World Academy of Science, Engineering and Technology
International Journal of Industrial and Manufacturing Engineering
Vol:3, No:5, 2009

A Cognitive Robot Collaborative Reinforcement Learning Algorithm

Amit Gil, Helman Stern, and Yael Edan

***Abstract*—**A cognitive collaborative reinforcement learning algorithm (CCRL) that incorporates an advisor into the learning process is developed to improve supervised learning. An autonomous learner is enabled with a self awareness cognitive skill to decide when to solicit instructions from the advisor. The learner can also assess the value of advice, and accept or reject it. The method is evaluated for robotic motion planning using simulation. Tests are conducted for advisors with skill levels from expert to novice. The CCRL algorithm and a combined method integrating its logic with Clouse's Introspection Approach, outperformed a base-line fully autonomous learner, and demonstrated robust performance when dealing with various advisor skill levels, learning to accept advice received from an expert, while rejecting that of less skilled collaborators. Although the CCRL algorithm is based on RL, it fits other machine learning methods, since advisor's actions are only added to the outer layer.

***Keywords*—**Robot learning, human-robot collaboration, motion planning, reinforcement learning.

I. Introduction

TO introduce robotic applications into real-world environments, robots must be constructed for a large variety of tasks and be able to adapt continuously to new and changing working conditions. Since it is impossible to model all environments and task conditions, the adaptation to new tasks cannot be achieved by regular end-user programming. Rather, the robot must be delivered with advanced capabilities to learn new tasks and new working conditions both autonomously and from its user.

A common learning approach in robotics is reinforcement learning (RL) [1], [2]. In RL the robot (agent) acts autonomously in a process guided by reinforcements from the environment, indicating how well it is performing the required task. RL is an attractive alternative for programming autonomous systems, as it allows the agent to learn behaviors on the basis of sparse, delayed reward signals [3]. Furthermore, RL does not require a detailed model of the environment or training examples, as it creates its own model and examples during the learning process.

Previous research has indicated that human-robot collaboration is essential to improve the learning and reduce the amount of time it takes a robot to accomplish a learning task (e.g., [4] – [7]). When introducing a human advisor into the RL learning process, the learning agent has access to supervised instructions, instead of relying solely on reinforcements provided by the environment. A human may aid the robot by showing it how to solve new tasks and how to improve or expand already learned behaviors.

Nevertheless, [8] indicates that past work tends to maintain a constant level of human involvement. Several methods are highly dependent on guidance, learning nothing without human interaction, while other approaches are almost entirely exploration based, using limited input from a teacher. Furthermore, many of the previous work assumes that human assistance is available at all times. Indeed, human intervention can improve the learning process and accelerate the robot's learning, but if it is required too frequently, the autonomous sense of the learning will be lost along with the initial purpose of a robot replacing the human. Hence, a central issue in human-robot collaboration, addressed in this research, is the determination of *whether* and *when* human intervention is required. The main challenge is to design a mechanism for deciding whether and when the learner should ask for advice. The goal here is to maximize the impact of the advisor's instruction, so that the learner develops its decision policy quickly and correctly, with as little training as possible [9].

Another deficiency in prior works is the assumption that the human advisor is an expert providing only optimal advice. This might not be the case when the instructor is tired for example, or when the human is a novice or unfamiliar with the case (e.g., if it is a child instructing a service robot performing daily household chores). Hence, the assumption of an expert advisor is relaxed in this research, so that non-expert instructors are also considered.

This paper presents a Cognitive Collaborative Reinforcement Learning algorithm (CCRL) that addresses the questions of *whether* and *when* the robot should solicit advice by endowing the robot with human-like cognitive abilities. The robot applies a result-oriented approach, seeking aid when it comes to the understanding that its performance is not sufficient. Furthermore, the robot is able to judge the worth of the advice it receives. This self-awareness is achieved by performing self tests designed to evaluate its learning performance according to acceptable performance thresholds. Compared to the concept of adjustable autonomy [10], our system provides "*shared learning*" with only two modes of operation: supervised and autonomous.

