Machine Learning Algorithms in Bipedal Robot Control

Shouyi Wang, Wanpracha Chaovalitwongse and Robert Babuška

Abstract—**In the past decades, machine learning techniques, such as supervised learning, reinforcement learning, and unsupervised learning, have been increasingly used in the control engineering community. Various learning algorithms have been developed to achieve autonomous operation and intelligent decision making for many complex and challenging control problems. One of such problems is bipedal walking robot control. Although still in their early stages, learning techniques have demonstrated promising potential to build adaptive control systems for bipedal robots. This paper gives a review of recent advances on the state-of-the-art learning algorithms and their applications to bipedal robot control. The effects and limitations of different learning techniques are discussed through a representative selection of examples from the literature. Guidelines for future research on learning control of bipedal robots are provided in the end.**

Index Terms—**Bipedal walking robots, learning control, supervised learning, reinforcement learning, unsupervised learning**

I. Introduction

Bipedal robot control is one of the most challenging and popular research topics in the field of robotics. We have witnessed an escalating development of bipedal walking robots based on various types of control mechanisms. However, unlike the well-solved classical control problems (e.g., control of industrial robot arms), the control problem of bipedal robots is still far from being fully solved. Although many classical model-based control techniques have been proposed to bipedal robot control, such as trajectory tracking control [76], robust control [105], and model predictive control [57], these control laws are generally pre-computed and inflexible. The resulting bipedal robots are usually not satisfactory in terms of stability, adaptability and robustness. There are five exceptional characteristics of bipedal robots that present challenges and constrains to the design of control systems:

- Non-linear dynamics: Bipedal robots are highly non-linear and naturally unstable systems. The well-developed classical control theories for linear systems cannot be applied directly.
- Discretely changing in dynamics: Each walking cycle consists of two different situations in a sequence: the statically stable double-support phase (both feet in contact with the ground) and the statically unstable single-support phase (only one foot contacts with the ground). Suitable control strategies are required for step-to-step transitions.
- Under-actuated system: Walking robots are unconnected to the ground. Even if all joints of a bipedal robot are controlled perfectly, it is still not enough to completely control all the degrees of freedom (DOFs) of the robot.
- Multi-variable system: Walking systems usually have many DOFs, especially in 3-dimensional spaces. The interactions between DOFs and the coordination of multi-joint movements have been recognized as a very difficult control problem.
- Changing environments: Bipedal robots have to be adaptive to uncertainties and respond to environmental changes correctly. For example, the ground may become uneven, elastic, sticky, soft or stiff; there may be obstacles on the ground. A bipedal robot has to adjust its control strategies fast enough to such environmental changes.

In recent years, the great advances in computing power have enabled the implementation of highly sophisticated learning algorithms in practice. Learning algorithms are among the most valuable tools to solve complex problems that need 'intelligent decision making', and to design truly intelligent machines with human-like capabilities. Robot learning is a rapidly growing area of research at the intersection of robotics and machine learning [22]. With a classical control approach, a robot is explicitly programmed to perform the desired task using a complete mathematical model of the robot and its environment. The parameters of the control algorithms are often chosen by hand after extensive empirical testing experiments. On the other hand, in a learning control approach, a robot is only provided with a partial model, and a machine learning algorithm is employed to fine-tune the parameters of the control system to acquire the desired skills. A learning controller is capable of improving its control policy autonomously over time, in some sense tending towards an ultimate goal. Learning control techniques have shown great potential of adaptability and flexibility, and thus become extremely active in recent years. There have been a number of successful applications of learning algorithms on bipedal robots [11], [25], [51], [82], [104], [123]. Learning control techniques appear to be promising in making bipedal robots reliable, adaptive and versatile. In fact, building intelligent humanoid walking robots has been one of the main research streams in machine learning. If such robots are ever to become a reality, learning control techniques will definitely play an important role.

Shouyi Wang is with the Department of Industrial and Manufacturing Systems Engineering, University of Texas at Arlington, Arlington, TX 76019, USA.
E-mail: shouyiw@uta.edu
Wanpracha Chaovalitwongse is with the department of Industrial and Systems Engineering and the department of Radiology, University of Washington, Seattle, WA 98104, USA.
E-mail: artchao@uw.edu
Robert Babuška is with the Delft Center for Systems and Control, Delft University of Technology, Delft, 2628CD Netherlands.
E-mail: r.babuska@tudelft.nl

There are several comprehensive reviews of bipedal walking robots [16], [50], [109]. However, none of them has been specifically dedicated to providing the review of the state-of-the-art learning techniques in the area of bipedal robot control. This paper aims at bridging this gap. The main objectives of this paper are two-fold. The first goal is to review the recent advances of mainstream learning algorithms. And the second objective is to investigate how learning techniques can be applied to bipedal walking control through the most representative examples.

The rest of the paper is organized as follows. Section II, presents an overview of the three major types of learning paradigms, and surveys the recent advances of the most influential learning algorithms. Section III provides an overview of the background of bipedal robot control, including stability criteria, classical model-based and biological-inspired control approaches. Section IV presents the state-of-the-art learning control techniques that have been applied to bipedal robots. Section V gives a technical comparison of learning algorithms by their advantages and disadvantages. Finally, we identify some important open issues and promising directions for future research.

II. Learning Algorithms

Learning algorithms specify how the changes in a learner's behavior depend on the inputs it received and on the feedback from the environment. Given the same input, a learning agent may respond differently later on than it did earlier on. With respect to the sort of feedback that a learner has access to, learning algorithms generally fall into three broad categories: supervised learning (SL), reinforcement learning (RL) and unsupervised learning (UL). The basic structures of the three learning paradigms are illustrated in Figure 1.

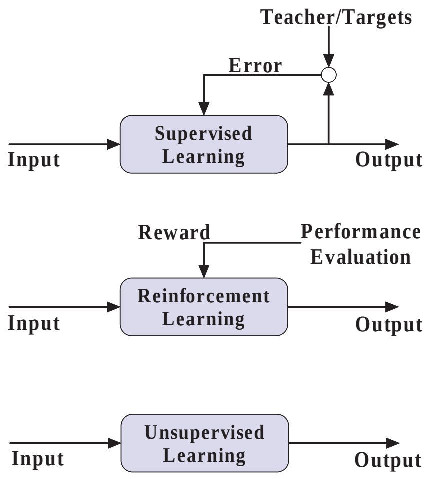

Fig. 1. Basic structures of the three learning paradigms: supervised learning, reinforcement learning and unsupervised learning.

A. Supervised Learning (SL)

SL is a machine learning mechanism that first finds a mapping between inputs and outputs based on a training data set, and then makes predictions to the inputs that it has never seen in training. To achieve good performance of generalization, the training data set should contain a fully representative collection of data so that a valid general mapping between inputs and outputs can be found. SL is one of the most frequently used learning mechanisms in designing learning systems. A large number of SL algorithms have been developed in the past decades. They can be categorized into several major groups as discussed in the following.

1) Neural Networks (NNs): NNs are powerful tools that have been widely used to solve many SL tasks where there exists sufficient amount of training data. There are several popular learning algorithms for training NNs (such as Perceptron learning rule, Widrow-Hoff rule), but the most well-known and commonly used one is backpropagation (BP) developed by Rumelhart in the 1980's [88]. BP adjusts the weights of a NN by calculating how the error changes as each weight is increased or decreased slightly. The basic update rule of BP is given by:

$$\omega_j = \omega_j - \alpha \frac{\partial E}{\partial \omega_j}, \tag{1}$$

where α is the learning rate that controls the size of weight changes at each iteration, $\frac{\partial E}{\partial \omega_j}$ is the partial derivative of the error function E with respect to weight ω_j. BP-based NNs have become popular in practice since they can often find a good set of weights in a reasonable amount of time. They can be used to solve many problems involve large amounts of data and complex mapping relationships. As a gradient-based method, BP is subject to the local minima problem, which is inefficient in searching of global optimal solutions. One of the approaches to tackle this problem is to try different initial weights until a satisfactory solution is found [119].

In general, the major advantage of NN-based SL methods is that they are convenient to use and one does not have to understand the solution in great detail. For example, one does not need to know anything about a robot's model; a NN can be trained to estimate the robot's model from the input-output data of the robot. However, the drawback is that the learned NN is usually difficult to interpret due to its complicated structure.

2) Locally Weighted Learning (LWL): Instead of mapping nonlinear functions globally (such as BP), LWL represents another class of methods which fit complex nonlinear functions by local kernel functions. A demonstration of LWL is shown in Figure 2. There are two major types of LWL: memory-based LWL which simply stores all training data in memory and uses efficient interpolation techniques to make predictions of new inputs [1]; non-memory-based LWL which constructs compact representations of training data by recursive techniques so as to avoid storing large amounts of data in memory [62], [107]. The key part of all LWL algorithms is to determine the region of validity in which a local model can be trusted. Suppose there are K local models, the region of validity can be calculated from a Gaussian kernel by:

$$\omega_k = \exp(-\frac{1}{2}(x-c_k)^T D_k (x-c_k)), \tag{2}$$

where c_k is the center of the k-th linear model, and D_k is the distance metric that determines the size and shape of the validity region of the kth linear model. Given a query point x, every linear model calculates a prediction $\hat{y}_k(x)$ based on the obtained local validity. Then the output of LWL is the normalized weighted mean of all K linear models calculated by:

$$\hat{y} = \frac{\sum_{k=1}^{K} \omega_k \hat{y}_k}{\sum_{k=1}^{K} \omega_k} . \tag{3}$$

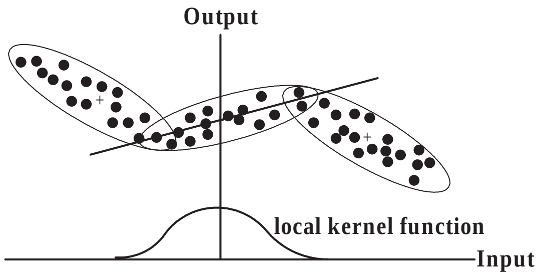

Fig. 2. A schematic view of locally weighted regression.

LWL achieves low computational complexity and efficient learning in high dimensional spaces. Another attractive feature of LWL is that local models can be allocated as needed, and the modeling process can be easily controlled by adjusting the parameters of the local models. LWL techniques have been used quite successfully to learn inverse dynamics or kinematic mappings in robot control systems [6], [7]. One of the most popular LWL algorithms is called locally weighted projection regression (LWPR), which has shown good capability to solve several on-line learning problems of humanoid robot control in [108].

