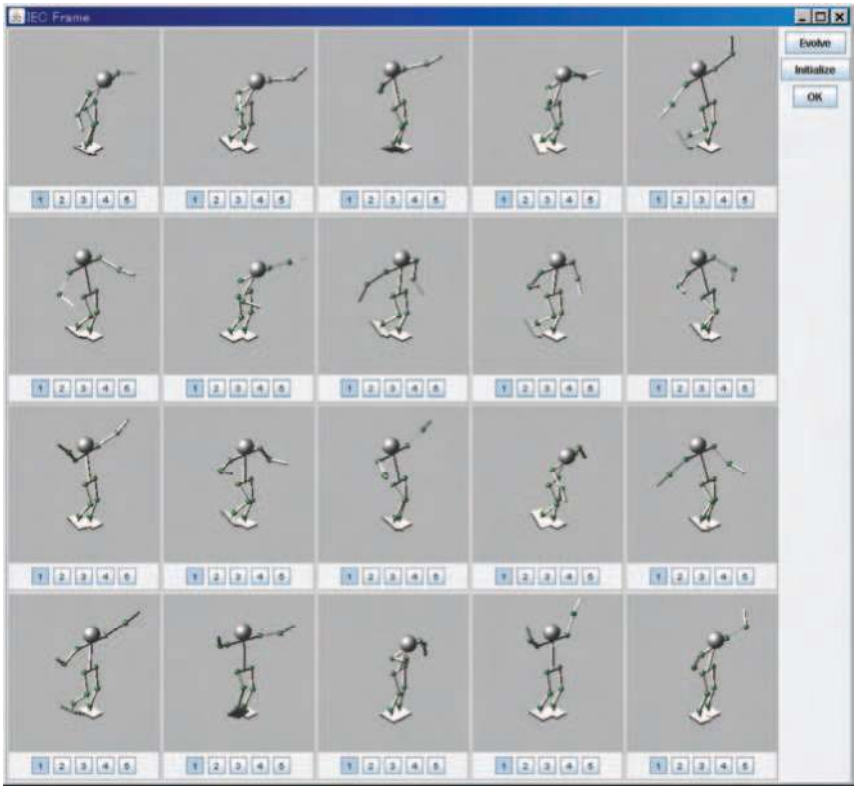


Figure 10. Evaluation screen for IEC

3.7 Stability of generated motions

We provide a stability comparison between the proposed method and the 3D-CG method described in section 2.2. In this experiment, randomly generated motions were evaluated in the dynamics simulator. The motions were expressed by three keyframes. The first and third keyframes were fixed upright postures, and only the second keyframe was randomly generated. After interpolating the keyframes, the motions were evaluated by whether the robot fell down or not in the dynamics simulator.

Method	Number of success	Number of failures	Success rate (%)
3D-CG method	9	991	1
Proposed method: total	974	28	97
Proposed method: both feet	326	8	98
Proposed method: right foot	326	8	98
Proposed method: left foot	322	12	96

Table 3. Success rate of randomly generated motions

The experimental result is shown in Table 3. The success rate of the motion generated by 3D-CG method is very low. So if 3D-CG method is applied to motion design for a humanoid robot, all the individuals must be inspected using the dynamics simulator. Moreover, it is usually very difficult to evolve such highly-fatal individuals using ordinary Evolutionary Computation.
On the other hand, the proposed method generates stable motions with a much higher rate than the 3D-CG method. ZMP errors and interferences during the interpolation period between the keyframes were observed as causes of the failed motions for the proposed method. Both of them can be detected during the interpolation stage. Therefore, the proposed method enables the effective generation of stable motions.

3.8 Evolved behaviours of a humanoid robot

Figure 11 shows the simulation result for a kicking motion. The elements that were searched for were only the types of poses and the position of the right ankle, which was the swing leg. All the other parameters were fixed. Moving slowly, the robot was able to achieve a kicking motion without falling down, although the ball travelled only a short distance. With the kicking motion, the robot came in contact with the ball as well as with the floor, so there was offset between the ZMP that was obtained through the calculation using IEC and the actual ZMP. The stronger the reaction force from the ball, the larger the offset became.