The CCRL algorithm uses the basic model of a RL learner incorporating on-line advisor-suggested actions or guidance. Upon receipt of an action from the advisor, the robot executes

The authors are with the Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel (phone: +972-8-6461434; fax: +972-8-6472958; e-mail: gilami@bgu.ac.il; helman@bgu.ac.il; yael@bgu.ac.il).

the action as if it had chosen the action with its own policy [9]. Thus, the basic RL algorithm used does not need to be modified to handle the advisors actions. The adjustable autonomy method includes two learning modes, supervised and autonomous, following the model introduced in [6].

CCRL is evaluated using a simulated 3D environment for a mobile robot motion planning task. In this task we focused on thorough statistical assessment of the performance and compared the CCRL algorithm to autonomous learning, to the *Introspection Approach* (IA) [9] and to a combined method that integrates CCRL and IA. A simulated adviser with various skill levels is used in the evaluations.

The paper is organized as follows: Section II presents the new CCRL algorithm. Section III describes the 3D motion planning task and introduces the representation of advisor skill levels and the IA method (used for comparison). Conclusions are provided in section IV.

II. The CCRL Algorithm

In the CCRL model the robot is endowed with two cognitive abilities to assess: (a) its performance and request advice when it is not sufficient, and (b) the value of the offered advice and decide whether to continue asking for it or stop the requests and revert to fully autonomous learning.

A. Collaborative Learning

Consider a collaborative learning model in which the system can be in one of two modes: (a) autonomous (unsupervised learning) and (b) guided (supervised by an outside intelligent agent). In the autonomous mode the robot decides which actions to take according to feedback from the environment (reinforcements), using a certain action selection method. The collaborative feature is added in this mode so that the learner can switch into the guided supervised mode and back. In the guided supervised mode an agent such as a human advisor suggests actions. This knowledge is incorporated into the learning function if it is deemed worthy. The learning itself can be done using any RL algorithm (e.g., SARSA, *Q*-learning). The advice from an outside guidance agent is unnecessary as long as the robot learns policies and adapts to new states while showing improvement. Only when the robot senses its performance is not improving at the desired rate, is the advisor solicited to intervene and suggest actions. The robot then performs the suggested action, and updates its *Q* values according to the action taken as though it had chosen the action itself.

The robot is endowed with two cognitive capabilities that allow it to decide *whether* and *when* to switch between the autonomous and guided modes. These decisions are triggered when two performance thresholds are exceeded: Λ, used to determine when to ask for advice, and Ω, used to determine whether the advice is acceptable or not. These self tests are based on the ability of the robot to assess its own learning performance.

B. Self-Performance Assessment Capability

The robot must determine whether its performance is sufficient in order to decide when to switch between the two learning modes. Since the optimal solution is unknown a priori, the threshold for triggering a request for advice cannot be set as a constant measure, above which advisor assistance will be desired. The robot can sense it is not learning fast enough by comparing its current performance with past performance. The robot wishes to achieve a certain improvement rate during the learning session. When it does not achieve that rate, a request for advice is triggered. The improvement rate is defined as a ratio between moving averages of the number of steps of previous episodes, as shown in (1).

$$IR = \frac{T_p - T_c}{T_p}$$
$$T_c = \frac{\sum_{i=n-K}^{n-1}(T_i)}{K};\quad T_p = \frac{\sum_{i=n-2K}^{n-K+1}(T_i)}{K} \tag{1}$$

Here the performance of an episode ***i***, denoted as T_i, is the number of steps to reach a goal state in a problem in which the objective is to minimize the number of steps. n is the current episode and IR is the actual performance improvement rate, comparing the previous average number of steps T_p (average over previous K episodes, n-$2K$ to n-K+1) and the current average T_c (average over the most recent K episodes, n-K to n-1). If the current average is smaller than the previous one (less steps required to reach the goal – better performance) IR will be positive.

C. Advise Request Test

The CCRL *advice request* self-test compares IR with the threshold Λ, as shown in (2).

If $IR <$ Λ, then *request advice*
Else *learn autonomously* (2)

Here Λ is a predefined collaboration threshold, representing the desired improvement in performance. Before each learning episode begins, the actual improvement rate is compared to the threshold. If IR is greater than Λ, the robot will continue to learn autonomously and will not solicit advice (this implies that the actual rate is better than the desired). If IR is less than Λ, the improvement rate is not sufficient, and advisor assistance will be requested. When requested, the advisor will assist during the entire episode.