3) Support Vector Machine (SVM): SVM is a widely used classification technique in machine learning [20]. It has been used in pattern recognition and classification problems, such as handwritten recognition [96], speaker identification [95], face detection in images [74], and text categorization [42]. The most important idea of SVM is that every data instance can be classified by a hyperplane if the data set is transformed into a space with sufficiently high dimensions [14]. Therefore, a SVM first projects input data instances into a higher dimensional space, and then divides the space with a separation hyperplane which not only minimizes the misclassification error but also maximizes the margin separating the two classes. One of the most successful optimization formalism of SVM is based on robust linear programming. Consider two data groups in the n-dimensional real-space R^n, optimization formalism is given by:

$$\min_{\omega,\gamma,y,z} \frac{ey}{m} + \frac{ez}{k} \tag{4}$$

$$s.t.\ A\omega - e\gamma - e \geq y \tag{5}$$

$$-B\omega + e\gamma - e \geq z \tag{6}$$

$$y \geq 0, z \geq 0, \tag{7}$$

where A is a $m \times n$ matrix representing m observations in group one, and B is a $k \times n$ matrix representing k observations in group two. The two data groups are separated by a hyperplane (defined by $A\omega \geq e\gamma$, $B\omega \leq e\gamma$), and y and z are binary $\{0, 1\}$ decision variables indicating if a data instance in group A or B violates the hyperplane constraint. The objective function is, therefore, minimizing the average misclassifications subject to the hyperplane constraint for separating data instances of A from data instances of B. The training of a SVM obtains a global solution instead of local optimum. However, one drawback of SVM is that the results are sensitive to the choices of the kernel function. The problem of choosing appropriate kernel functions is still left to user's creativity and experience.

4) Decision Tree: Decision trees use a hierarchical tree model to classify or predict data instances. Given a set of training data with associated attributes, a decision tree can be induced by using algorithms such as ID3 [83], CART [13], and C4.5 [84]. While ID3 and C4.5 are primarily suitable for classification tasks, CART has been specifically developed for regression problems. The most well-know algorithm is C4.5 [84] which builds decision trees by using the concept of Shannon entropy [98]. Based on the assumption that each attribute of data instances can be used to make a decision, C4.5 examines the relative entropy for each attribute and accordingly splits the data set into smaller subsets. The attribute with the highest normalized information gain is used to make decisions. Ruggieri [87] provided an efficient version of C4.5, called EC4.5, which is claimed to be able to achieve a performance gain up to five times while compute the same decision trees as C4.5. Olcay and Onur [120] present three parallel C4.5 algorithms which are designed to be applicable to large data sets. Baik and Bala [9] present a distributed version of decision tree, which generates partial trees and communicates the temporary results among them in a collaborative way. The distributed decision trees are efficient for large data sets collected in a distributed system.

One of the most useful characteristics of decision trees is that they are simple to understand and easy to interpret. People can understand decision tree models after a brief explanation. It should be noticed that a common assumption made in decision trees is that data instances belonging to different classes have different values in at least one of their attributes. Therefore, decision trees tend to perform better when dealing with discrete or categorical attributes, and will encounter problems when dealing with continuous data. Moreover, another limitation of decision trees is that they are usually sensitive to noise.

B. Reinforcement Learning

Among other modes of learning, humans heavily rely on learning from interaction, repeating experiments with small variations and then finding out what works and what does not. Consider a child learning to walk – it tries out various movements, some actions work and are rewarded (moving forward), while others fail and are punished (falling). Inspired by animal and human learning, the Reinforcement Learning (RL) approach enables an agent to learn a mapping from states to actions by trial and error so that the expected cumulative reward in the future is maximized.

1) General RL Scheme: RL is capable of learning while gaining experience through interactions with environments.

It provides both qualitative and quantitative frameworks for understanding and modeling adaptive decision-making problems in the form of rewards and punishments. There are three fundamental elements in a typical RL scheme:

- state set S, in which a state $s \in S$ describes a system's current situation in its environment,
- action set A, from which an action $a \in A$ is chosen at the current state s,
- scalar reward $r \in \mathbb{R}$ indicates how well the agent is currently doing with respect to the given the task.

At each discrete time step t, a RL agent receives its state information $s_t \in S$, and takes an action $a_t \in A$ to interact with its environment. The action a_t changes its environment state from s_t to s_{t+1} and this change is communicated to the learning agent through a scalar reward r_{t+1}. Usually, the sign of reward indicates whether the chosen action a_t was good (positive reward) or bad (negative reward). The RL agent attempts to learn a policy that maps state s_t to action a_t so that the sum of the expected future reward R_t is maximized. The sum of future rewards is usually formulated in a discounted way [102], which is given by:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+2} + ... = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}, \qquad (8)$$

where γ is called the discounting rate that satisfies $0 < \gamma < 1$. Applications of RL have been reported in areas such as robotics, manufacturing, computer game playing, and economy [60]. Recently, RL has also been used in psychology and cognitive models to simulate human learning in problem-solving and skill acquisition [31].

2) Two Basic RL Structures: Many RL algorithms are available in the literature. The key element of most of them is to approximate the expected future rewards for each state or each state-action pair (under the current policy). There are two prevalent RL structures: *actor-critic* scheme [56] and *Q-learning* scheme [114] algorithms.

- An actor-critic algorithm has two separate function approximators for action policy and state values, respectively. The learned policy function is known as actor, because it is used to select actions. The estimated value function is known as critic since it evaluates the actions made by the actor. The value function and policy function are usually both updated by temporal difference error.
- Q-learning algorithms learn a state-action value function, known as Q-function, which is often represented by a lookup table indexed by state-action pairs. Since Q-table is constructed on state-action space rather than just state space, it discriminates the effects of choosing different actions in each state. Compared to actor-critic algorithms, Q-learning is easier to understand and implement.

The basic structure of actor-critic learning and Q-learning algorithms are shown in Figure 4 and Figure 3, respectively.

3) Recent Advances in RL: Most RL algorithms suffer from the curse of dimensionality as the number of parameters to be learned grows exponentially with the size of the state space. Thus most of the RL methods are not applicable to high-dimensional systems. One of the open questions in RL is how to scale up RL algorithms to high-dimensional state-action spaces. Recently, policy-gradient methods have attracted great attention in RL research since they are considered to be applicable to high-dimensional systems. The policy-gradient RL have been applied to some complex systems with many degree of freedoms (DOFs), such as robot walking [25], [55], [70], [104], and traffic control [86]. Peters et al. [77] made a comprehensive survey of policy-gradient-based RL methods, and developed a class of RL algorithms called natural actor-critic learning, for which the action policy was updated based on natural policy gradients [48]. The efficiency of the proposed learning algorithms was demonstrated by a 7-DOF real robot arm which was programmed to learn hit a baseball. The natural actor-critic algorithm is currently considered the best choice among the policy-gradient methods [78]. In recent years, hierarchical RL approaches have also been developed to handle the curse of dimensionality [61]. Multi-agent or distributed RL are also an emerging topic in current research of RL [33]. Some researchers also use predictive state representation to improve the generalization of RL [85].

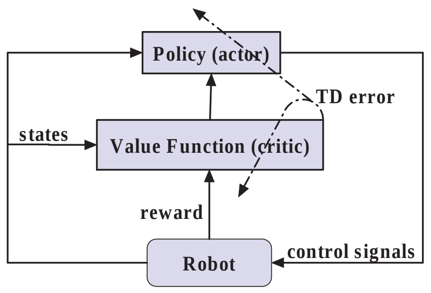

Fig. 3. The actor-critic learning architecture for robot control.

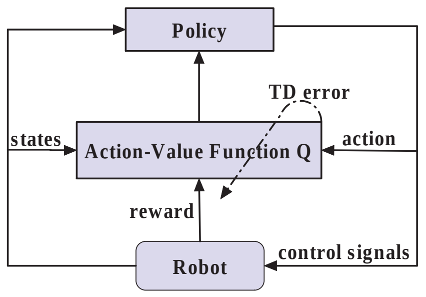

Fig. 4. The Q-learning architecture for robot control.

C. Unsupervised Learning

UL is inspired by the brain's ability to extract patterns and recognize complex visual scenes, sounds, and odors from sensory data. It has roots in neuroscience/psychology and is based on information theory and statistics. An unsupervised learner receives no feedback from its environment at all. It only responds to the received inputs. At first glance this seems impractical, since how can we train a learner if we do not know what it is supposed to do. Actually most of these algorithms perform some kind of clustering or associative rule learning.

1) Clustering: Clustering is the most important form of UL. It deals with data that have not been pre-classified in any way, and does not need any type of supervision during its learning process. Clustering is a learning paradigm that automatically partitions input data into meaningful clusters based on the degree of similarity.

The most well-known clustering algorithm is k-means clustering, which finds k cluster centers that minimize a squared-error criterion function [23]. Cluster centers are represented by the gravity center of data instances; that is, the cluster centers are arithmetic means of all data samples in the cluster. k-means clustering assigns each data instance to a cluster whose center is nearest to it. Since k-means clustering generates partitions such that each pattern belongs to one and only one cluster, the obtained clusters are disjoint. Fuzzy c-means (FCM) was developed to allow one data instance to belong to two or more clusters rather than just being assigned completely to one cluster [24]. Each data instance is associated with each cluster by a membership function, which indicates the degree of membership to that cluster. The FCM algorithm finds the weighted mean of each cluster and then it assigns a membership degree to each data sample in the cluster. For example, data samples on the edge of a cluster belong to the cluster to a lower degree than the data around the center of the cluster.

Recently, distributed clustering algorithms have attracted considerable attention to extract knowledge from large data sets [41], [4]. Instead of being transmitted to a central site, data can be firstly clustered independently at different local sites. Then in the subsequent step, the central site establishes a global clustering based on the local clustering results.

2) Hebbian Learning: The key idea of Hebbian learning [37] is that neurons with correlated activity increase their synaptic connection strength. It is used in artificial neural networks to learn associations between patterns that frequently occur together. The original Hebb's hypothesis does not explicitly address the update mechanism for synaptic weights. A generalized version of Hebbian learning, called differential Hebbian rule [58], [54] can be used to update the synaptic weights. The basic update rule of differential Hebbian learning is given by

$$w_{ij}^{new} = w_{ij}^{old} + \eta \Delta x_i \Delta y_j, \tag{9}$$

where w_{ij} is the synaptic strength from neuron i to neuron j, Δx_i and Δy_j denote the temporal changes of presynaptic and postsynaptic activities, and η is the learning rate to control how fast the weights get modified in each step. Notably, differential Hebbian learning can be used to model simple level of adaptive control that is analogous to self-organizing cortical function in humans. It can be applied to construct an unsupervised, self-organized learning control system for a robot to interact with its environment without any evaluative information. Although it seems to be a low level of learning, Porr and Wörgötter [80] showed that this autonomous mechanism can develop rather complex behavioral patterns in closed loop feedback systems. They confirmed this idea on a real bipedal robot, which was capable of walking stably using the unsupervised differential Hebbian learning [32].

III. Background of Bipedal Walking Control

According to a US army report, more than 50 percent of the earth surface is inaccessible to traditional vehicles with wheels or tracks [5], [10]. However, we have to transport over rough terrains in many real-world tasks, such as emergency rescue in isolated areas with unpaved roads, relief after a natural disaster, and alternatives for human labor in dangerous working environments. To date, the devices available to assist people in such tasks are still very limited. As promising tools to solve these problems, bipedal robots have become one of the most exciting and emerging topics in the field of robotics. Moreover, bipedal robots can also be used to develop new types of rehabilitation tools for disabled people and to help elderly with household work. The important prospective applications of bipedal walking robots are shown in Figure 5.