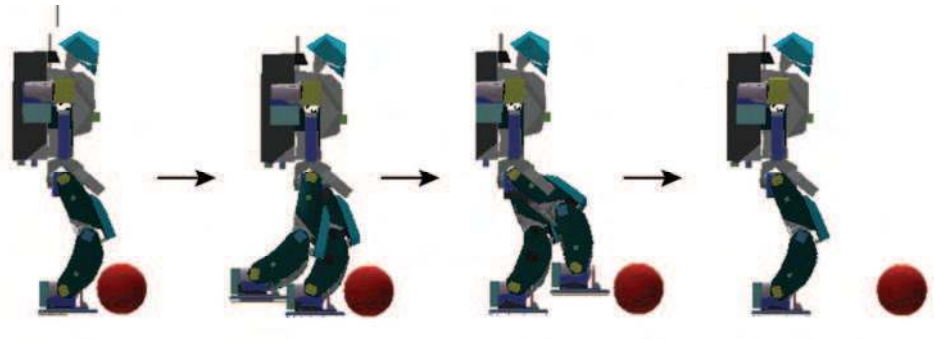


Figure 11. Example of motion design: Kicking

Another example is the cooperation dance of multiple robots. The dance of each robot was created with the above-mentioned method (see Figure 12). Although the robot moved slowly, the dance, which included the elements of a shift in the centre of mass and a tilting of the upper body, was carried out by the actual robots. With the simulator, the generated motion was stable without adding any particular changes. With the actual robots, however, when the dance was executed by the three HOAP-1 robots, one of the robots was unstable when lifting its feet. When the motions were modified so that the feet were not lifted as high, all three robots demonstrated the stable motion shown in Fig. 13. The learning process of cooperative behaviours among humanoid robots is described in details in (Inoue et al., 2004, 2007).

