World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:1, 2008

A Probabilistic Reinforcement-Based Approach to Conceptualization

Hadi Firouzi, Majid Nili Ahmadabadi, and Babak N. Araabi

Abstract—Conceptualization strengthens intelligent systems in generalization skill, effective knowledge representation, real-time inference, and managing uncertain and indefinite situations in addition to facilitating knowledge communication for learning agents situated in real world. Concept learning introduces a way of abstraction by which the continuous state is formed as entities called concepts which are connected to the action space and thus, they illustrate somehow the complex action space. Of computational concept learning approaches, action-based conceptualization is favored because of its simplicity and mirror neuron foundations in neuroscience. In this paper, a new biologically inspired concept learning approach based on the probabilistic framework is proposed. This approach exploits and extends the mirror neuron's role in conceptualization for a reinforcement learning agent in nondeterministic environments. In the proposed method, instead of building a huge numerical knowledge, the concepts are learnt gradually from rewards through interaction with the environment. Moreover the probabilistic formation of the concepts is employed to deal with uncertain and dynamic nature of real problems in addition to the ability of generalization. These characteristics as a whole distinguish the proposed learning algorithm from both a pure classification algorithm and typical reinforcement learning. Simulation results show advantages of the proposed framework in terms of convergence speed as well as generalization and asymptotic behavior because of utilizing both success and failures attempts through received rewards. Experimental results, on the other hand, show the applicability and effectiveness of the proposed method in continuous and noisy environments for a real robotic task such as maze as well as the benefits of implementing an incremental learning scenario in artificial agents.

***Keywords*—**Concept learning, Probabilistic decision making, Reinforcement Learning.

I. Introduction

INTELLIGENT creatures should be capable of abstracting their perceptual information (stimuli) to manage the overwhelming amount of data they perceive. *Conceptualization* as a tool facilitates the process of abstraction by introducing some meaningful abstract pieces of knowledge called *concepts*. From the cognitive psychological view, a concept is defined as a meta-knowledge utilized to classify things into categories where each category captures some common characteristics of the stimuli [2]. Subsequently, concept learning can be characterized as the gradual process of creating the concepts by the creature itself.

Authors are with Control and Intelligent Processing Center of Excellence (CIPCE), Electrical and Computer Engineering Department, University of Tehran, Tehran, Iran (e-mail: hfirouzi@ece.ut.ac.ir, mnili@ut.ac.ir, araabi@ut.ac.ir).

On the other hand, if the creature is required to respond properly to each stimulus, it would be of main concern to generate the optimal response for each stimulus (i.e., the optimal policy) which is the ultimate aim of the decision making approaches. Here, the conjecture is that conceptualization can play a positive role in finding and encoding the optimal policy. Biological findings corroborate this surmise especially those which focus on mirror neurons system [3]. According to these evidences, a mirror neuron has the ability to abstract a variety of the stimuli to a concept and then relate it to the proper response (action). In fact, mirror neurons classify the perceptual space based on the available actions [4]-[6].

Here, the proposed concept learning approach is presented in reinforcement learning framework which implies that the learning process is governed by the reinforcement signal issued by either the environment or the teacher. Some other models have also been proposed which all have the common theme of abstracting the continuous perceptual space through the reinforcement signal. Preliminary works include the manual abstraction and decomposition of the perceptual and action spaces [7]. Incorporating self-adaptability, Smith [8] has proposed to abstract and quantize the continuous perceptual and action spaces using two SOMs (self organizing map) and then relate them via a Q-table. Similarly, Mobahi et. al. [5], [6] have reported a successful imitative concept learning approach for phoneme acquisition problem which uses mapping functions for conceptualizing the perceptual space instead of SOM.

However, most of these models abstract the perceptual space in a deterministic manner and ignore the incompleteness latent in real world. The probabilistic formalism and especially the Bayesian framework appear to be useful in this case. The Bayesian framework recently employed in many decision making and Robotics tasks (for example, Bayesian Robot Programming framework [9]) converts the unmanageable incompleteness into the manageable uncertainty.

In this paper, we propose a new approach to partition (conceptualize) the reinforcement learning agent's perceptual space based on its available actions in order to increase the average received reward during the time. The proposed method incorporates the Bayesian formalism and representation to challenge the uncertainty of both the environment and the agent's perception. Moreover, the proposed learning algorithm is designed so that the agent can learn from its failures in addition to its successes.

The organization of the paper is as follows: in section II, the problem under consideration is described in more details. Then, it is formulated in the Probabilistic framework in section III. Section IV illustrates the proposed approach followed by the simulation and the experimental results are demonstrated in section V and section VI respectively. Finally, some conclusions are presented in section VII.

II. The Problem Statement

A reinforcement learning (RL) agent can be characterized as one which tries to learn the optimal policy from the reinforcement signal *r(t)* received for its action *a(t)* in response to stimulus vector *X(t)*. In real world systems, RL agents face some major problems. The most challenging of them is that generally the perceptual space is a multi-dimensional continuous space which inundates the agent's mind. To deal with this problem, abstraction is employed. Using abstract knowledge, the RL agents are capable of generalizing their world, defeating the high dimensional continuous perceptual space, communicating with each other in a high level fashion and speeding up their learning by a facilitated cooperation. Moreover, abstraction provides an economical cognitive architecture for RL agents.