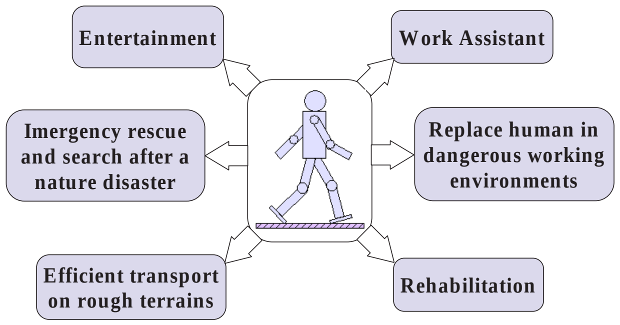
Entertainment
Work Assistant
Imergency rescue and search after a nature disaster
Replace human in dangerous working environments
Efficient transport on rough terrains
Rehabilitation

Fig. 5. Prospective applications of bipedal walking robots.

A. Stability Criteria in Bipedal Robot Control

Bipedal robot walking can be broadly characterized as static walking, quasi-dynamic walking and dynamic walking. Different types of walking are generated by different walking stability criteria as follows.

- Static Stability: the position of Center of Mass (COM) and Center of Pressure (COP) are often used as stability criteria for static walking. A robot is considered stable if its COM or COP is within the convex hull of the foot support area. Static stability is the oldest and the most constrained stability criterion, often used in early days of bipedal robots. A typical static walking robot is SD-2 built by Salatian et al. [89].
- Quasi-Dynamic Stability: the most well known criterion for quasi-dynamic walking is based the concept of Zero Moment Point (ZMP) introduced by M. Vukobratović et al. in [111]. ZMP is a point on the ground where the resultant of the ground reaction force acts. A stable gait can be achieved by making the ZMP of a bipedal robot stay within the convex hull of the foot support area during walking. ZMP is frequently used as a guideline in designing reference walking trajectories for many bipedal robots. An illustration of the ZMP criterion is shown in Figure 6. Recently, Sardain and Bessonnet [92] proposed a virtual COP-ZMP, which extended the concept of ZMP to stability on uneven terrains. Another criterion for quasi-dynamic walking is the foot rotation point (FRI),

which is a point on the ground where the net ground reaction force acts to keep the foot stationary [36]. This walking stability requires keeping the FRI point within the convex hull of the foot support area. One advantage of FRI point is that it is capable of indicating the severity of instability. The longer the distance between FRI and the foot support boundary, the greater the degree of instability.

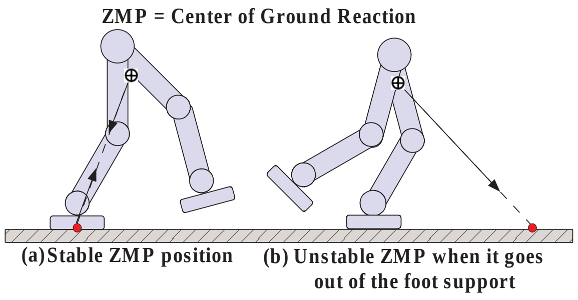
ZMP = Center of Ground Reaction
(a)Stable ZMP position
(b) Unstable ZMP when it goes out of the foot support

Fig. 6. An illustration of the Zero Moment Point (ZMP) stability criterion.

- Dynamic Stability: the stability of dynamic walking is a relatively new stability paradigm. The most well-known criterion was introduced by McGeer [67], who proposed the concept of 'passive dynamic walking' (PDW) in 1990. The stability of a bipedal robot depends solely on its dynamic balance. As a result, this stability criterion has the fewest artificial constraints, and thus has more freedom to yield efficient, fast and natural-looking gaits. A number of dynamic bipedal walking robots have been built since the 90's. A simplified example of PDW is shown in Figure 7.

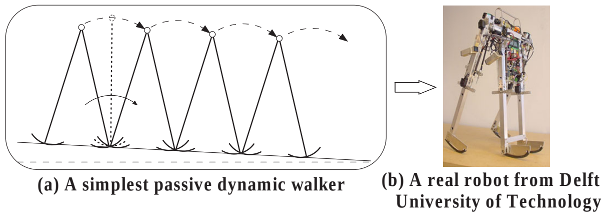
(a) A simplest passive dynamic walker
(b) A real robot from Delft University of Technology

Fig. 7. A demonstration of the simplest passive dynamic walker as well as a real PDW robot prototype from Delft University [116].

Table I compares the walking speeds of some typical bipedal robots using different stability criteria. In general, the static stability is straightforward to ensure stable gaits, but the resulting gaits are usually very slow and energy inefficient. Quasi-dynamic stability is less restrictive than static stability, because the COP or COM of a bipedal robot is allowed to be outside of the support polygon of the feet. However, the resulting gait is still restricted in terms of efficiency and speed. Dynamic stability has the fewest restrictions, allowing more freedom in generating fast and natural walking patterns [19].

B. Control Techniques for Bipedal Robots

Various control approaches have been developed for bipedal robot locomotion. Two main streams can be distinguished: dynamic model-based methods and biologically-inspired methods. This categorization is further detailed in Figure 8.

TABLE I
WALKING SPEED OF BIPEDAL ROBOTS USING DIFFERENT STABILITY CRITERIA (THE RELATIVE SPEED = WALKING SPEED/LEG LENGTH)

Stability Criterion	Relevant Robots	Walking speed (m/s)	Leg length (m)	Relative speed
Static	SD-2 [89]	0.12	0.51	0.24
Quasi -Dynamic	HRP-2L [46]	0.31	0.69	0.44
	QRIO [26]	0.23	0.30	0.77
	ASIMO [38]	0.44	0.61	0.72
Dynamic	Meta [97]	0.58	0.60	0.97
	Rabbit [17]	1.2	0.8	1.50
	RunBot [32]	0.8	0.23	3.48

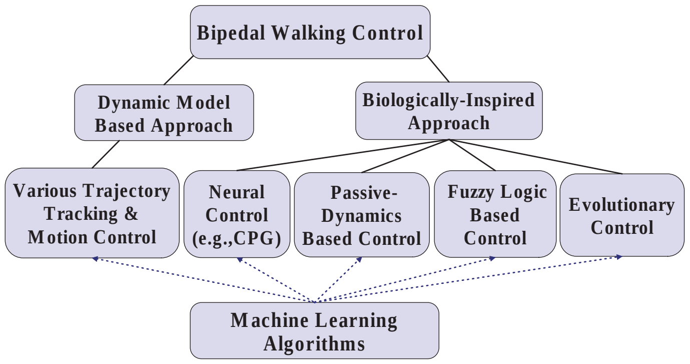
Bipedal Walking Control
Dynamic Model Based Approach
Biologically-Inspired Approach
Various Trajectory Tracking & Motion Control
Neural Control (e.g.,CPG)
Passive-Dynamics Based Control
Fuzzy Logic Based Control
Evolutionary Control
Machine Learning Algorithms

Fig. 8. Categorization of bipedal walking control approaches. Machine learning algorithms have been applied in each group of approaches to enhance their control performance in terms of adaptability, robustness and scalability.

1) Model-Based Control Approaches: With this approach, the kinematics and the dynamics of a bipedal robot as well as its environments are assumed to be precisely modeled. Trajectory-tracking methods have been intensively studied, based on traditional control theory. Trajectories of joint angles or torques are obtained either from real world human walking or by using walking pattern generators. Most controllers of this type use the ZMP stability criterion. The reference trajectory of a robot is defined such that the resulting ZMP motion is stable at all times.

Park and Chung [76] applied an adaptive trajectory tracking controller to a 6-DOF bipedal robot using online ZMP information. However, the adaptation only allowed small changes in the prescribed trajectory. To deal with larger disturbances, Denk and Schmidt [21] proposed a method to use a set of trajectories. Their bipedal robot was able to choose different trajectories for different situations. However, the drawback of this method is that in order to deal with many possible situations, it needs a large set of trajectories and switching between the trajectories may cause unexpected effects in real-time experiments. An improved method was presented by Chevallereau and Sardain [17], where a continuous set of parameterized trajectories was used to avoid the switching problem. However, it is still very costly to design appropriate trajectories for each joint of a bipedal robot.

Robust control theory has also been applied to bipedal walking robots. Tzafestas et al. [105] applied a sliding-mode control to a 9-link bipedal robot. The sliding-mode controller ensured the joint trajectories to move towards a sliding surface and reach it from any initial condition within a finite time horizon. Since the control law involved a switching function, the designed walking robot suffered from the undesirable effects of control signal chattering.

Model predictive control (MPC) for bipedal walking was investigated by Kooij et al. [57] and Azevedo et al. [8]. Based on MPC, the walking control problem reduces to a quadratic optimization problem. The physical limitations, the geometry of environments and the motion specifications are described as a set of mathematical equations and inequalities. By adjusting the parameters of these constrains, a simulated bipedal robot managed to walk on a slope. However, the long optimization time makes this method unsuitable for real-time implementation.

There are also some studies that consider the single-support phase of bipedal walking as an inverted pendulum. As a result, a number of bipedal walking control systems have been built based on the simple inverted pendulum model (IPM) and its variations [46], [47], [99], [103]. Kajita et al. [47] built a 2D bipedal model based on a linear inverted pendulum, and developed an inverted pendulum based control scheme for their bipedal robot to walk on rugged terrains. In a further study, they extended the control scheme to 3D by analyzing the dynamics of a 3D inverted pendulum. Albert and Gerth [2] proposed two models called TMIPM (two masses IPM) and MMIPM (multiple masses IPM) for the path planning of a bipedal robot without a trunk. This method can be considered as an extension of the concept of IPM and achieved higher gait stability compared to other IPM approaches.

2) Biologically-Inspired Approaches: Animals are capable of moving with elegance and in a highly energy-efficient way. There is a considerable amount of literature focusing on biologically inspired control systems for bipedal robots. According to different types of biological aspects studied, the research of biologically-inspired bipedal walking control can be divided into four major groups: PDW-based methods, neural oscillator-based methods, fuzzy control methods, and evolutionary computing-based methods.

A PDW robot [67], inspired by human walking down a slope, exhibits a very efficient and natural dynamic walking pattern. However, passive dynamic walkers lack controllability and have poor robustness. Several researchers expanded McGeer's work to actuate PDW robots while keeping the energy efficiency and natural walking properties of PDW. Goswami et. al [35] presented a control policy to increase the robustness of a two-link PDW walker. Collins [19] actuated a 3D PDW walker by implementing ankle torque to the robot. Wisse [116] built a 3D PDW-based walker which can walk on a level surface through a pneumatic actuator mounted on the hip of the robot. Tedrake [104] actuated a 3D PDW walker and achieved efficient and natural bipedal walking on a flat surface by using a RL controller.

Neural oscillator-based approaches are inspired by central pattern generators (CPGs) which have been identified in the spinal cord of many animals. CPGs are considered to be responsible for generating rhythmic movements that are robust to environment changes [68]. A CPG controller consists of coupled neural oscillators, some of which are excitatory and the others are inhibitory (Figure 9-a). Each pair of coupled oscillators controls one joint of a robot. Through proper coordination between these oscillators, different types of walking patterns can be generated [73]. The most prominent advantage of using CPG is that the control signal produced by CPG is effectively restricted within the space determined by the inherent rhythmic patterns of the oscillators. The search for an optimal policy becomes easier than that without any restrictions.

Fuzzy logic is another popular biologically-inspired paradigm in bipedal robot control. A fuzzy controller usually consists of linguistic if-then rules which capture human knowledge. A number of fuzzy control systems have been developed for bipedal walking robots [51], [118]. Evolutionary computation approaches, such as genetic algorithms (GAs), are inspired by the biological evolution mechanisms of reproduction, crossover, and mutation. GAs have been shown to be effective in exploring optimal solutions in large spaces for many complex control problems [34]. GA-based methods have also been used to obtain optimal control solutions for bipedal walking [15], [106], [121].