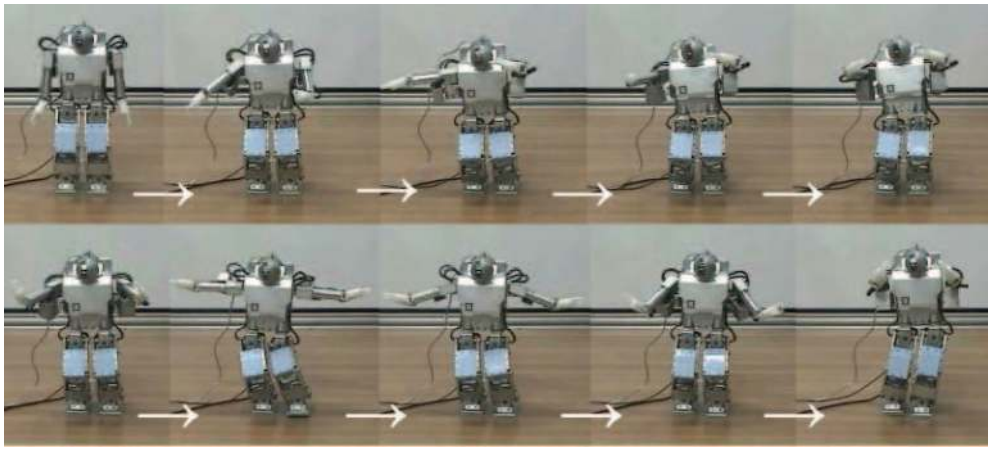


Figure 12. Example of motion design: A single dance

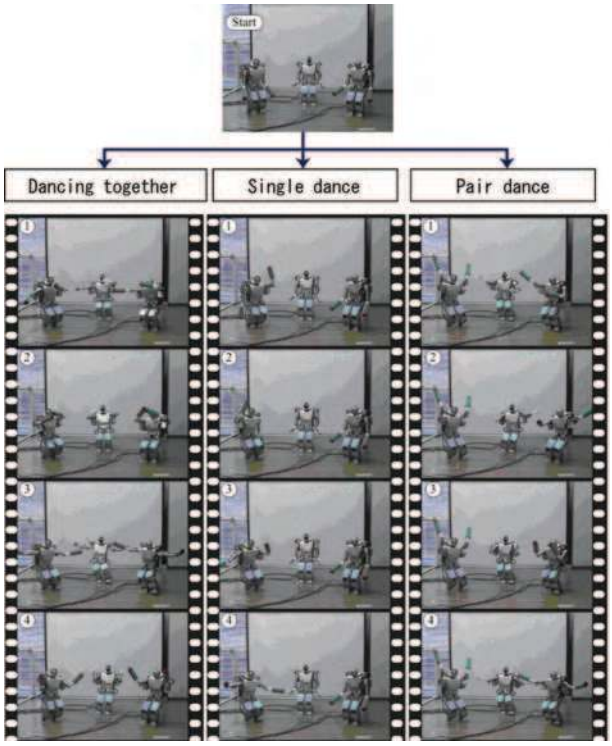


Figure 13. Example of motion design: Cooperative dance

4. Motion optimization using a GA

In the previous section, motions were generated through IEC according to the relation between the support foot and ZMP. However, generated motions were not necessarily stable. This is because the precondition for the stability, i.e., supporting polygon or external forces, is dynamically changing, especially when a robot is in contact with some object which is not a floor. In order to solve this difficulty, we employ real-valued GA for the sake of optimizing the generated motions in terms of the stability. This section describes two successful examples, i.e., optimizing a kicking motion and a sitting motion.

4.1 Optimizing a sitting motion

In order for a humanoid robot to sit on a box from a standing position, he has to move his gravity centre from his foot back to the contact place of the box and his waist link. Thus, it is very difficult to design by using only IEC.

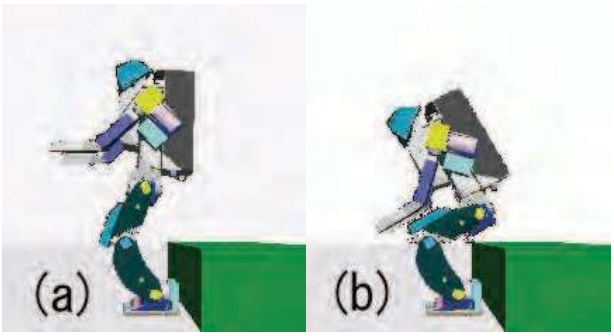


Figure 14. Sitting motion: Initial Pose, Final Pose

Suppose that two keyframes shown in Fig. 14 have been generated from IEC. Fig. 14 shows an initial standing position, whereas a robot finally bends his hip and knee joints at 90 degrees in Fig. 14.

We use GTYPE encoding the following items:

- the angles of the hip joint, the knee joint and the ankle joint for the intermediate position and the final position
- the time step for the intermediate position
- the time step for the final position

The fitness value is derived in the following way:

$$fitness = \exp(-\sin(\theta_{\max})) + \exp(-V_{\max}) + 2\exp(-r), \tag{2}$$

where $\theta_{\max}$, $V_{\max}$, r are defined as follows:

- Maximum lean of chest link during the motion ($\theta_{\max}$)
- Maximum velocity of chest link during the motion ($V_{\max}$)
- Distance between waist link and surface of box (r)

The terms of $\exp(-\sin(\theta_{\max}))$ and $\exp(-V_{\max})$ are expression for evaluating the stability, whereas $\exp(-r)$ represents how successfully the task is achieved. In the above definition, $\exp(-r)$ is multiplied by two for the purpose of equalizing the two evaluation criteria.

Fig. 15 shows the evolved motion in a typical run. As can be seen from the third and fourth snapshots, the robot turns his angle joint and bends his body so that he can sit while keeping his gravity centre within the support polygon. The fourth and fifth snapshots show the contact of the corner of the waist parts with the box. In the fifth and sixth snapshots, the robot moves its gravity centre to the box while turning round the corner of the waist parts, as a result of which the robot can successfully achieve the sitting task.

Figure 16 plots the waist position in x and z coordinates for the best evolved individual and the linear interpolation method. The linear interpolation is commonly used in robotics for the sake of interpolating poses, e.g., Sony SDR-4X (Kuroki et al., 2003). Initial position corresponds to the upper right corner, whereas final position is on the lower left corner. Dots are plotted every 0.2 second so that the slower the moving velocity, the narrower the interval between the two dots.