These benefits as a whole lead us to develop a learning approach which tries to abstract the agent's perceptual space in a formal manner. To do so, conceptualization has been considered as a basic approach in this paper. More precisely, conceptualization categorizes the perceptual space into *similarity classes* (namely concepts) which unify similar state vectors as separated concepts. Indeed, concepts capture some common and hidden properties which are the true cause of similarity. Before going on the discussion, we should specify what we mean by the similarity of two states. From a functional view, two states are similar if the agent receives the maximum reward by performing similar actions in both states. This view is very close to the mirror neuron functionality in neuroscience which partitions the animals perception based on its actions [2].

To avoid unnecessary entanglement in theoretical debates between and within relevant communities, below we emphasize two broad types of relations that appear to unite events within a category [2]. 1) In *perceptual concepts*, stimuli are grouped primarily on the basis of shared physical features. 2) In *relational concepts*, it is not the physical features of stimuli but the relations among these features that are grouped. We employ relational concepts in our proposed framework in order to keep generality. As perceptual concepts are the building blocks of the relational concepts first we focus on how a perceptual concept is dealt with.

To group stimuli based on the physical feature we need a kind of similarity/dissimilarity measure to quantify the state similarity. One of the straightforward similarity measures is distance; if we assume that the perceptual space is metric, the Euclidean distance can be used to indicate the similarity of two states. Based on this assumption, if two state vectors *X* and *Y* have a short Euclidean distance from each other, we can conclude that they are similar and subsequently the agent should perform similar actions in both of the states to receive the maximum reward. This is what is called *Perceptual Concepts* in concept learning literature [2].

Although distance works well for many cases of the state similarity, there may be two similar state vectors which have a long distance from each other. This is specially the case when a concept has representatives in different locations of the perceptual space. Thus, we need a similarity measure more general than distance which in turn results in defining a more general concept type: *Relational Concepts* [2]. A relational concept groups the states which are not necessarily neighbors in perceptual space; in fact, the real cause of their similarity is something beyond the locality in the perceptual space. The way to compute the similarity in relational concepts greatly depends on the approach used for state-action value representation and the decision making method which will be explained in the following sections.

On the other hand, RL agents should handle the uncertainty existing in the environment. To do that, in this paper, the proposed method is developed based on the Probabilistic framework which enables the agent to handle the uncertainty of the environment more conveniently.

As delayed reward and multiple steps tasks are challenging in RL systems especially when the continuous RL is used, the proposed frame work is designed to deal with delayed rewards problems and can handle discounted reward case as well.

Based on issues discussed till now, the original problem can be reduced to the problem of online reinforcement-based classification and clustering of the perceptual space in order to get the similarity classes (concepts) which are in turn directly related to optimal actions. In the next two sections a Probabilistic framework to formalize the solution and a learning algorithm for this framework are presented, respectively.

III. The Probabilistic Solution

A. Modeling

In the probabilistic formalism, the suitability of the i^{th} action in the state *X* can be encoded as the probability *P(action$_i$ | X)*. As mentioned before, concepts are formed based on actions. This permits us to use them interchangeably. Thus, we can assess that:

$$\forall i \in [1..r]: \quad P(action_i \mid X) \equiv P(C_i \mid X) \tag{1}$$

where *r* is the number of concepts (actions) and *P(Ci | X)* can be interpreted as the probability with which the perceptual vector *X* belongs to the concept *Ci*. Therefore, our decision making problem can be reduced to computing the posterior probabilities *P(Ci | X)*. On the other hand, by applying the Bayes rule, we have:

$$P(C_i \mid X) = \eta . P(X \mid C_i) P(C_i) \tag{2}$$

where *P(X | Ci)* and *P(Ci)* are the likelihood of *Ci* and the prior probability of *Ci*, respectively. Thus, to compute *P(Ci |*

X), distributions *P(X | Ci)* and *P(Ci)* should be estimated which is the main focus of the next section. However, before estimation, we should first specify a parametric form for the likelihood *P(X | Ci)*. As mentioned before, a concept may have representatives in different locations of the perceptual space; that is, for the concept *Ci*, there may be more than one location (vector) in the state space where the likelihood *P(X | Ci)* is high. We call these points the modes or the modals of the probability distribution *P(X | Ci)*. As a result, *P(X | Ci)* should be modeled as a multi-modal distribution to capture the scattered nature of the concept in the perceptual space. To do so, the mixture densities model [10] is used. In other words, the likelihood of the concept *Ci* is decomposed as:

$$P(X \mid C_i) = \sum_{j=1}^{q} P(X \mid M_j) P(M_j \mid C_i) \tag{3}$$

where *Mjs* are the components of the mixture. *P(X | Mj)* conveys the probability of observing *X* if the observation is generated by the component *Mj* and *P(Mj | Ci)* is the contribution weight of the component *Mj* for the concept *Ci*. For the sake of simplicity, the components of different concepts are all unified in the set *M* whose cardinality is *q* (We simply set *P(Mj | Ci)* to zero if the component *Mj* does not belong to the concept *Ci*.) By substituting equation (3) into equation (2), we get:

$$P(C_i \mid X) = \frac{P(C_i) \sum_{j=1}^{q} P(X \mid M_j) P(M_j \mid C_i)}{\sum_{k=1}^{r} P(C_k) \sum_{j=1}^{q} P(X \mid M_j) P(M_j \mid C_k)} \tag{4}$$

According to equation (4), we can conclude that to compute *P(Ci | X)*, we should first estimate the probability distributions *P(Ci)*, *P(Mj | Ci)* and *P(X | Mj)*. In the next subsection, the method used to model these distributions is described.