When the robot converges to the optimum, obviously there will be no improvement in the performance, and advisor assistance will be asked recurrently without need. This problem is solved by applying the following definition and rule for convergence: If after 2K episodes the robot produces the same result, it assumes it has reached the optimum and stops asking for aid. Even if the optimum found was a local one, if $2K$ episodes using human assistance did not help the

robot escape it, then there is no sense in continuing the requests.

D. Advice Assessment Capability

Until here the assumption was that the advisor provides good instructions, but what happens if the advice is bad? Wrong advice will not promote learning, and might even cause deterioration in performance. By endowing the robot with the capability to assess the value of the advice, such situations may be avoided. The robot judges the advisor's suggestions by comparing its performance when using the advisor's aid with past performance. If assistance does not improve performance, the robot learns to stop asking for it. The number of steps achieved at episodes performed with advisor assistance is compared to the average number of steps over the K episodes previous to the assisted episodes. In the assisted episodes, when the number of steps to reach the goal, T_a, is higher (worse) than the average this implies that advisor instructions are worthless and possibly even misleading. The number of times in which the episode with advisor assistance produced worse results than the average is denoted as ML, and updated as shown in (3).

$$\text{If } T_a(n) > \frac{\sum_{i=n-K}^{n-1} (T_i)}{K}, \text{ then } ML = ML + 1 \tag{3}$$

E. Advice Rejection Test

When ML exceeds a predefined threshold (which occurs when the human misleads the robot too many times), the robot refuses the advice, and switches to a fully autonomous learning mode until the end of the session. The advice rejection test is elaborated in (4).

$$\begin{aligned} &\text{If } ML > \Omega, \text{ then } \textit{refuse advice} \\ &\text{Else } \textit{continue requesting advice when } IR < \Lambda \end{aligned} \tag{4}$$

Here ML is the number of occasions in which the human misled the robot causing the episode with advisor assistance to achieve worse results than the average results of the K previous episodes, and Ω is a predefined threshold for such occasions, above which collaboration is stopped.

When the human has poor expertise, the episodes performed with his assistance will result in decreased performance, ML will rapidly rise and exceed Ω, and the robot will stop asking for advisor aid, as it should. In this final structure of the algorithm, collaboration is defined by the two threshold parameters, Λ and Ω, determining the desired improvement rate and the acceptable number of human misleads, respectively.

III. Simulated Motion Planning Task

The CCRL algorithm is evaluated by applying it to a simulated three-dimensional mobile robot motion planning task. The following cases are compared: (i) fully autonomous learning (a base-line used for comparison), (ii) learning using the Introspection Approach (IA), and (iii) learning with a combined CCRL and IA method. A simulated adviser with various skill levels is used in the evaluations. Advisor skill levels are represented by *softmax* temperature values varying the suggested actions from optimal to random.

A. Introspection Approach (IA)

The Introspection Approach (IA) [9] is a method by which the learning agent determines when it requires aid, and is used here as a benchmark for comparison. In IA the agent asks for instruction when it is confused or unable to decide upon a course of action. Guidance received via IA is shown to be more informative than random guidance, thus making better use of the training agent [9].

IA is implemented using a test developed to determine whether the learner is unsure of its choices, indicating the need for help in novel situations. When discussing an automated learner, it is fairly easy to specify exactly when they are unsure: one has access to the decision policy and the evaluations on which the decision is based. The test examines the two extreme values of the value function $Q(s,a)$. If these values sufficiently close to each other it implies that the learner has not experienced this state often enough to produce a clear choice. In this case the test succeeds and the learner asks for aid. Sufficiency is determined by comparing the difference between the minimum and maximum Q values to a width parameter Ψ - if the difference is smaller than the width parameter the test succeeds. With a small width parameter, the learner rarely asks for assistance, while with a large width parameter, the learner asks for aid quite frequently. The IA advice request self test is shown in (5).

$$\begin{aligned} &\text{If } \mathrm{Max}_i\, Q(s,a_i) - \mathrm{Min}_i\, Q(s,a_i) < \Psi \\ &\text{Then } \textit{request advice for current state } s \\ &\text{Else } \textit{choose action autonomously} \end{aligned} \tag{5}$$

Where $Q(s,a_i)$ is the value of taking action a_i when at state s, and Ψ is the width parameter.