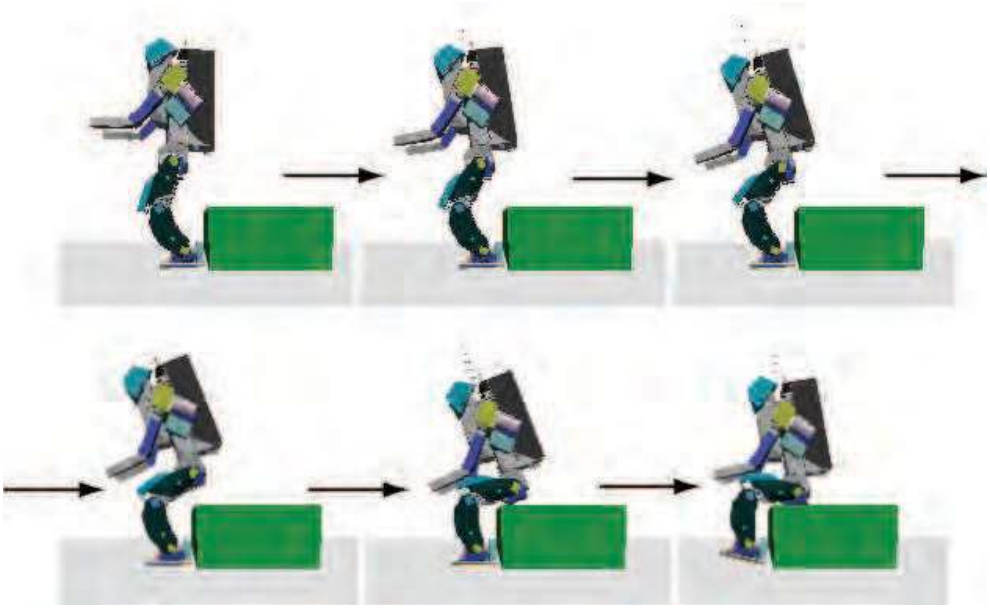


Figure 15. Sitting motion: best individual

Note that the slope of the best individual's curve is decreasing, which means that robot motions are changing gradually from vertically to horizontally. In the left corner, the slope is suddenly changed. This is the branching point when the robot moves its weight from the support polygon of its foot back to the box surface. The vertical distance of the waist link at that time was less than 5[mm] for the best individual, whereas it was about 20[mm] for the linear interpolation. The intervals between two dots are relatively wider for the linear interpolation, which means the motion velocity is faster. On the other hand, the narrow

intervals of the best individual reflect the slow motions, which can be observed in the fifth and sixth snapshots in Fig. 15.

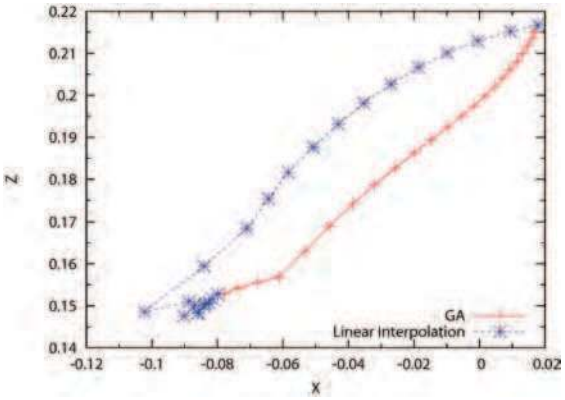


Figure 16. Sitting motion: Position of waist link

4.2 Optimizing a kicking motion

The kicking motion created using IEC in the previous section was carried out slowly, so that the ball did not travel a great distance.

In order to increase the distance travelled by the ball, the speed at which the tip of the foot is moving at the instant it contacts the ball has to be increased. However, the following elements, which reduce the stability, also increase as the speed of the tip of the foot increases:

- The reaction force from the ball
- The raising moment of the foot

Thus, we try to increase the distance travelled by the ball by using the real-valued GA to search for the poses in a keyframe and the pose times as explained below. The four keyframes of the kicking motion created using IEC, shown in Figure 11, were provided as the initial conditions. Three new keyframes were also added, in which the right foot was in the intermediate position of each pose, so searching was done with a total of seven keyframes. In this experiment, in order to obtain smooth motions, we have interpolated the intervals between keyframes using the natural cubic spline method.