B. The Parametric Forms

In this section, the parametric forms used for each of the three mentioned probability distributions are explained in details. Here the Bayesian approach is adopted to estimate the parametric forms; that is, a meta-level parametric probability distribution function (which encodes our belief) is defined for each of the parameters and the parameters of these new distributions are estimated instead.

Before assuming these probability distributions, we have to consider matrix $B = \left[b_{ij} \in \Re \right]_{r \times q}$ which b_{ij} is the non-normalized belief of belonging the modal M_j to the concept C_i. It is a fundamental matrix in our framework and in the following sections the way of updating and its relationship with other parts will be explained.

- ***P(Mj | Ci):*** This distribution can be parameterized as:

$$P(M = M_j \mid C = C_i, f_{ji}) = P(M_j \mid C_i, f_{ji}) = f_{ji} \tag{5}$$

where f_{ji} $(i \in [1..r], j \in [1..q])$ encodes our belief about the fact that M_j belongs C_i. In fact this is the normalized version of b_{ij}. There are many methods to normalize but choosing a good one is somehow tricky. This method is experimentally resulted better than others such as Boltzmann method.

$$\hat{b} = \left| \min\left(\{ b_{ij} \} \right) \right| \Big/ 1 + \left| \min\left(\{ b_{ij} \} \right) \right| - \min\left(\{ b_{ij} \} \right) \tag{6}$$
$$i \in [1..r], j \in [1..q]$$

where $\hat{b}$ is the normalization factor. As a result our belief about belonging *Mj* to *Ci* is computed based on the follow.

$$P\left(M_j \mid C_i \right) = \left(b_{ij} + \hat{b} \right) \Big/ \sum_{k=1}^{q} \left(b_{ik} + \hat{b} \right) \tag{7}$$

- ***P(Ci):*** From equation (7), the element b_{ji} of matrix *B* can be interpreted as the ratio of observed stimuli which simultaneously belong to the concept C_i and are absorbed by the component M_j. Thus, the probability *P(Ci)* can be computed directly from matrix *B*:

$$P(C_i) = \sum_{j=1}^{q} \left(b_{ji} + \hat{b} \right) \Big/ \sum_{k=1}^{r} \sum_{j=1}^{q} \left(b_{jk} + \hat{b} \right) \tag{8}$$

- ***P(X | Mj):*** Due to the fact that *P(X | Mj)* measures the proximity of *X* to the center of the component M_j in a nonlinear fashion, a symmetric unimodal distribution is suitable to model it. Among different symmetric unimodal distributions, the normal distribution is selected because of its appropriate properties. However, *P(X | Mj)* will have the normal distribution only if we know its mean and covariance matrix in advance, but this is not the case here. Thus, we should define parametric distributions for these unknown parameters to encode our belief about them. As a result, the covariance matrix and the mean vector are set to take the *Wishart* distribution and the normal distribution conditional on the covariance matrix, respectively [11]. Using these distributions, it can be proved that *P(X | Mj)* will take *multivariate t* distribution with parameters α_j, β_j, μ_j and v_j (interested readers can refer to [11] for the proof):

$$P(X \mid M_j) = t\left(X; \alpha_j - n + 1, \mu_j, \frac{v_j (\alpha_j - n + 1)}{v_j + 1} \beta_j^{-1} \right) \tag{9}$$

To explain each of the four parameters of *P(X | Mj)*, we first define S_j to be the set of the perceptual vectors based on which the distribution *P(X | Mj)* is estimated. Then the parameters are defined as follows: v_j is the cardinality of S_j; μ_j and β_j are the empirical mean and the non-normalized empirical covariance matrix of the S_j's members, respectively. Normally, α_j is set to $v_j - 1$.

Regarding to the parametric forms derived for *P(Ci)*, *P(Mj | Ci)* and *P(X | Mj)* in this section, the parameters $B = [b_{ji}]_{q \times r}$ and $D = [\alpha_j \ \mu_j \ \beta_j \ v_j]_{q \times 4}$ should be learnt from data to compute the distribution *P(Ci | X)*; moreover, the number of

the components (q) used to cluster the perceptual space should also be determined gradually from observations. In section IV, a reinforcement-based algorithm is proposed for learning the presented model from the sequential data.