3) Implementation of Learning Control: Human walking is a marvel of coordination, all aspects of movement control need to be meticulously adjusted. In addition, the gait should be adaptive to different environments. For example, walking on ice is different from walking on solid ground, and walking uphill is different from downhill. No matter whether model-based or biologically inspired approaches are employed, there is an intrinsic need to equip bipedal robots with adaptive control strategies. Therefore, the key step of most control system designs becomes how one can formulate the control scheme so that the parameter tuning or policy adjustment can be easily and efficiently carried out while avoiding high computational workload for real-time implementation.

It is noticed that traditional adaptive control methods usually suffer from sophisticated parameter tuning process and often run into the problems of mathematical tractability, limited extensibility and limited biological plausibility. On the other hand, learning algorithms are generally less restrictive and are capable of acquiring appropriate control policies through an autonomously self-tuning process. Learning control has three distinguishable advantages as follows:

- Learning algorithms are capable of learning a good control solution automatically, thus do not highly rely on the modeling of the robot's dynamics.
- Learning controllers can easily adapt to changes of the robots' dynamics or environment. This means that a learning control scheme can be transferred from one robot to another even they have quite different dynamics.
- Control policies can be continuously improved with an increasing experience as the learning process proceeds.

Learning control is promising for walking robots that have to cope with unstructured environments without continuous human guidance. As shown in Figure 8, machine learning algorithms can be implemented in each mainstream of control methods to improve the control performance of adaptability, robustness and scalability [40], [90], [91]. The following section provides a comprehensive review of learning control techniques that have been applied to bipedal walking robots.

IV. Learning Algorithms for Bipedal Robot Control

In the following subsections, we discuss how learning algorithms have been applied to bipedal walking control.

A. Supervised Learning Approaches

Supervised learning (SL) methods learn to perform a task with the assistance of a teacher, who provides target input-output information to train a control system. A SL agent updates control parameters to minimize the difference between the desired and actual outputs of a system. Four popular SL learning approaches in bipedal walking control are discussed below.

1) Backpropagation-based Neural Control Methods: Wang et al. [112] trained a multilayer perceptron (MLP) to learn a predesigned controller for a 3-link bipedal robot via a standard backpropagation (BP) algorithm. Although the MLP was only trained to mimic a predesigned controller, the learned neural controller provided a superior performance against large disturbances, because of the NN's generalization. BP-based MLPs are often employed in trajectory tracking control of bipedal walking robots. For example, Juang et al. [45] applied a three-layer MLP to control a simulated five-link bipedal robot. A variation of the BP algorithm called *backpropagation through time* was employed to train the neural controller to drive the bipedal robot to follow a set of reference trajectories of hip and swing leg. After training, the bipedal robot was able to walk in a stable fashion on a flat surface. Later on the authors improved the neural control scheme by adding a slope-information MLP, which was trained to provide compensated control signals to enable the bipedal robot to walk on slopes. Shieh et al. [100] applied BP-based MLP to a real bipedal robot with 10 DOF. The MLP was trained to control joint angles to follow the desired ZMP trajectories. Experimental validation confirmed that the bipedal robot achieved a stable gait on a flat surface. It was also capable of adjusting the walking posture and keeping balanced walking when the ground was uneven or inclined.

BP-based neural control has gained popularity since it is relatively simple to implement and generally works well. However, the NNs obtained are usually very difficult to analyze and explain due to their complicated internal structure. A common disadvantage of BP-based methods is that the learning process is usually slow and inefficient. Moreover, the training may get stuck in local minima and result in sub-optimal solutions.

2) Locally Weighted Learning (LWL) Methods: Compared to BP-based neural learning methods, LWL methods offer a more understandable structure to learn complex nonlinear control policies. LWL approaches have achieved impressive success in some real-time humanoid robot learning control problems, such as complex inverse dynamics learning, and inverse kinematics learning [94]. Since LWL has low computational complexity for learning in high-dimensional spaces, it has demonstrated a very good potential to deal with high dimensional learning problems. Nakanishi et al. [72] applied LWL to train a 5-link biped to imitate human demonstrated walking trajectories. The trajectories of the robot were represented by a non-linear function approximator using local linear models. Through tuning of the parameters of local models, the LWL method enabled the biped to walk stably on a flat surface. Loken [63] applied LWPR to two bipedal robots with 3-link and 5-link, respectively. LWPR was used as an efficient function approximator that builds local linear regressions of adaptive nonlinear control policies. The locally structured control policies enabled the bipeds to follow the reference human walking motions on a flat surface very fast.

3) Support Vector Machine (SVM) Methods: SVM techniques provide powerful tools for learning classification and regression models in high-dimensional problems. A bipedal walking control system often has high-dimensional feedback sensory signals, SVM can be applied to classify feedback signals and provide categorized input signals to the control system. Kim et al. [53] applied SVM to detect the falling of a bipedal robot based accelerometer and force sensor data. Ferreira et al. [30] proposed a ZMP-based control strategy of walking balance using support vector regression (SVR). The ZMP-based controller was designed based on a simulated robot model. When implemented on the real bipedal robot, the designed controller would generate significant errors between the real and desired ZMP positions due to the difference between the real robot and its mathematical model. The difference between the real and desired ZMP positions can be offset by adaptively adjusting the angle of the robot's torso. The SVR was used to calculate the correction of the robot's torso based on the real ZMP positions and its variations to the desired ZMP positions. The training of SVR was based on simulation data and it successfully enabled the real bipedal robot to keep stable walking through adaptive torso control.

4) Decision Tree Methods: Decision tree methods have also been proposed to tackle the problems of adaptive walking control under varying environmental conditions. Miyashita et al. [69] designed a decision tree-based control system using C4.5. The tree-based adaptive control strategy enabled a bipedal robot to cope with several walking surfaces with different elasticity and viscous friction coefficients. Once a decision tree was obtained, the robot was capable of selecting appropriate control actions when it walked on different types of terrains.

B. Reinforcement Learning Approaches

We have discussed several successful examples of supervised learning for bipedal walking control. However, in many cases, it is extremely hard or expensive to find a good 'teacher', such as the gait trajectories on uneven surfaces. Moreover, learning only from a teacher allows a SL controller acting at most as good as the teacher. On the other hand, RL is powerful since a learning agent is not told which action it should take; instead it has to discover through interactions with the system and its environment which action yields the highest reward. In the following, the most popular RL methods for bipedal robot control are presented.

1) Actor-Critic Learning: Actor-critic learning generally approximate two functions separately, namely the state value function and the control policy function. Different function approximation methods result in different types of actor-critic methods as discussed in the following.

Multilayer Perceptron: RL has been widely used to train MLPs for bipedal robot walking. Salatian et al. [89], [90] applied RL to train a MLP controller for a simulated bipedal robot with 8 DOFs. The control system was designed to maintain the COP of the robot within the foot support region during walking. The foot force signals were used to calculate the position of COP. A MLP was trained by RL to map the relationship between the foot forces and the adjustment of joint positions. In particular, every joint of the robot was associated with a neuron called joint neuron; every joint neuron was attached to two pairs of neurons, called direction neurons. Each neuron possessed a value of activation function called neuron value. During the learning process, a joint neuron with the maximum neuron value was selected to modify the position of the corresponding joint, and the direction neuron was selected to determine the direction of the modification. If the selected joint and direction neuron result in a correct motion (the robot remains stable), this selection was reinforced by increasing the corresponding neuron value. Otherwise, the neuron value was reduced. The weights of the MLP were adjusted until the force sensors indicated that the robot had achieved a stable gait. The RL-trained MLP controller successfully made the bipedal robot walk on a flat surface. The biped was then placed on a slope and a new stable gait was found after 20 rounds of trials. However, since this study used a static walking stability criterion (COP), the resulting gait is very slow compared to normal dynamic walking.

Neural Oscillator: Neural oscillators have become a focus of interest in bipedal walking control in recent years [11]. The most popular method is called CPG as we have mentioned in Section III-B2. Neural oscillators with appropriate weight settings are capable of generating different types of stable walking patterns [73]. This kind of methods is discussed here since most neural oscillator-based controllers are trained by RL algorithms in the bipedal robot literature. The basic structure of a typical neural oscillator is shown in Figure 9 (a), and the schematic structure of a general neural oscillator-based control system for bipedal robots is given in Figure 9 (b).

Mori et al. [70] presented a CPG-based actor-critic RL controller. There were 12 pairs of neurons; each composed of a primary neuron and a supplementary neuron. Each supplementary neuron was solely connected to its primary neuron by excitation-inhibition mutual connections. A combination of two primary neurons and two supplementary neurons behaved as a neural oscillator. Each neural oscillator was responsible for controlling one joint of a robot. The neural oscillators were trained by an actor-critic RL algorithm. The actor (neural oscillators) mapped the sensory feedback signals into joint torques, and the critic predicted the expected cost in the future. The parameters of the actor were updated so that the future cost predicted by the critic became smaller. The critic was updated based on a policy gradient method. A lower-dimensional projection of the value function was used to reduce the complexity of estimating the original value function

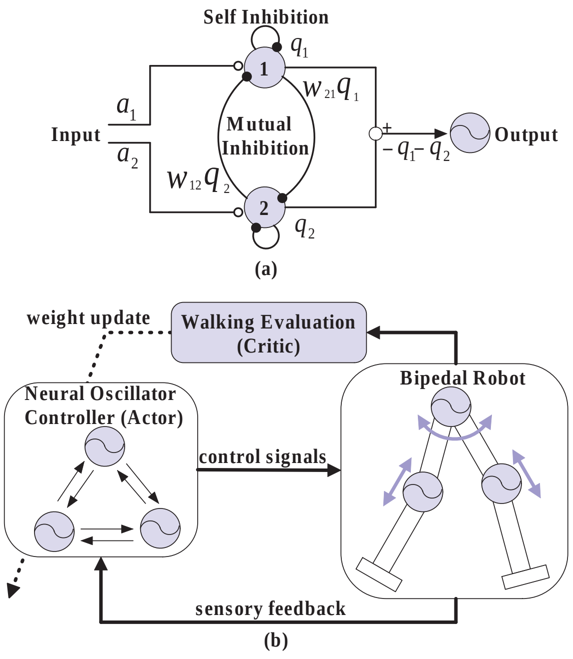

Fig. 9. (a): Schematic structure of a coupled neural oscillator. (b): Basic structure of a neural oscillator-based actor-critic RL controller.

in a high-dimensional space. After 50,000 learning episodes, the simulated biped achieved stable walking on a flat surface. The gait learned was also robust to environmental disturbances such as up and down slopes. Their simulation experiments were quite successful. However, one big disadvantage of the method is that too many training episodes were required. A real robot cannot afford so many failures during the training.

Matsubara et al. [66] combined a CPG-based RL controller with a state-machine. The CPG controller was composed of two pairs of extensor/flexor neurons to exert hip torques to the left and right legs, respectively. The state-machine controlled the knee joints according to the four transition states defined by the hip joint angles and the foot placement information. A policy gradient method was used to train the neural oscillators. The CPG-based learning controller was able to acquire an appropriate control policy after a few hundred of simulated trials. The controller trained in simulation was successfully applied to a 5-link 2D real bipedal robot. This study demonstrated that the proposed RL controller was robust against the mismatch between the simulation model and the real robot as well as small ground disturbances.