Using real-valued GA, we searched for swinging of the arms in the forward and backward directions, raising of the right foot, and swinging of the upper body in the forward and backward directions. The GTYPE used in our experiment has the eight real elements shown in Table 3 for each keyframe. Because the operation targeted five keyframes, excluding the first and last keyframes, and the time of the last keyframe, a total of 41 real-value parameters were optimized.

For the position of the ankle link, global coordinates (x, y, z) were used as the GTYPE elements. Because of the usage of global coordinates, the positions specified for the ankle link are sometimes unlikely to be feasible, but in those cases a lethal gene results.

For each individual, a dynamics simulation was carried out using OpenHRP, and the results were used to evaluate the robot. The fitness value was provided using the following equation:

$$fitness = \exp(-V_{\max}) + (1 - \exp(-r)), \tag{3}$$

where the maximum value for the chest-link velocity ($V_{\max}$) and the ball travel distance (r) were used as the evaluation. $V_{\max}$ is the penalty in relation to the instability of the motion, and should be as small as possible. The fitness value for any robot that fell down was set to be 0.

Element name	Degrees of freedom
Time	1
Waist-link attitude	2
Arm joint angle	2
Ankle-link position	3

Table 4. Elements of GTYPE: kicking

Fig. 17 shows the fitness transition for the best individual with generations. A significant increase is observed in the distance travelled by the ball from the 10th to the 20th generation, with an accompanying increase in the fitness value. The stability decreased soon after that. Then, the stability subsequently recovered.

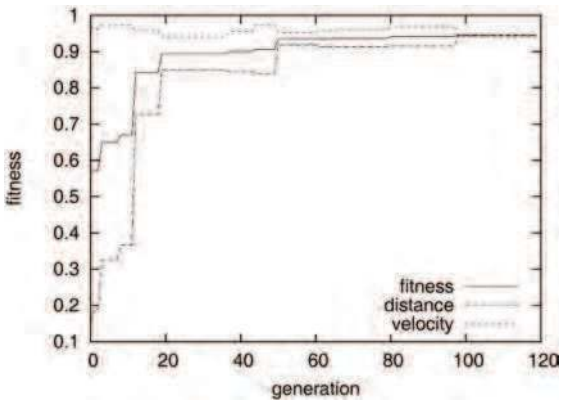


Figure 17. Fitness transition for the kicking motion

Fig. 18 shows the best individual at the initial generation (ball travel distance = 0.17 [m]). Fig. 19 shows the best individual after 120 generations (ball travel distance = 2.91 [m]). The numbers of Fig. 18 and Fig. 19 show the times (sec.) of keyframes.

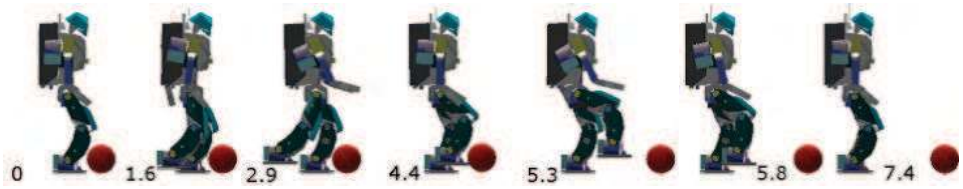


Figure 18. Best individual at the initial generation

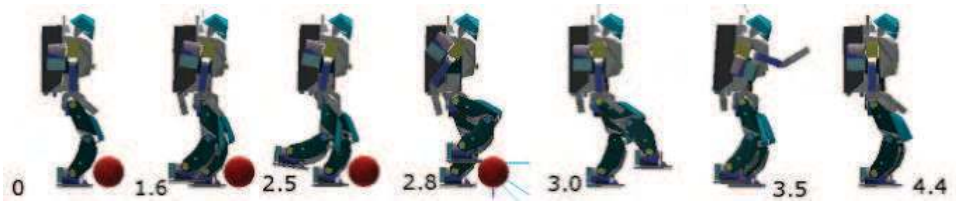


Figure 19. Best individual after 120 generations

The initial generation largely comprised motions without much variety that were performed at the same overall pace. The 120th generation, on the other hand, included varied motions that had the best individual slowly raising its foot upward from 0 seconds to 2.5 seconds and then slowly returning its foot to the floor from 3.0 seconds to 4.4 seconds, followed by rapid motions such as swinging its foot from 2.5 to 3.0 seconds. Because of this, the travel distance expanded from 0.19 [m] to 2.91 [m], or by approximately 17 times.