IV. The Learning Algorithm

The general scenario in each time step, is that the agent perceives the perceptual vector X from the environment and based on its current estimation of $P(C_i \mid X)$ it stochastically finds the most promising concept C_g (C_{guess}) to which the vector X belongs. Performing the equivalent action of C_g (i.e. $action_g$), the agent receives the reinforcement signal r form the environment which is a signed real number. Regarding to the reinforcement signal, the learning algorithm walks through different conceptual steps: *learning from positive samples*, *learning from negative samples* and *adaptation* which are illustrated in the following subsections. Prior to consider the learning algorithm in details, the different parts of them and their relationships are roughly considered.

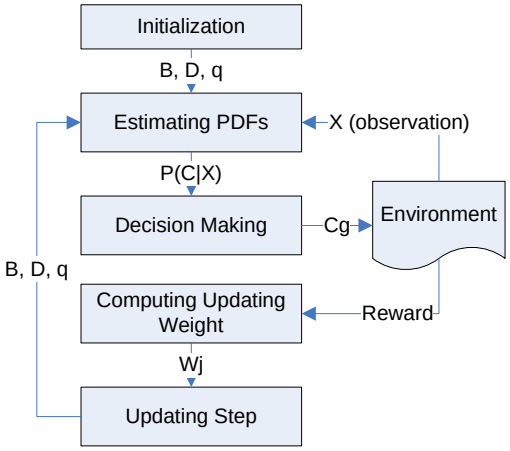

Fig 1 Structure overview of the proposed framework – different parts and their relationships

A. Initialization

In this step, the initial values for parameters q, $B=[b_{ji}]_{q\times r}$ and $D=[\alpha_j\ \mu_j\ \beta_j\ v_j]_{q\times 4}$ are set:

$r \leftarrow$ (The number of concepts)

$q \leftarrow 1$

$B \leftarrow Rand_{r\times q}$

$\forall j \in [1..q]$:

$\alpha_j \leftarrow -1, \quad \mu_j \leftarrow [0\,0\,0\,...\,0]_{1\times n}{}^T$

$v_j \leftarrow 0, \quad \beta_j \leftarrow InitialVariance \times I_n$

where, *InitialVariance* is a scalar representing the initial variance of the components, n is the perceptual space dimension, *Rand* is a random matrix, and I is the identity matrix.

B. Estimating Pdfs

Although $P(Ci)$, $P(Mj|Ci)$, $P(X|Mj)$ were considered in the modeling section, also some other *Pdfs* need to be estimated in order to employ the proposed method. These *Pdfs* are considered in details based on the following subsections.

- ***P(Mj\|X)*:** the probability of occurring component Mj by having observation X.

$$P(M_j \mid X) = \frac{P(X \mid M_j)P(M_j)}{P(X)} = \frac{P(X \mid M_j)P(M_j)}{\sum_{k=1}^{PN} P(X \mid M_k)P(M_k)} = \sigma P(X \mid M_j)P(M_j) \tag{10}$$

- ***P(Mj)*:** the probability of occurring the component Mj.

$$P(M_j) = \left(\sum_{i=1}^{r}\left(b_{ij} + \hat{b}\right)\right) \Big/ \left(\sum_{i=1}^{r}\sum_{k=1}^{q}\left(b_{ik} + \hat{b}\right)\right) \tag{11}$$

- ***P(Mj\|Ci,X)*:** the probability of occurring the component Mj by acting the action i (Ci) and receiving the observation X.

$$P(M_j \mid C_i, X) = \frac{P(X \mid M_j)P(M_j \mid C_i)}{\sum_{k=1}^{q} P(X \mid M_k)P(M_k \mid C_i)} = \sigma P(X \mid M_j)P(M_j \mid C_i) \tag{12}$$

- ***P(Mj\|~Ci,X)*:** the probability of occurring the component Mj by not acting action i (Ci) and receiving the observation X.

$$P(M_j \mid \bar{C}_i, X) = \frac{P(X \mid M_j)P(M_j \mid \bar{C}_i)}{\sum_{k=1}^{r} P(X \mid M_k)P(M_j \mid \bar{C}_k)} = \frac{P(X \mid M_j)\left[1 - P(M_j \mid C_i)\right]}{\sum_{k=1}^{r} P(X \mid M_k)\left[1 - P(M_j \mid C_k)\right]} \tag{13}$$

C. Decision Making

As mentioned before the action by which we receive a higher reward is more suitable to decide and in our probabilistic framework this suitability is encoded in $P(Ci|X)$. However we use an extra variable T (temperature) to go smoothly from random decision making to greedy one.

$$P_e(C_i \mid X) = \exp\left(P(C_i \mid X)/T\right) \Big/ \sum_{i=1}^{r} \exp\left(P(C_i \mid X)/T\right) \tag{14}$$

$$C_g = \arg soft\max_i \left\{P_e(C_i \mid X)\right\} \tag{15}$$

where Cg is the guess Concept (Action) which the agent is going to act.

D. Computing Updating Weight

Similar to Temporal Difference, the most popular member of Reinforcement Learning family, in our framework the TD error is computed and the updating weight is calculated based on the computed TD error.