In most neural-oscillator-based controllers, each oscillator is allocated at a joint and exerts joint torque to drive walking motions. As the number of neural-oscillators increases, it becomes more difficult to obtain appropriate cooperation and coordination for all the oscillators, especially for the cases of a robot system with many DOFs. Endo et al. [26], [27] proposed a novel arrangement of neural-oscillators, which only uses six pairs of neural-oscillators to control a 3D full-body humanoid robot with 38 DOFs. A policy-gradient-based actor-critic RL algorithm was used to train the neural-oscillator-based controller. At first, the control scheme was applied to a simulated bipedal robot. It took 1,000 trials on average to enable the biped to walk stably on a flat surface. The RL controller obtained from simulation was successfully implemented on a 3D real robot. Most recently, Park et al. [75] developed a CPG controller to generate full-body joint trajectories for a real 26-DOF bipedal robot, called HSR-IX. The neural oscillators in the CPG were designed to generate rhythmic control signals for each joint. The parameters of the CPG controller were optimized by a quantum-inspired evolutionary algorithm using a simulated robot model. The optimized CPG controller was then applied to the real robot, which was able to walk stably on a flat surface using the fine-tuned CPG parameters in real experiments.

Cerebellar Model Arithmetic Controller: CMAC was first created as a simple model of the cortex of cerebellum by Albus in 1975 [3]. Since then it has been used in a wide range of applications. Besides its biological relevance, the main reason for using CMAC is that it operates very fast, and has a potential in real-time control problems. A schematic structure of CMAC learning is shown in Figure 10.

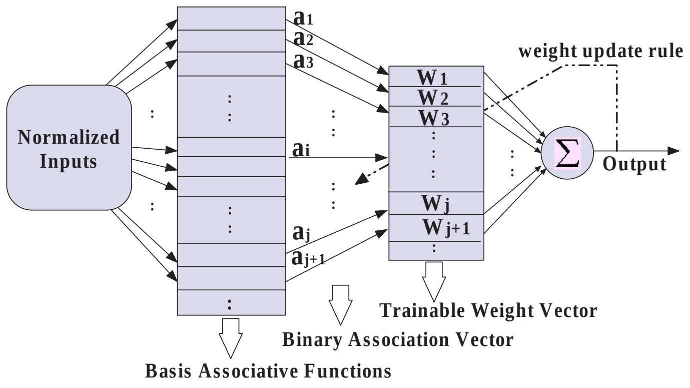

Fig. 10. Schematic representation of CMAC learning.

Miller [40] presented a hierarchical controller which combines three CMAC networks, two of which were used for front/back balance and right/left balance, and the third one was used to learn kinematically consistent robot postures. The training of the CMAC networks was realized by RL. The reward function was defined by the difference between the desired and measured foot placement on the ground. The proposed learning controller was applied to a real ten-axis bipedal robot. After training, the bipedal robot was capable of keeping dynamic balance on a flat surface. However, the resulting walking speed was very slow and was also sensitive to ground disturbances. Kun et al. [59] proposed an improved approach. The complete control structure consisted of high-level and low-level controllers. The high-level controller had seven components: gait generator, simple kinematics block and five CMAC controllers. The CMACs were used for compensation of right and left lift-lean angle correction, reactive front-back offset, right-left lean correction, right and left ankle correction and front-back lean correction. The training of the CMACs was realized by RL. The reward was defined based on the ZMP, which can be calculated from foot force signals [110]. The proposed RL controller enabled a complex 3D humanoid robot to maintain dynamical walking balance. However, more research efforts are needed to increase the walking speed to achieve natural dynamic walking. Smith proposed a CMAC controller called FOX [101]. The weights of the CMAC were updated by RL with an eligibility trace assigned to each weight. The eligibility was used to update weights in a manner analogous to the cerebellar modulation of spinal cord reflexes in human movement. The proposed control scheme was applied to a simulated bipedal robot with 18 DOFs. The simulated bipedal robot was able to walk with flexible gait patterns on both flat and slope surfaces.

In general, CMAC has the quality of fast learning and efficient digital hardware implementation due to its special architecture. However, a serious drawback of CMAC is its large memory requirement. Especially when the state space is high dimensional, CMAC may become impractical to implement due to the huge memory it requires.

Function Approximators: Various function approximators are also employed to estimate state value function and control policy function. Since most function approximators used in RL are usually differentiable, the policy gradient-based RL algorithms play an important role in this type of methods. An excellent example is that of Tedrake [104], who applied a policy gradient-based actor-critic RL controller to a 3D 9-DOF real bipedal robot. Both the control policy function and the state value function were represented by a linear combination of basis functions. All the parameters of the control policy and state values were initialized at zero. The unactuated robot exhibited passive dynamic walking down a mild slope of 0.03 radians, which was taken as the reference walking pattern. Several fixed points on the corresponding Poincaré map of the reference pattern were used to train the actor-critic RL controller. The reward was given by the difference between the actual and desired fix points on the return map. The control policy and the state values were both updated by the TD (temporal difference) error. The most attractive part of this work is that the robot was able to learn a stable walking pattern from scratch. In particular, the robot was able to learn in about one minute to start walking from standing still. The walking orbit converged to the desired limit cycle in less than 20 minutes on average.

Morimoto et al. [71] applied Receptive Field Weighted Regression (RFWR) [93] as a function approximator for the control policy and the state-value functions in an actor-critic RL framework. The proposed RL controller was tested on a 5-link real bipedal robot. The walking performance was evaluated by comparing four fixed points on the Poincaré map with their reference values extracted from human walking patterns. The robot acquired a control policy of stable walking after about 100 trials of learning on a flat surface.

Most of the existing learning methods only focus on numerical evaluative information. However in real life, we often use linguistic critical signals such as ‘near fall down’, ‘almost success’, ‘slow’, ‘fast’ to evaluate human walking. Fuzzy evaluation feedback signals are considered to be much closer to human learning in real world [12]. A number of researchers have incorporated fuzzy-logic in designing RL controllers for bipedal robots [43], [118], [51]. A general flowchart of the information integration for a fuzzy logic-based controller is shown in Figure 11.

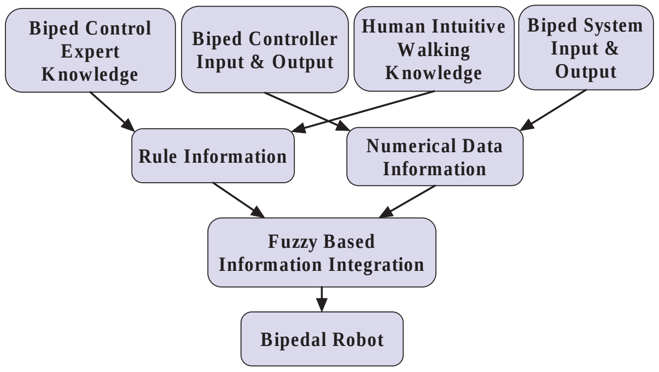
Biped Control Expert Knowledge
Biped Controller Input & Output
Human Intuitive Walking Knowledge
Biped System Input & Output
Rule Information
Numerical Data Information
Fuzzy Based Information Integration
Bipedal Robot

Fig. 11. Fuzzy-based linguistic-numerical information integration for bipedal walking control.

Zhou et al. [122], [123] applied fuzzy logic to a RL-based neural controller (Figure 12), which consisted of three parts: action selection network (ASN), action evaluation network (AEN), and stochastic action modifier (SAM). Both ASN and AEN were constructed in neuro-fuzzy architectures in the form of 5-layer NNs, while the SAM was used to make a trade-off between exploration and exploitation during learning. The proposed learning structure was actually a modified version of actor-critic RL. The critic (AEN) was updated by TD error, the actor (ASN) was updated by the BP algorithm. The reward was generated by a fuzzy rule base, which represented the expert knowledge derived based on the ZMP stability criterion. The proposed fuzzy-RL controller was tested with a simulated bipedal robot.

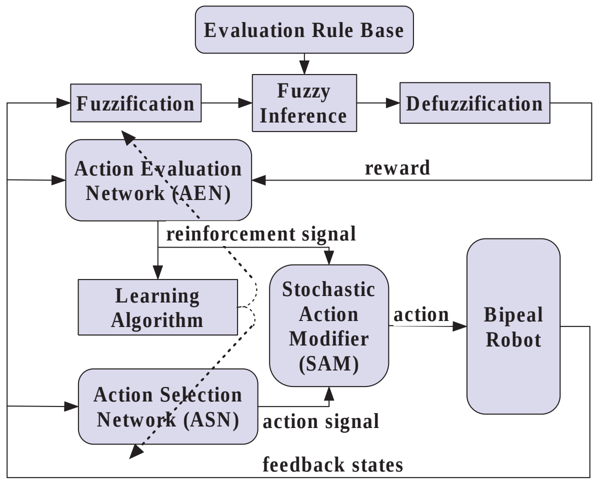

Fig. 12. Architecture of a RL controller with fuzzy evaluative feedback [123].

Most recently, Katić et al. [51] proposed a fuzzy logic integrated control structure. The control system consisted of two parts. A dynamic controller was used to track a pre-designed nominal walking trajectory; a fuzzy actor-critic RL controller was used to make efficient compensation of ZMP reactions during walking. The walking performance (reward) was evaluated by fuzzy rules obtained from human intuitive knowledge. Based on tracking errors and rewards, the critic generated reinforcement signals, by means of which the TD error was calculated and used to update the actor and the critic. Fuzzy evaluation was considered much closer to the human's evaluation than regular numerical values. Their simulation results also showed that fuzzy evaluation considerably sped up the learning process.

Integration of Evolutionary Computing: Evolutionary computation techniques such as genetic algorithms (GAs) have been widely used for many complex problems in optimization and machine learning [34], [115]. Some researchers have also incorporated evolutionary computation in a RL framework to obtain optimal control solutions for bipedal robots. A typical example in this area comes from Zhou et al. [121] who proposed a GA-based actor-critic RL controller for bipedal robot walking. It differs from the traditional actor-critic methods in that the actor was updated by a GA instead of using the TD error, while the critic was still updated by the TD error. With the global optimization capability of GA, the learning controller was able to solve the local minima problem of the traditional gradient-based actor-critic RL algorithms.

2) Q-Learning: Instead of constructing the critic and actor functions separately, Q-learning builds a single value function called Q-value function, in the (discretized) state-action space. RL with tabular Q-value function has been proven to converge to the optimal policy as the number of trials tends to infinity [52]. Compared to actor-critic algorithms, Q-learning is easier to implement since the Q-function is actually a lookup table indexed by discrete state-action pairs. There are several applications of Q-learning to bipedal walking robot control.