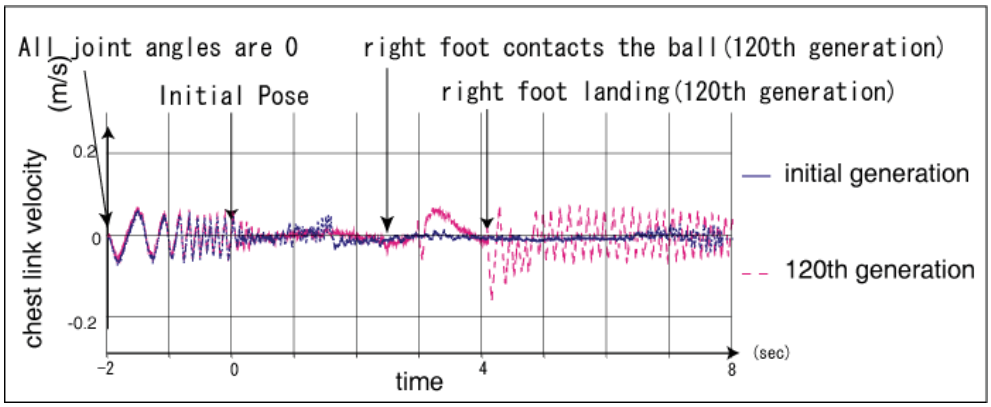


Figure 20. Variation of chest-link velocity

Figure 20 shows the chest-link velocity of x-axis. After the 120th generation, the robot learned to swing his leg so fast that the chest-link velocity was increased after the contact with the ball. Especially, the increase became large when the robot landed his foot on the floor. This is considered as a side effect of the fast swing. However, the robot never fell down because of the obtained motion that cancelled the moment by having the right hand swing down as the right foot swung forward, which is something that humans do(from 1.2 to 3.5 seconds in Fig. 19).

5. Discussion

5.1 Optimization using Multi-objective GA

In Section4, GA was successfully applied to optimizing the humanoid motions. However, the task is essentially multi-objective optimization. For instance, in case of kicking, we have to consider the trade-off between the stability of the robot and the ball distance. Therefore, we have apply Multi-objective GA (MOGA) (Fonseca & Fleming, 1993) with two fitness, i.e., the chest-link velocity ($V_{\max}$) and the ball travel distance (r). We empirically derive the velocity when the robot falls down and set the value as the velocity limitation of the chest link (V_{limit}). Rank-based fitness assignment method and niche-formation method are employed to determine the fitness for maximizing ($V_{\text{limit}} - V_{\max}$) and r. Roulette selection is also used according to the fitness values.