Before computing the TD error let's turn our thought to

which component that observes the *X* more than others. This component is named "*most observant component*" and is computed as below.

$$M_{mac} = \arg \underset{j}{Max}\ P\left(M_j \mid C_g, X\right) \tag{16}$$

Indeed we merely notice the most observant component M_{mac} instead of all existing components in updating step. Subsequently the guess action is acted and then next most observant component M'_{mac} is also computed by receiving the next observation X'.

$$C'_g = \arg \underset{i}{Max}\ P\left(C_i \mid X'\right) \tag{17}$$

$$M'_{mac} = \arg \underset{j}{Max}\ P\left(M_j \mid C'_g, X'\right) \tag{18}$$

C'_g, M'_{mac} are the next guess action and the next most observant component respectively and finally the TD error is computed as below.

$$TD_{err} = \alpha\left(r + \gamma b_{C'_g M'_{mac}} - b_{C_g M_{mac}}\right) \tag{19}$$

where α is the learning rate, γ is the forgetting factor and r is the received reward by agent.

Due to the value of the TD error, the updating weights are computed based on the following as.

$$\begin{aligned}
&\forall j \in [1..q] \\
&if\ \left(TD_{err} > thr_{positive_sample}\right) \\
&\qquad w_j = P\left(M_j \mid C_g, X\right) \\
&elseif\ \left(TD_{err} < thr_{negative_sample}\right) \\
&\qquad w_j = P\left(M_j \mid \bar{C}_g, X\right) \\
&else \\
&\qquad w_j = P\left(M_j \mid X\right)
\end{aligned} \tag{20}$$

where $thr_{positive\_sample}$, $thr_{negative\_sample}$ are thresholds that indicate the guess action is the correct choice or is not.

E. Updating Step

In this step all parameters ($B = [b_{ji}]_{q\times r}$ and $D = [\alpha_j\ \mu_j\ \beta_j\ \mathrm{v}_j]_{q\times 4}$) are updated. First updating of B is considered.

$$\begin{aligned}
b_{gJ} &\leftarrow b_{gJ} + TD_{err} \times w_J \\
J &= \arg \max_j \{w_j\}
\end{aligned} \tag{21}$$

Only b_{gJ} relating to the maximum weight is updated that is so-called *hard updating*. It means that instead of updating all b_{gj}, just b_{gJ} is updated. This way, because of not propagating the uncertainty of the TD error along with all components, the convergence rate of the learning is increased.

As the second sub-step of updating, we have to consider matrix $D = [\alpha_j\ \mu_j\ \beta_j\ \mathrm{v}_j]_{q\times 4}$. Basically this matrix is belongs to the infrastructure of *P(X|Mj)* and defines the components in the perceptual space. Thus there are two sub-tasks when we faced with a new observation such as *X*; it can be used for updating the existing components or can be used to create a new component. A predetermined threshold is defined to indicate which sub-step should be chosen.

$$\begin{aligned}
&if\ \left(\max\left[P\left(X \mid M_j\right)\right]_{j=1}^{q} > thr_{new_component}\right) \\
&\qquad Updating\ components \\
&else \\
&\qquad Adding\ a\ new\ component
\end{aligned} \tag{22}$$

where the value of $thr_{new\_component}$ can determine the density of distributing the components in the perceptual space. In other words, if this threshold is too small most of the observations are used to update existence components and the perceptual space will be encoded by a few and big components and vice versa there will be a lot of small components which are close to each other if the threshold is huge. Therefore this threshold can specify which level of concepts might be recognized.

1) Updating Components

Similar to the updating step of the matrix B, in this case just one component is updated in order to build a more local model of perceptual space by components. Which component that maximize the *P(X|Mj)* is one that will be updated.

$$\hat{j} = \arg \max_j \left\{P\left(X \mid M_j\right)\right\} \tag{23}$$

After determining the component which should be updated we have to consider the details based on the following as:

$$\beta_{\hat{j}} = \beta_{\hat{j}} + \frac{v_{\hat{j}} w_{\hat{j}}}{v_{\hat{j}} + w_{\hat{j}}}\left(X - \mu_{\hat{j}}\right)\left(X - \mu_{\hat{j}}\right)^T \tag{24}$$

$$\mu_{\hat{j}} = \mu_{\hat{j}} + \frac{w_{\hat{j}}}{v_{\hat{j}} + w_{\hat{j}}}\left(X - \mu_{\hat{j}}\right) \tag{25}$$

$$\begin{aligned}
\alpha_{\hat{j}} &= \alpha_{\hat{j}} + w_{\hat{j}} \\
v_{\hat{j}} &= v_{\hat{j}} + w_{\hat{j}}
\end{aligned} \tag{26}$$

2) Adding a New Component

As mentioned before a new component is created if some criterion is satisfied; that is, in our algorithm, a new component is created if the likelihood *P(X | Mj)* is less than a predetermined threshold ($thr_{new\_component}$) which represents the minimum likelihood an exemplar of a component should have (There are also other criteria like ones mentioned in Adaptive Mixtures [13]).