Wang et al. [113] proposed a Q-learning controller for a simulated two-link passive dynamic walking robot, which is an abstraction of a mechanical prototype. The state represented the velocity of the stance leg, and the action was an additional torque applied to the hip joint. Simulation results demonstrated that the bipedal robot quickly learnt to apply additional hip torque to adapt its walking gaits to ground disturbances within 20 trials. The bipedal robot was able to walk through a test scenario with 16 different step-down disturbances, which were up to 10 percent of the leg length. Schuitema et al. [97] applied Q-learning to a 7-link simulated bipedal robot. The state space of the bipedal walking problem consisted of 6 dimensions: angle and angular velocity of upper stance leg, upper swing leg, and the lower swing leg. The action was the torque exerted to the hip joint. The total 7-dimensional state-action space resulted in a large Q-table with 1,000,000 state-action pairs. Simulation results showed that a stable gait was achieved on a flat surface within 20 minutes of learning on average. Er and Deng [28] proposed a novel fuzzy Q-learning (FQL) framework, which was capable of generating and tuning fuzzy rules automatically by the self-organizing fuzzy inference. Er and Zhou [29] then applied this learning framework to enable a bipedal robot to walk on uneven terrains by using adaptive trunk control. The FQL system was started with an initial set of fuzzy rules, and learned to improve the ZMP stability through RL and fuzzy-rule updating. Simulation results showed that their bipedal robot achieved a good ZMP stability on uneven surfaces. Chew and Pratt [18] applied Q-learning to a 3D biped model with 6 DOFs for each leg. The Q-learning algorithm was employed to train a CMAC network, which successfully learned the control strategy of the swing leg to achieve stable walking with variable desired walking speed.

C. Unsupervised Learning Approaches

Unsupervised learning (UL) does not need a teacher or any evaluative feedback to acquire a control policy. Instead it builds underline structures or associative networks for input data. For bipedal robot control, there are two main UL approaches in the literature: clustering methods and Hebbian learning. Clustering techniques discover structures in data, while Hebbian learning primarily aims at finding an associative network between inputs and control actions.

1) Clustering: Clustering is a very active field of research. It is usually not used to learn control policies directly; instead it plays a role in the analysis and reduction of raw data. For example, we have mentioned that CMAC-based neural controllers have fast computation but require large memory. Hu et al. [39] applied a clustering technique in a bipedal walking system to reduce the memory requirement of a CMAC-based learning controller.

2) Differential Hebbian Learning: Unsupervised Hebbian learning has not been studied for bipedal robot control until the recent studies of Porr and Wörgötter [80], [79], [81]. They developed a modified version of classical Hebbian learning, differential Hebbian learning, which is applicable to closed loop control systems. The basic architecture of Hebbian learning control is shown in Figure 13. The control signal is derived from the correlations between two temporally related input signals: one is an early input x_1 called presynaptic activity and the other one is a later input x_0 called postsynaptic or reflex activity. Each time when the robot falls, a strong reflex signal is triggered. The reflex signal together with the predictive signal drives the weight updating in Hebbian learning. The learning goal is to change the gait parameters in an appropriate way in order to prevent the robot from falling.

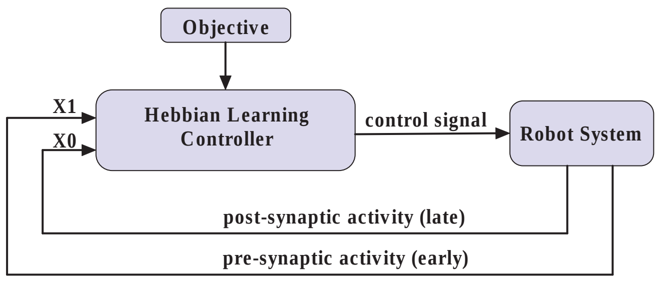
Objective
X1
X0
Hebbian Learning Controller
control signal
Robot System
post-synaptic activity (late)
pre-synaptic activity (early)

Fig. 13. Architecture of Hebbian learning Control.

An impressive application of differential Hebbian learning to real bipedal robot control was conducted by Manoonpong et al. [64], [65]. They designed an adaptive neuronal control system for a real bipedal robot called RunBot, which has four active leg joints (left/right hips and knees) and an upper body component that can be actively moved backward or forward to shift the center of mass. The neuronal control scheme has two modules: one controls leg joints and the other controls the upper body component. The neuronal controllers have a distributed implementation at each active joint. The differential Hebbian learning rule was applied to adjust the synaptic strengths of neurons according to the temporal relation between their inputs and outputs. Without any explicit gait calculation or trajectory control, the neuronal control network was capable of synchronizing the leg and body movements of the robot for a stable locomotion. In addition, with learned parameters on a flat surface, the robot was also able to adapt its gaits to an 8-degree ramp after only 3 to 5 falls. The most attractive part of this study is that the obtained stable walking fully relies on its neuronal control network in an unsupervised manner.

V. Conclusion and Outlook

This paper we first gave an overview of the state-of-the-art learning algorithms, and then discussed their applications to bipedal walking robots according to three learning paradigms, namely supervised learning, reinforcement learning, and unsupervised learning. Each learning strategy has its merits as well as drawbacks. A comparison of the learning methods discussed is summarized in Table II. In general, the theory of learning control is still in its infancy, and has to cope with several challenges. Firstly, many sophisticated machine learning algorithms (e.g., reinforcement learning, Hebbian Learning) are still not understood well enough to always converge in acceptable time for real robot control. Theoretical guarantee of convergence are not always available. Secondly, a real world robot typically cannot afford many training and evaluation runs. Learning algorithms need to converge faster in practice with an estimate of convergence rates and training times. Moreover, the learning parameters of many learning algorithms (such as NNs) are often difficult to set.

This comprehensive survey demonstrated that learning control techniques achieved impressive results in many bipedal walking control problems. However, the performance of learning control systems for real-time high-dimensional bipedal robots is still by far not good enough in terms of stability, adaptability and robustness. As the complexity of bipedal walking control systems scales up in complex environments, the problem of cooperation of many different actuators becomes severe in high dimensional spaces. Therefore, constructing a hierarchical learning architecture might be promising to tackle complex control problems in high dimensional spaces. Hierarchical learning approaches decompose a problem into subproblems which can work with smaller state spaces and simpler control functions. The local solutions of the subproblems can be combined to solve the original problem. Careful decomposition of a complex control problem in a hierarchical way really helps reduce the original problem into a tractable one. However, how to make proper hierarchical learning on real-time bipedal walking robots is still a challenging and less studied research area.

Human brain undoubtedly implements the most efficient learning control system available to date. It is believed that human beings make full use of the three learning paradigms: unsupervised learning, supervised learning, and reinforcement learning. In our view, as shown in Figure 14, the effective integration of the three learning paradigms as well as strategic

TABLE II
A COMPARISON OF DIFFERENT CONTROL STRATEGIES.

Learning Mechanism	Algorithms	Advantages	Disadvantages	References
Supervised Learning	BP	• is suitable to train a NN to learn complex relationships between input and output patterns, such as inverse kinematic and inverse dynamics of a robot model.	• is gradient based, may get trapped in local minima. • slows down dramatically as the number of hidden layers and neurons in a NN increases.	Wang et al. [112], Juang and Lin [45], Juang [44]
	SVM	• leads to global optimization models. • has great ability to classify feedback signals and provides categorized input signals to a control system. • its computational complexity does not depend on the dimensionality of input space.	• is sensitive to the choice of kernel functions, which are still left up to the user. • the inherent quadratic programming requires extensive memory in large-scale tasks. • optimal design of multiclass SVM is still an open problem.	Ferreira et al. [30], Kim et al. [53], Kang et al. [49]
	LWL	• achieves low computational complexity and higher flexibility to approximate complex nonlinear functions (such as robot models) by local kernel functions. • is capable of fast and efficient learning in high dimensional space.	• may have difficulties to define locality appropriately. • its results are sensitive to the choice of distance metric, the number of linear functions, and the effective range of local kernel functions.	Atkeson et al. [6], [7], Nakanishi et al. [72], Loken [63]
	Decision Tree	• is easy to understand and implement, and does not require expensive modeling. • is easy to encode expert knowledge in a tree model. • works fast and can perform well for large data set in a short time.	• may fail if data are very noisy. • ignores the relationships among the attributes of input data.	Miyashita et al. [69], Wong et al. [117]
Reinforcement Learning	Actor-Critic	• responds to smoothly varying states with smoothly varying actions. • constructs a continuous mapping from states to actions. It is advantageous to treat both states and actions as continuous variables in many control problems. • has good potentials to learn in high dimensional continuous spaces.	• training process is more complex compared to Q learning, since both state function and action function are required to be properly adjusted. • is sensitive to its exploration policy, since state value is updated based on the current action policy. • usually has low learning rate.	Zhou and Meng [122], [123], Tedrake et al. [104], Katić et al. [51], Matsubara et al. [66], Smith [101]
	Q-learning	• is easy to understand, implement, and tune learning parameters, since learning problem is reduced to search a lookup table of state-action pairs. • requires low computation and has fast learning rate with convergence guarantees in theory. • is exploration-insensitive, since it learns without necessarily following the current policy.	• learning rate increases exponentially as the number of states and actions increases. • may have poor generalization for unexplored state-action spaces. • generally only considers states and actions in discrete space.	Er and Deng [28], Schuitema et al. [97], Wang et al. [113]
Unsupervised Learning	Clustering	• is effective to extract knowledge from large amount of data, such as vision data. • can be used for preprocessing feedback signal in a control system.	• is a subjective process, which is sensitive to feature extraction and similarity measures.	Kang et al. [49], Hu et al. [39]
	Hebbian Learning	• mimics human reflex control. • exhibits natural walking patterns in some real-time experiments, which seem to be biologically plausible.	• needs carefully design the timing and coordination of each neural oscillator.	Porr et al. [80], [79], [81]

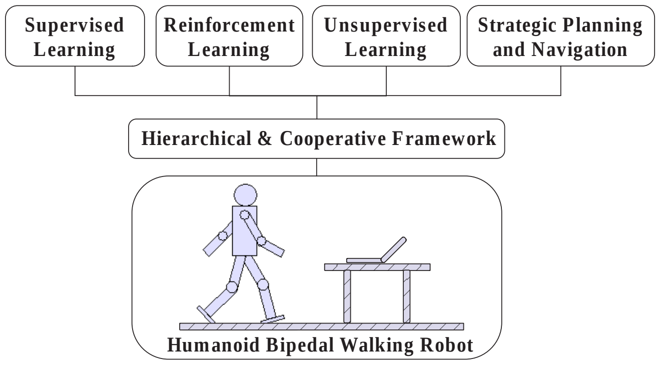
Supervised Learning
Reinforcement Learning
Unsupervised Learning
Strategic Planning and Navigation
Hierarchical & Cooperative Framework
Humanoid Bipedal Walking Robot

Fig. 14. Hierarchical integration of robot learning control.

planning tools in a hierarchical framework should be an inevitably trend in designing learning control systems for future intelligent bipedal walking robots. The great potentials and capabilities of bipedal robots have not been fully utilized. The performance improvements that bipedal robots can gain by incorporating suitable learning control techniques are huge.

REFERENCES

[1] D. Aha. Lazy learning. *Artificial Intelligence Review*, 11:325–337, 1997.

[2] A. Albert and W. Gerth. Analytic path planning algorithms for bipedal robots without a trunk. *Journal of Intelligent and Robotic Systems*, 36:109–127, 2003.

[3] J.S. Albus. A new approach to manipulator control: the cerebellar model articulation controller (CMAC). *Transactions of the ASME: Journal of Dynamic Systems, Measurement, and Control*, 97:220–227, 1975.

[4] P.D. Alevizos, D.K. Tasoulis, and M.N. Vrahatis. Parallelizing the unsupervised k-windows clustering algorithm. In *Lecture Notes in Computer Science*, volume 3019, pages 225–232. Springer-Verlag, 2004.