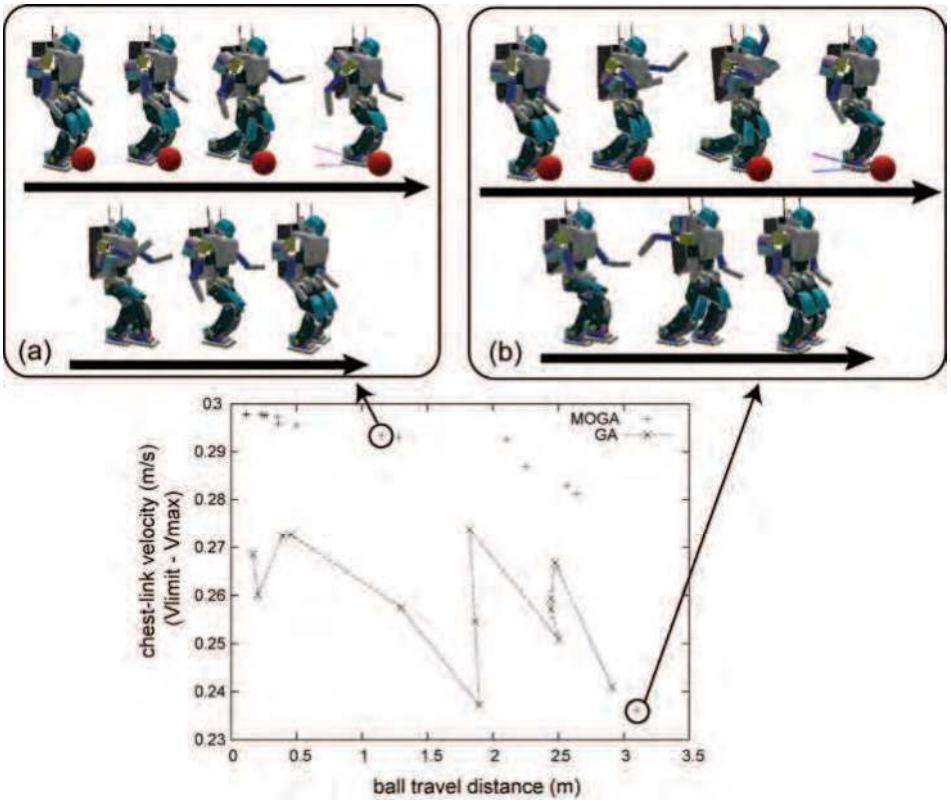


Figure 21. Fitness values of best individuals for Simple GA and Pareto-optimal individuals for MOGA

Fig. 21 shows the performance difference between simple GA and MOGA, in which 30 individuals were evolved for 50 generations. Best individuals at every generation are shown as the "×" dots for simple GA, whereas Pareto-optimal individuals are plotted as "＋" dots

for MOGA. As can be seen from the figure, the chest-link velocity is smaller for MOGA when the robot tries to kick a ball at the same travel distance. This shows that MOGA, if applied effectively, is more suitable for designing humanoid motions. We will work on this topic, i.e., the extension of MOGA to a more complex task, for future research.

5.2 Future research

Currently, because the stability is not compensated using the controller of the actual robots, the motion range is restricted by the instability of the motion caused by the differences between the individuals. A future issue will be to compensate the stability using the actual robots, and thus to expand the range of motions that are feasible for the actual robots. In optimizing the kicks, the torque was not limited in the present experiment, so the results could not be applied to the actual robots without modification. In the future, our goal will be to apply these results to the actual robots, taking elements such as torque limits and angular velocity limits into consideration.

6. Conclusion

Through IEC, we proposed a motion design method for humanoid robots which does not need any specialized robotics knowledge, such as kinematics or dynamics. As an example, the designed dance motions were confirmed using the actual robots, and kicking motions were confirmed using a dynamics simulator. At the same time, however, in order to realize motions that deviate from the presuppositions of the physical model, the motions used with IEC were evolved using a GA.

In case of designing a sitting motion, it was possible to find the stable behaviour by searching for a space of joint angle trajectories. A kicking motion was set as a task, and the distance travelled by the ball was improved on a dynamics simulator. In addition, since the task is a multi-objective optimization, we empirically found MOGA successfully applicable to designing the motion.

Also, the ball travel distance and the stability, which we used as standards for evaluation, have a trade-off relationship with each other. In order to determine the optimum solution based on these two competing target functions, we would like to carry out optimization using a somewhat larger real-valued GA. Our future works include the application to more complicated tasks in consideration of physical constraints.

7. References

Fonseca, C.M. & Fleming, P.J. (1993). Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp 416-423, Morgan Kaufmann

Inoue, Y.; Tohge, T. & Iba, H. (2007). Cooperative transportation system for humanoid robots using simulation-based learning, *Applied Soft Comput*ing, Vol. 7, No.1, pp 115-125

Inoue, Y.; Tohge, T. & Iba, H. (2004). Learning to acquire autonomous behavior - cooperation by humanoid robots, *Proceedings of the Genetic and Evolutionary Computation Conference*, pp 1394-1399

Kanehiro, F.; Hirukawa, H. & Kajita, S. (2004). Open architecture humanoid robotics platform, *Journal of Robotics Research*, Vol.23, No.2, pp 155-165, October