Once a new component is created its new parameters are initialized the same as initial step expect that b_{gq} is valued based on the *TD* error where was computed before.

$$q \leftarrow q+1$$
$$\alpha_q = n, \qquad v_q = n+1 \tag{27}$$
$$\mu_q = X, \qquad \beta_q = initialVariance \times I_n$$

$$B_{:q} \leftarrow Rand_{1\times q} \tag{28}$$

$$\begin{aligned} &if\ \left(TD_{err} > thr_{positive\_sample}\right) && b_{gq} = 1 \\ &elseif\ \left(TD_{err} < thr_{negative\_sample}\right) && b_{gq} = 0 \end{aligned} \tag{29}$$

In equation (28) if *TDerr* is bigger than $thr_{positive\_sample}$ most probably the guess action (*Cg*) will be the correct choice and accordingly by initializing b_{gq} with "1", the probability of selecting *Cg* in future will increase and on the other hand, setting b_{gq} to zero will decrease the probability of being selected.

V. Simulation Results

To demonstrate the performance and the general applicability of the proposed learning algorithm in this paper, it has been applied on a maze problem and also the result has been compared with a typical Q-Learning algorithm. As a result, the proposed algorithm has achieved better average reward within a shorter time. In the simulation the following conditions are set: the perceptual space is a two dimensional space; the values of $thr_{new\_component}$ is set to 0.5 and also $thr_{positive\_sample}$ and $thr_{negative\_sample}$ are set to 5, -5 respectively; and the initial variance is set to 0.1. In this problem a simulated robot is placed in an area with specific width and length in which there are some obstacles and a goal. The robot is supposed to learn the shortest path of the goal while is avoiding the obstacles. In each decision making step, we let the robot move in one of the 8 (r = 8) predetermined directions (i.e. 0, π/4, π/2, 3π/4, π, 5π/4, 3π/2, and 7π/4) with 0.5 unit relative movement which means that the robot classifies the perceptual space into 8 distinct categories. Indeed, this correspondence comes from our action-based conceptualization view. After acting the decision the robot will receive three different rewards; if it comes into an empty space (neither goal nor obstacle) it gets -0.1 as the reward, if it faces with an obstacle, the reward is -5 and finally if it reaches the goal, it receives 20. In addition, the learning rate (α) has changed from 0.9 to 0.05, the forgetting factor has set to 0.8, and the temperature has changed from 0.2 to 0.0001. The conditions of the Q-learning simulation are exactly the same as above.

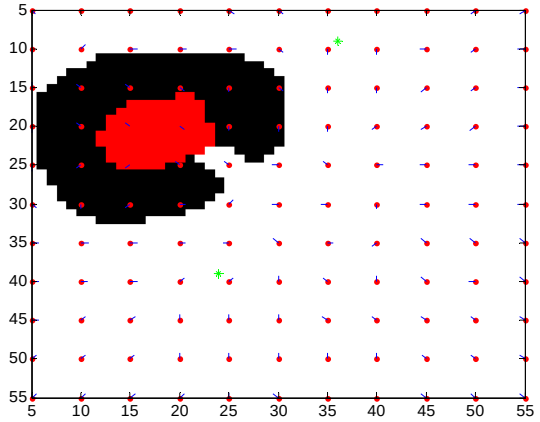

Fig. 2 The learnt optimal policy by the proposed algorithm, red area is the goal, black area is the obstacle, and white area is the empty place

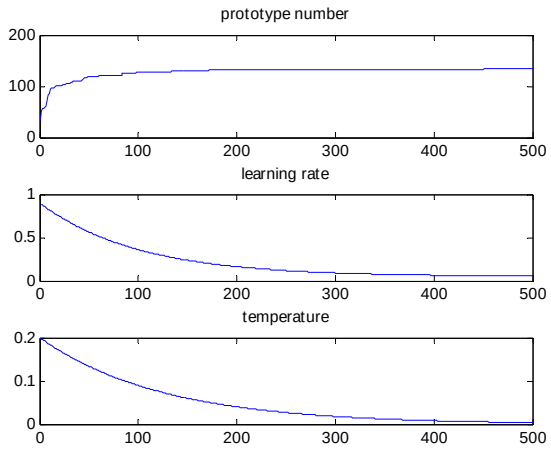

Fig. 3 The prototype number (component number), the learning rate, and the temperature during the learning

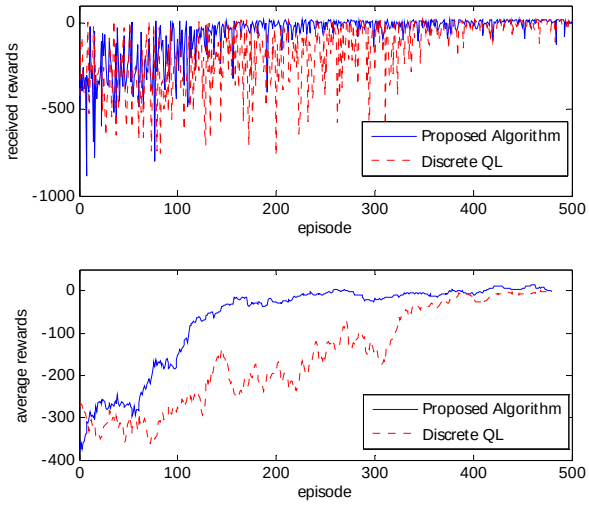

Fig. 4 Received rewards and average rewards by the proposed algorithm in compared to the Q-Learning algorithm (quantization level of the Q-Learning algorithm is 0.5)

In the simulation, powerful generalization and convergence speed of the proposed algorithm in compared to a typical Q-learning algorithm have been obviously shown. Also for

faithfully comparing, the quantization level of Q-learning algorithm has been regarded as the robot movement step (0.5) which is the most proper quantization level.