[5] Anon. Logistical vehicle off-road mobility. Technical report, U.S. Army Transprotation Combat Developments Agency, Fort Eustis, Virginia, 1967. Project TCCO 62-5.

[6] C.G. Atkeson, A.W. Moore, and S. Schaal. Locally weighted learning. *Artificial Intelligence Review*, 11:11–73, 1997.

[7] C.G. Atkeson, A.W. Moore, and S. Schaal. Locally weighted learning for control. *Artificial Intelligence Review*, 11:75–113, 1997.

[8] C. Azevedo, P. Poignet, and B. Espiau. Artificial locomotion control: from human to robots. *Robotics and Autonomous Systems*, 47(4):203–223, 2004.

[9] S. Baik and J. Bala. A decision tree algorithm for distributed data mining: Towards network intrusion detection. In *Lecture Notes in Computer Science*, volume 3046, pages 206–212, 2004.

[10] M.G. Bekker. *Introduction to Terrain Vehicle Systems*. University of Michigan Press, March 1969.

[11] H. Benbrahim and J.A. Franklin. Biped dynamic walking using reinforcement learning. *Robotics and Autonomous Systems*, 22:283–302, 1997.

[12] H.R. Berenji and P. Khedkar. Learning and tuning fuzzy logic controllers through reinforcements. *IEEE Transactions on Neural Networks*, 3:724–740, 1992.

[13] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and regression trees*. Wadsworth & Brooks/Cole Advanced Books & Software, Monterey, CA, 1984.

[14] C. Burges. Tutorial on support vector machines for pattern reccognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[15] G. Capi, Y. Nasu, L. Barolli, K. Mitobe, and K. Takeda. Application of genetic algorithms for biped robot gait synthesis optimization during walking and going up-stairs. *Advanced Robotics*, 15(6):675–694, 2001.

[16] C. Chevallereau. *Bipedal Robots: Modeling, Design and Walking Synthesis*. Wiley-ISTE, Dec. 2008.

[17] C. Chevallereau and P. Sardain. Design and actuation optimization of a 4 axes biped robot for walking and running. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3365–3370, San Francisco, CA, Apr. 2000.

[18] C.M. Chew and G.A. Pratt. Dynamic bipedal walking assisted by learning. *Robotica*, 20:477–491, 2002.

[19] S.H. Collins, A. Ruina, R. Tedrake, and M. Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307(5712):1082–1085, 2005.

[20] N. Cristianini and J.S. Taylor. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, Cambridge, 2000.

[21] J. Denk and G. Schmidt. Walking primitive synthesis for an anthropomorphic biped using optimal control techniques. In *Proceedings of the International Conference on Climbing and Walking Robots*, pages 819–826, Karlsruhe, Germany, Sep. 2001.

[22] M. Dorigo. Editorial introduction to the special issue on learning autonomous robots. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, 26(3):361–364, June 1996.

[23] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. John Wiley & Sons, New York, 2nd edition, 2001.

[24] J.C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3:32–57, 1973.

[25] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng. Learning CPG sensory feedback with policy gradient for biped locomotion for a full-body humanoid. In *Proceedings of the 20th national conference on Artificial intelligence*, pages 1267–1273, Pittsburgh, Pennsylvania, Jul. 2005.

[26] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng. Learning CPG-based biped locomotion with a policy gradient method: application to a humanoid robot. *The International Journal of Robotics Research*, 27(2):213–228, 2008.

[27] G. Endo, J. Nakanishi, J. Morimoto, and G. Cheng. Experimental studies of a neural oscillator for biped locomotion with QRIO. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 598–604, Barcelona, Spain, Apr. 2005.

[28] M.J. Er and C. Deng. Online tuning of fuzzy inference systems using dynamic fuzzy Q-learning. *IEEE Transaction on Systems, Man and Cybernetics, Part B*, 34:1478–1489, 2004.

[29] M.J. Er and Y. Zhou. Intelligent fuzzy Q-learning control of humanoid robots. In *The Second International Symposium on Neural Networks*, volume 3498, pages 216–221, Chongqing, China, May 2005.

[30] J.P. Ferreira, M. Crisostomo, A.P. Coimbra, and B. Ribeiro. Simulation control of a biped robot with support vector regression. In *IEEE International Symposium on Intelligent Signal Processing*, pages 1–6, 2007.

[31] W.T. Fu and J.R. Anderson. From recurrent choice to skill learning: A reinforcement-learning model. *Journal of Experimental Psychology: General*, 135(2):184–206, 2006.

[32] T. Geng, B. Porr, and F. Wörgötter. Fast biped walking with a sensor-driven neuronal controller and real-time online learning. *International Journal of Robotic Research*, 25(3):243–259, 2006.

[33] M. Ghavamzadeh, S. Mahadevan, and R. Makar. Hierarchical multi-agent reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 13:197–229, 2006.

[34] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Kluwer Academic Publishers, Boston, MA, 1989.

[35] A. Goswami, B. Espiau, and A. Keramane. Limit cycles in a passive compass gaitbiped and passivity-mimicking control laws. *Autonomous Robots*, 4(3):273–286, 1997.

[36] Ambarish Goswami. Foot Rotation Indicator (FRI) Point: A New Gait Planning Tool to Evaluate Postural Stability of Biped Robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 47–52, Detroit, MI, May 1999.

[37] D.O. Hebb. *The Organization of Behaviour*. John Wiley & Sons, New York, 1949.

[38] M. Hirose and K. Ogawa. Honda humanoid robots development. *Philosophical Transactions of the Royal Society A*, 365(1850):11–19, 2007.

[39] J. Hu, J. Pratt, and G. Pratt. Stable adaptive control of a bipedal walking robot with CMAC neural networks. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 1950–1956, Detroit, USA, May 1999.

[40] W. Thomas Miller III. Real-time neural network control of a biped walking robot. *IEEE Control Systems Magazine*, 14(1):41–48, 1994.

[41] E. Januzaj, H.P. Kriegel, and M. Pfeifle. Towards effective and efficient distributed clustering. In *Workshop on Clustering Large Data Sets (ICDM)*, pages 49–58, Melbourne, FL, Nov. 2003.

[42] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, volume 1398, pages 137–142, Chemnitz, Germany, Apr. 1998.

[43] J.G. Juang. Fuzzy neural network approaches for robotic gait synthesis. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, 30(4):594–601, 2000.

[44] J.G. Juang. Intelligent locomotion control on sloping surfaces. *Information Sciences*, 147:229–243, 2002.

[45] J.G. Juang and C.S. Lin. Gait synthesis of a biped robot using back propagation through time algorithm. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, volume 3, pages 1710–1715, Washington DC, USA, Jun. 1996.

[46] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Yokoi, and H. Hirukawa. Biped walking pattern generation by a simple three-dimensional inverted pendulum model. *Advanced Robotics*, 17:131–147, 2003.

[47] S. Kajita and K. Tani. Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1405–1411, Apr. 1991.

[48] S. Kakade. A natural policy gradient. *Advances in Neural Information Processing Systems*, 14(26):1531–1538, 2002.

[49] T.K. Kang, H. Song, D. Kim, and G.T. Park. Environment recognition system for biped walking robot using vision based sensor fusion. *New Trends in Applied Artificial Intelligence*, 4570:405–414, 2007.

[50] D. Katić and M. Vukobratović. Survey of intelligent control techniques for humanoid robots. *Journal of Intelligent and Robotic Systems*, 37(2):117–141, 2003.

[51] D. Katić and M. Vukobratović. Control algorithm for humanoid walking based on fuzzy reinforcement learning. In *The 4th Serbian-Hungarian Joint Symposium on Intelligent Systems*, pages 81–93, Subotica, Serbia, Sep. 2006.

[52] M. Kearns and S. Singh. Finite-sample convergence rates for Q-learning and indirect algorithms. *Neural Information Processing Systems*, 12:996–1002, 1999.

[53] J.J. Kim, T.Y. Choi, and J.J. Lee. Falling avoidance of biped robot using state classification. In *Proceedings of 2008 IEEE International Conference on Mechatronics and Automation*, pages 72–76, Takamatsu, Japan, Aug. 2008.

[54] A.H. Klopf. A drive-reinforcement model of single neuron function: an alternative to the Hebbian neuronal model. *AIP Conference Proceedings on Neural Networks for Computing*, pages 265–270, 1987.

[55] N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proceedings of the IEEE international conference on robotics and automation*, pages 2619–2624, New Orleans, LA, May 2004.

[56] V.R. Konda and J.N. Tsitsiklis. Actor-critic algorithms. In *SIAM Journal on control and optimizaiton*, pages 1008–1014. MIT Press, 2001.

[57] H. Kooij, R. Jacobs, B. Koopman, and F. Helm. An alternative approach to synthesizing bipedal walking. *Biological Cybernetics*, 88(1):46–59, 2003.

[58] B. Kosko. Differential hebbian learning. *AIP Conference Proceedings on Neural Networks for Computing*, pages 277–282, March 1987.

[59] A.L. Kun and W. Thomas Miller III. Control of variable-speed gaits for a biped robot. *IEEE Robotics and Automation Magazine*, 6(3):19–29, 1999.

[60] I. Kwee, M. Hutter, and J. Schmidhuber. Market-based reinforcement learning in partially observable worlds. In *Proceedings of International Conference on Artificial Neural Networks*, pages 865–873, Vienna, Austria, Aug. 2001.

[61] W. Li, Q.T. Ye, and C.M. Zhu. Application of hierarchical reinforcement learning in engineering domain. *Journal of Systems Science and Systems Engineering*, 14(2):207–217, Jul. 2005.

[62] L. Ljung and T. Söderström. *Theory and practice of recursive identification*. Cambridge MIT Press, 1986.

[63] K. Loken. Imitation-based learning of bipedal walking using locally weighted learning. Master's thesis, Computer Science Department, The University of British Columbia, 2006.

[64] P. Manoonpong, T. Geng, T. Kulvicius, B. Porr, and F. Wörgötter. Adaptive, fast walking in a biped robot under neuronal control and learning. *PLoS Computational Biology*, 3(7):e134, 2007.

[65] P. Manoonpong and F. Wörgötter. Efference copies in neural control of dynamic biped walking. *Robotics and Autonomous Systems*, 57(11):1140–1153, 2009.

[66] T. Matsubara, J. Morimoto, J. Nakanishi, M. Sato, and K. Doya. Learning CPG-based biped locomotion with a policy gradient method. *Robotics and Autonomous Systems*, 54:911–920, 2006.

[67] T. McGeer. Passive dynamic walking. *International Journal of Robotics Research*, 9(2):62–82, 1990.

[68] T.A. McMahon. *Muscles, Reflexes, and Locomotion*. Princeton University Press, 1984.

[69] T. Miyashita, K. Shinozawa, N. Hagita, and H. Ishiguro. Behavior selection and environment recognition methods for humanoids based on sensor history. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3468–3473, Beijing, China, Oct. 2006.

[70] T. Mori, Y. Nakamura, M. Sato, and S. Ishii. Reinforcement learning for a CPG-driven biped robot. In *Proceedings of the 19th National Conference on Artificial Intelligence*, pages 623–630, San Jose, California, Jul. 2004.