Kuroki, Y.; Blank, B.; Mikami, T.; Mayeux, P.; Miyamoto, A.; Playter, R.; Nagasaka, K.; Raibert, M.; Nagano, M. & Yamaguchi, J. (2003). Motion creating system for a small biped entertainment robot. *Proceedings of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, pp 1394-1399

Nakaoka, S.; Nakazawa, A.; Yokoi, K. & Ikeuchi, K. (2003). Leg motion primitives for a humanoid robot to imitate human dances. *Proceedings of Sixth International Conference on Humans and Computers(HS2003)*, August

Nishiwaki, K.; Kagami, S.; Kuniyoshi, Y.; Inaba, M. & Inoue, H. (2002). Online generation of humanoid walking motion based on a fast generation method of motion pattern that follows desired ZMP, *Proceedings of the 2002 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, EPFL, Lausanne, Switzerland, pp 2684-2689

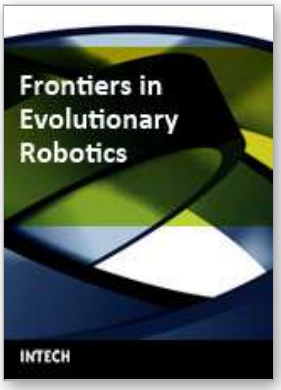
Takagi, H. (1998), Interactive evolutionary computation - cooperation of computational intelligence and human kansei, *Proceedings of 5th International Conference on Soft Computing and Information/Intelligent Systems*, pp 41-50, October

Vukobratovic, M. & Stepanenko, J. (1972). On the stability of anthropomorphic systems, *Mathematical Biosciences*, Vol. 15, pp 1-37

Wakaki, H. & Iba, H. (2002). Motion design of a 3D-CG avatar using interactive evolutionary computation, *Proceedings of 2002 IEEE international Conference on Systems, man and Cybernetics (SMC'02)*. IEEE Press

Wolff, K. & Nordin, P. (2003). Learning biped locomotion from first principles on a simulated humanoid robot using linear genetic programming. *Proceedings of the Genetic and Evolutionary Computation Conference*, pp 495-506

Yamane, K. & Nakamura, Y. (2002). Synergetic cg choreography through constraining and deconstraining at will, *Proceedings of International Conference on Robotics and Automation*, pp 855-862, May



Frontiers in Evolutionary Robotics

Edited by Hitoshi Iba

ISBN 978-3-902613-19-6
Hard cover, 596 pages
Publisher I-Tech Education and Publishing
Published online 01, April, 2008
Published in print edition April, 2008

This book presented techniques and experimental results which have been pursued for the purpose of evolutionary robotics. Evolutionary robotics is a new method for the automatic creation of autonomous robots. When executing tasks by autonomous robots, we can make the robot learn what to do so as to complete the task from interactions with its environment, but not manually pre-program for all situations. Many researchers have been studying the techniques for evolutionary robotics by using Evolutionary Computation (EC), such as Genetic Algorithms (GA) or Genetic Programming (GP). Their goal is to clarify the applicability of the evolutionary approach to the real-robot learning, especially, in view of the adaptive robot behavior as well as the robustness to noisy and dynamic environments. For this purpose, authors in this book explain a variety of real robots in different fields. For instance, in a multi-robot system, several robots simultaneously work to achieve a common goal via interaction; their behaviors can only emerge as a result of evolution and interaction. How to learn such behaviors is a central issue of Distributed Artificial Intelligence (DAI), which has recently attracted much attention. This book addresses the issue in the context of a multi-robot system, in which multiple robots are evolved using EC to solve a cooperative task. Since directly using EC to generate a program of complex behaviors is often very difficult, a number of extensions to basic EC are proposed in this book so as to solve these control problems of the robot.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Toshihiko Yanase and Hitoshi Iba (2008). Evolutionary Motion Design for Humanoid Robots, Frontiers in Evolutionary Robotics, Hitoshi Iba (Ed.), ISBN: 978-3-902613-19-6, InTech, Available from: http://www.intechopen.com/books/frontiers_in_evolutionary_robotics/evolutionary_motion_design_for_humanoid_robots

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821