VI. Experimental Result

In this section, a real robotic task has been designed to demonstrate the applicability of using the proposed framework in the noisy environment in addition to dealing with the continuous perceptual space of the robot.

The physical robot employed in the experiments is an E-puck robot [14] (Fig. 5). This mobile robot is equipped with two stepper motors by which it can be navigated. Moreover, the robot has USB Bluetooth communication system which permits us to run the learning algorithm on our PC instead of on the limited hardware of the robot.

Similar to the simulation, the robot is supposed to learn to reach a specific area while it is avoiding the obstacles which are in the environment. To achieve this goal, the robot should distinguish different areas of the environment in order to learn and exhibit the desired behavior. In other words, in this problem, concepts (the robot's actions) are closely related to different positions of the robot in the environment. In each decision making step, we let the robot move in one of the eight ($r = 8$) predetermined directions (i.e. $-3\pi/4$, $-\pi/2$, $-\pi/4$, 0, $\pi/4$, $\pi/2$, $3\pi/4$ and π) which means that the robot classifies all of its positions into eight distinct categories. Using a camera which can capture the whole environment globally, two-dimensional position of the robot is obtained. In fact, the sole perceptual space in the experiment is a 2D continuous one as robot position which is computed by the camera.

Based on the new position which is obtained after each robot's action, an internal reinforcement signal is generated. If the robot goes to the empty area, the reward -0.1 is generated, if it contacts with the obstacles, the reward -5 is generated, and if it reaches the goal, the reward 20 is given to the robot.

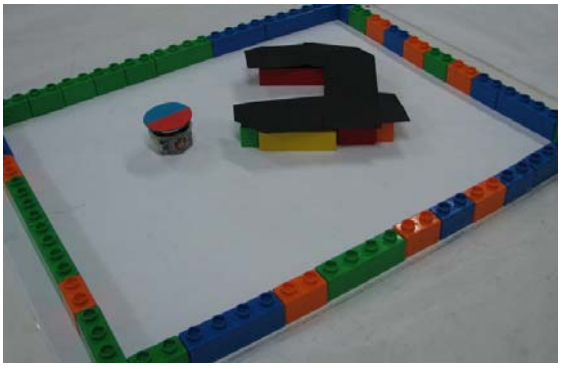

Fig. 5 The E-puck robot and the experimental environment

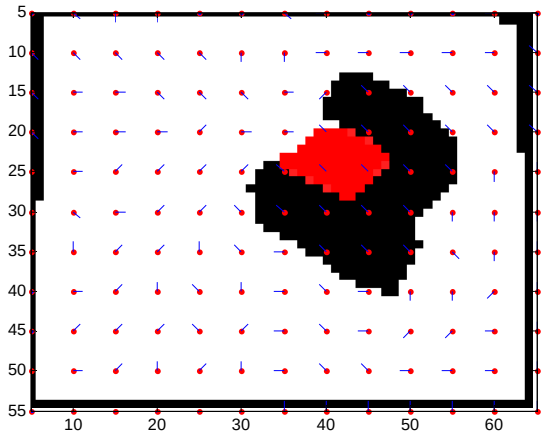

Fig. 6 The learnt optimal policy by the proposed algorithm, red area is the goal, black area is the obstacle, and white area is the empty place (model of the real environment which is used in the first step)

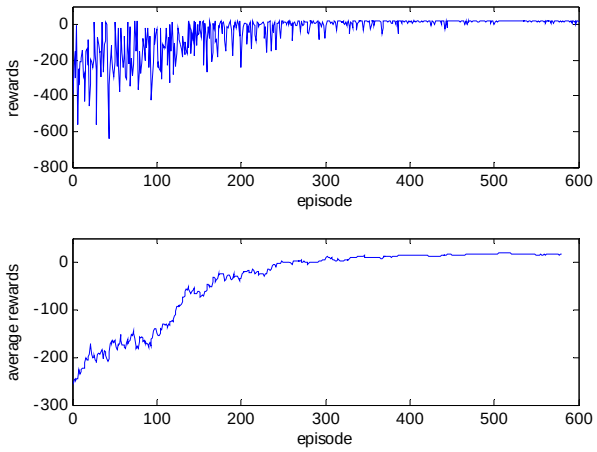

Fig. 7 Received rewards and average rewards during the learning in the first step