[71] J. Morimoto, G. Cheng, C.G. Atkeson, and G. Zeglin. A simple reinforcement learning algorithm for biped walking. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 3, pages 3030–3035, New Orleans, LA, Apr. 2004.

[72] J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato. Learning from demonstration and adaptation of biped locomotion. *Robotics and Autonomous Systems*, 47:79–91, 2004.

[73] G.N. Orlovsky, T.G. Deliagina, and S. Grillner. *Neuronal Control of Locomotion: From Mollusc to Man*. Oxford University Press, 1999.

[74] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: an application to face detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 130–136, San Juan, Puerto Rico, Jun. 1997.

[75] Chang-Soo Park, Young-Dae Hong, and Jong-Hwan Kim. Full-body joint trajectory generation using an evolutionary central pattern generator for stable bipedal walking. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 160–165, oct. 2010.

[76] J. Park and H. Chung. ZMP compensation by on-line trajectory generation for biped robots. In *Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, pages 960–965, Tokyo, Japan, Oct. 1999.

[77] J. Peters and S. Schaal. Policy gradient methods for robotics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2219–2225, Beijing, China, Oct. 2006.

[78] J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21:682–697, 2008.

[79] B. Porr, C. Ferber, and F. Wörgötter. ISO-learning approximates a solution to the inverse-controller problem in an unsupervised behavioral paradigm. *Neural Computation*, 15:865–884, 2003.

[80] B. Porr and F. Wörgötter. Isotropic sequence order learning. *Neural Computation*, 15:831–864, 2003.

[81] B. Porr and F. Wörgötter. Isotropic sequence order learning in a closed loop behavioural system. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 361(1811):2225–2244, 2003.

[82] B. Porr and F. Wörgötter. Strongly improved stability and faster convergence of temporal sequence learning by utilising input correlations only. *Neural Computation*, 18(6):1380–1412, 2006.

[83] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[84] J.R. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, 1993.

[85] E. Rafols, M. Ring, R.S. Sutton, and B. Tanner. Using predictive representations to improve generalization in reinforcement learning. In *Proceedings of International Joint Conferences on Artificial Intelligence (IJCAI)*, pages 835–840, Edinburgh, Scotland, Aug. 2005.

[86] S. Richter, D. Aberdeen, and J. Yu. Natural actor-critic for road traffic optimisation. In *Advances in Neural Information Processing Systems*, volume 19, pages 1169–1176. MIT Press, 2007.

[87] S. Ruggieri. Efficient C4.5. *IEEE Transactions on Knowledge and Data Engineering*, 14(2):438–444, 2001.

[88] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, 1:318–362, 1986.

[89] W. Salatian, K.Y. Yi, and Y.F. Zheng. Reinforcement learning for a biped robot to climb sloping surfaces. *Journal of Robotic Systems*, 14(4):283–296, 1997.

[90] W. Salatian and Y.F. Zheng. Gait synthesis for a biped robot climbing sloping surfaces using neural networks. I. Static learning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 3, pages 2601–2606, Nice, France, May 1992.

[91] W. Salatian and Y.F. Zheng. Gait synthesis for a biped robot climbing sloping surfaces using neural networks. II. Dynamic learning. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 3, pages 2607–2611, Nice, France, May 1992.

[92] P. Sardain and G. Bessonnet. Forces acting on a biped robot. center of pressure-zero moment point. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 34:630–637, 2004.

[93] S. Schaal and C. G. Atkeson. Constructive incremental learning from only local information. *Neural Computation*, 10(8):2047–2084, 1998.

[94] S. Schaal, C.G. Atkeson, and S. Vijayakumar. Scalable techniques from nonparametric statistics for real time robot learning. *Applied Intelligence*, 17(1):49–60, 2002.

[95] M.S. Schmidt. Identifying speakers with support vector networks. *Interfaces*, 1996.

[96] B. Scholkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In *Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining*, pages 252–257, Montreal, Canada, Aug. 1995.

[97] E. Schuitema, D.G.E. Hobbelen, P.P. Jonker, M. Wisse, and J.G.D. Karssen. Using a controller based on reinforcement learning for a passive dynamic walking robot. In *Proceedings of the 5th IEEE-RAS International Conference on Humanoid Robots*, pages 232–237, Tsukuba, Japan, Dec. 2005.

[98] C.E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, Jul. 1948.

[99] M. Shibuya, T. Suzuki, and K. Ohnishi. Trajectory planning of biped bobot using linear pendulum mode for double support phase. *32nd Annual Conference on IEEE Industrial Electronics*, pages 4094–4099, Nov. 2006.

[100] M. Y. Shieh, K. H. Chang, C. Y. Chuang, and Y. S. Lia. Development and implementation of an artificial neural-network-based controller for gait balance of a biped robot. In *Proceedings of the 33rd Annual Conference of the IEEE Industrial Electronics Society (IECON)*, pages 2776–2782, Taipei, Taiwan, Nov. 2007.

[101] R.L. Smith. *Intelligent Motion Control With an artificial cerebellum*. PhD thesis, University of Auckland, New Zealand, 1998.

[102] R. Sutton and A. Barto. *Reinforcement learning: An Introduction*. MIT Press, 1998.

[103] Z. Tang and M.J. Er. Humanoid 3D gait generation based on inverted pendulum model. *22nd IEEE International Symposium on Intelligent Control Part of IEEE Multi-conference on Systems and Control*, pages 339–344, Oct. 2007.

[104] R.L. Tedrake. *Applied Optimal Control for Dynamically Stable Legged Locomotion*. PhD thesis, Massachusetts Institute of Technology, 2004.

[105] S.G. Tzafestas, T.E. Krikochoritis, and C.S. Tzafestas. Robust sliding-mode control of nine-link biped robot walking. *Journal of Intelligent and Robotic Systems*, 20(2-4):375–402, 1997.

[106] E.D. Vaughan. Evolution of 3-dimensional bipedal walking with hips and ankles. Master's thesis, Sussex University, 2003.

[107] S. Vijayakumar, A. D'Souza, and S. Schaal. Incremental online learning in high dimensions. *Neural Computation*, 17(12):2602–2634, 2005.

[108] S. Vijayakumar, A. D'souza, T. Shibata, J. Conradt, and S. Schaal. Statistical learning for humanoid robots. *Autonomous Robots*, 12:55–69, 2002.

[109] M. Vukobratović. *Biped Locomotion*. Springer-Verlag New York, Inc., Secaucus, NJ, 1990.

[110] M. Vukobratović. Zero-moment-point thirty five years of its life. *International Journal of Humanoid Robotics*, 1(1):157–173, 2004.

[111] M. Vukobratović, A. Frank, and D. Juricic. On the stability of biped locomotion. *IEEE Transactions on Biomedical Engineering*, 17(1):25–36, 1970.

[112] H. Wang, T.T. Lee, and W.A. Gruver. A neuromorphic controller for a three link biped robot. *IEEE Transanction on Systems, Man and Cybernetics*, 22(1):164–169, 1992.

[113] S. Wang, J. Braaksma, R. Babuška, and D. Hobbelen. Reinforcement learning control for biped robot walking on uneven surfaces. In *Proceedings of International Joint Conference on Neural Networks*, pages 4173–4178, Vancouver, BC, Canada, Jul. 2006.

[114] Christopher J.C.H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

[115] D. Whitley, S. Dominic, R. Das, and C.W. Anderson. Genetic reinforcement learning for neurocontrol problems. *Machine Learning*, 13:259–284, 1993.

[116] M. Wisse. *Essentials of dynamic walking; analysis and design of two-legged robots*. PhD thesis, Delft University of Technology, 2004.

[117] C.C. Wong, C.T. Cheng, K.H. Huang, and Y.T. Yang. Design and implementation of humanoid robot for obstacle avoidance. In *FIRA Robot World Congress*, San Francisco, USA, Jun. 2007.

[118] H. Wongsuwarn and D. Laowattana. Neuro-fuzzy algorithm for a biped robotic system. *International Journal of Applied Mathematics and Computer Sciences*, 3(4):195–201, 2006.

[119] J. Yam and W. Chow. Feedforward networks training speed enhancement by optimal initialization of the synaptic coefficients. *IEEE Transactions on Neural Networks*, 12:430–434, 2001.

[120] O.T. Yildiz and O. Dikmen. Parallel univariate decision trees. *Pattern Recognition Letters*, 28:825–832, 2007.

[121] C. Zhou. Robot learning with GA-based fuzzy reinforcement learning agents. *Information Sciences*, 145:45–68, 2002.

[122] C. Zhou and Q. Meng. Reinforcement learning with fuzzy evaluative feedback for a biped robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 3829–3834, San Francisko, CA, USA, Apr. 2000.

[123] C. Zhou and Q.C. Meng. Dynamic balance of a biped robot using fuzzy reinforcement learning agents. *Fuzzy Sets and Systems*, 134:169–187, 2003.

Shouyi Wang received a B.S. degree in Control Science and Engineering from Harbin Institute of Technology, Harbin, China, in 2003, and a MS degree in Systems and Control Engineering from Delft University of Technology, Netherlands in 2005. Currently, he is working toward the Ph.D. degree at the Department of Industrial and Systems Engineering, Rutgers, the State University of New Jersey, USA. His current research interests include data mining and pattern discovery, machine learning, intelligent decision-making systems, and multivariate time series modeling and forecasting.

Wanpracha Chaovalitwongse received a B.S. degree in Telecommunication Engineering from King Mongkut Institute of Technology Ladkrabang, Thailand, in 1999 and M.S. and Ph.D. degrees in Industrial and Systems Engineering from University of Florida in 2000 and 2003. He previously worked as a Post-Doctoral Associate in the NIH-funded Brain Dynamics Laboratory, Brain Institute and in the departments of Neuroscience and Industrial and Systems Engineering at University of Florida. He also worked for one year at the Corporate Strategic Research, ExxonMobil Research & Engineering, where he managed research in developing efficient mathematical models and novel statistical data analysis for upstream and downstream business operations. He was an Assistant Professor from 2005 to 2010 and an Associate Professor from 2010 to 2011 in the department of Industrial and Systems Engineering, Rutgers University. Currently, he is an Associate Professor in the department of Industrial & Systems Engineering and the department of Radiology at Medical Center, University of Washington.

Robert Babuška received the M.Sc. degree in control engineering from the Czech Technical University in Prague, Prague, Czech Republic, in 1990, and the Ph.D. degree from the Delft University of Technology, Delft, The Netherlands, in 1997. He was with the Department of Technical Cybernetics, Czech Technical University in Prague and with the Faculty of Electrical Engineering, Delft University of Technology. Currently, he is a Professor at the Delft Center for Systems and Control, Faculty of Mechanical Engineering, Delft University of Technology. He is involved in several projects in mechatronics, robotics, and aerospace. He is the author or coauthor of more than 190 publications, including one research monograph (Kluwer Academic), two edited books, 24 invited chapters in books, 48 journal papers, and more than 120 conference papers. His current research interests include neural and fuzzy systems for modeling and identification, fault-tolerant control, learning, and adaptive control, and dynamic multiagent systems. Prof. Babuška is the Chairman of the IFAC Technical Committee on Cognition and Control. He has served as an Associate Editor of the IEEE Transactions on Fuzzy Systems, Engineering Applications of Artificial Intelligence, and as an Area Editor of Fuzzy Sets and Systems.