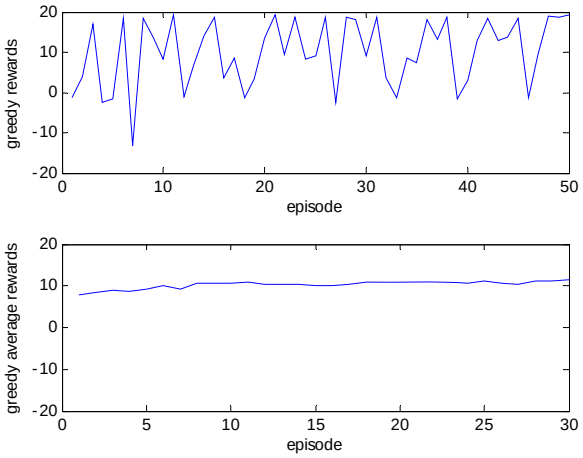

Fig. 8 Received rewards and average rewards in the second step (greedy decision making)

This experiment includes two steps. In the first one, a model of the real environment is employed to learn, see Fig. 6. In other words, the robot first learns the desired behaviors by simulation. Then in the second step, the robot uses the learnt behaviors in the real environment, see Fig. 5. In fact, in this

step the robot makes greedy decisions. Fig. 7 illustrates the received rewards in the first step during the 600 episodes and Fig. 8 demonstrates the received rewards in the second step during the 50 episodes.

The experiment reported in this section is one example of many robotic applications with continuous perceptual spaces on which the proposed framework can be applied. It is enough to define a proper reinforcement function for the problem and let the robot explore the environment and interactively learn the optimal response to each stimulus using the proposed conceptualization method. Besides, the experiment shows that the algorithm is robust in face of the environmental and perceptual noises due to its probabilistic foundation.

VII. Conclusion

In this paper, a new concept learning approach was presented to abstract the RL agent's perception and knowledge of its environment. Inspired from the mirror neurons functionality, the proposed method was simultaneously classifying and clustering the agent's perceptual space based on its available actions. Moreover, the learning algorithm was constructed based on the Probabilistic model which enabled the agent to face the uncertainty of its perception and its environment. Utilizing the mixture destinies model with adaptive number of components, the developed model was capable of learning concepts with any probability. On the other hand, the learning algorithm was designed so that it could learn through received rewards and could manage the multi step problems as well as discounted rewards. This property improved the learning process by speeding up the convergence and directing the learning curve to a higher asymptotic value. Simulation and experimental results confirmed these claims.

Future works can include extensions for handling the continuous action case as well as equipping the proposed method with babbling techniques to create concepts (actions) automatically. Additionally, some parallel mechanisms can be incorporated to determine and update existing thresholds, (e.g. $thr_{new\_component}$, $thr_{positive\_sample}$, and $thr_{negative\_sample}$) adaptively during the learning process.

Acknowledgment

This work is supported in part by Iran Telecommunication Research Center (ITRC).

References

[1] S. Amizadeh, M. N. Ahmadabadi, B. N. Araabi, R. Siegwart, A Bayesian Approach to Conceptualization Using Reinforcement Learning, IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2007.

[2] T. R. Zentall, M. Galizio, and T. S. Critchfield, "Categorization, concept Learning and behavior analysis," in *Journal of the Experimental Analysis of Behavior*, vol. 78, no. 3, pp. 237–248, November 2002.

[3] G. Buccino, S. Vogt, A. Ritzl, G. R. Fink, K. Zilles, H. J. Freund and G. Rizzolatti, "Neural circuits underlying imitation learning of hand actions: an event related fMRI study," in *Neuron*, vol. 42, pp. 323–334, April 2004.

[4] A. Billard and M. J. Mataric, "Automatic learning human arm movements by imitation: evaluation of a biologically inspired connectionist architecture," in *Robotics and Autonomous Systems*, vol. 941, pp. 1–16, 2001.

[5] K. Doya, "Reinforcement learning in continuous time and space," *Neural Computation*, vol. 12, pp. 219–245, 2000.

[6] H. Mobahi, M. Nili Ahmadabadi, and B. N. Araabi, "Concept oriented imitation towards verbal human-robot interaction," In *Proc. 2005 IEEE Int. Conf. Robotics and Automation*, pp. 1495–1500, April 2005.

[7] H. Mobahi, M. Nili Ahmadabadi, and B. N. Araabi, "A biologically inspired for conceptual imitation using reinforcement learning," to be published in *Applied Artificial Intelligence*.

[8] S. Mahadevan and J. Connell, "Automatic programming of behavior-based robots using reinforcement learning," in *Artificial Intelligence*, vol. 55, no. 2-3, pp. 311–365, June 1992.

[9] A. J. Smith, "Applications of the self-organizing map to reinforcement learning," in *Neural Networks*, vol. 15, pp. 1107–1124, 2002.

[10] O. Lebeltel, P. Bessiere, J. Diard and E. Mazer, "Bayesian robot programming," in *Autonomous Robots*, vol. 16, pp. 49–79, 2004.

[11] R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification (2nd Edition)*. New York: Wiley-Interscience, 2000.

[12] R. E. Neapolitan, *Learning Bayesian Network*. New Jersey: Pearson Prentice Hall, 2003.

[13] C. E. Priebe, "Adaptive mixtures," in *Journal of the American Statistical Association*, vol. 89, no. 427, pp. 796–806, September 1994.

[14] E-puck, EPFL Education Robot, http://www.e-puck.org.