B. Representation of Advisor Skill Levels

Since we assume that perfect guidance cannot always be provided, we analyze the effect of various skill levels of the human advisor, by considering a continuum from novice to expert. When asked to give advice, the advisor examines the current state of the learner and provides the action that it considers best. In case of an expert advisor, this action is optimal. Lesser skilled advisors may provide either optimal or suboptimal actions. By adjusting the frequency by which the advisor responds with suboptimal actions, a wide range of problem-solving expertise can be simulated, from an expert advisor with perfect knowledge and skills to a novice with poor skills.

The skill level of the advisor is represented by the *softmax* action selection rule [2], based on an optimal Q table. The optimal Q table is assumed to be known for the advisor simulation, but is of course unknown to the learning agent. In *softmax* the action probabilities are varied as a graded function of the estimated value. The greedy action is given the highest

selection probability, but all the others are ranked and weighted according to their value estimates. The *softmax* method uses a Boltzmann distribution, choosing action a on the *t*-th step with the probability shown in (6).

$$\text{Prob}(a) = \frac{e^{Q_t(a)/\tau}}{\sum_{i=1}^{n} e^{Q_t(i)/\tau}} \quad (6)$$

Where $Q_t(i)$ is the value of taking an action *i* from the current (*t*-th step) state, and τ is a positive parameter referred to as the *temperature*. High temperatures cause the actions to be all (nearly) equiprobable. Low temperatures cause a greater difference in selection probability for actions that differ in their value estimates. In the limit as $\tau \to 0$, *softmax* action selection becomes the same as greedy action selection.

Human skill level is adjusted using τ, the temperature parameter. The advisor's action selection is performed on the basis of an optimal *Q* table. Using very small temperature values implies choosing actions greedily resulting in suggesting optimal actions at each state. Using higher temperature values will result in more random action suggestions. Therefore, a human with perfect skills can be represented by using a very low temperature while a human with poor skills will be represented using a relatively high temperature. Fig. 1 presents an example for the performance of advisors with various skill levels, represented using a range of temperatures, τ = 0.01 - 1

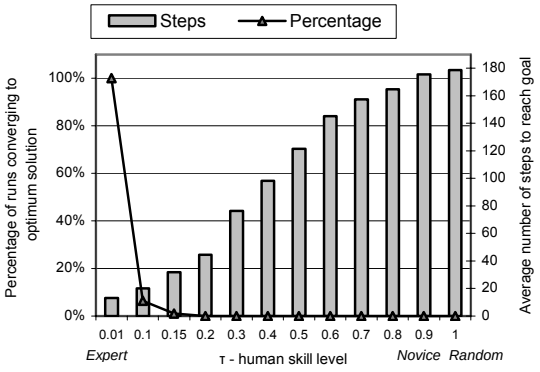

Fig. 1 Human advisor representation

The most skillful advisor, represented using τ = 0.01, chooses actions that result in reaching the goal in the optimal number of steps (13 steps in this example) in 100% of the cases, while less skilled advisors achieve optimal solutions with lower percentages and a higher average number of steps. The poorest skilled advisor (a novice) is represented by level 0.9. A temperature value of 1 represents a random selection of actions. Here we assume that the most inexperienced human will not select actions randomly; and hence a temperature value of 1 is an unrealistic representation of a human skill. Nevertheless, this value was included to provide a boundary point for the analysis and as shown in the graph such random action selections do not achieve the optimal solution at all and requires an average of 178 steps to reach the goal state. It is important to understand that here there is no learning process of the advisor, but only a use of the optimal *Q* values to simulate the human suggestions to the robot.

C. Fully Autonomous Learning (Base-Line)

The base-line robotic learner employs *Q*-learning to develop its policy. A value *Q*, associated with a state-action pair, (s_t,a_t), represents how "good" it is to perform a specific action a_t when at state s_t. A learning episode is defined as a finite sequence of time steps, during which the agent traverses from the starting state to the goal state. A learning session is a series of learning episodes (each of length N). Action selection uses the softmax equation (6) with an adaptive temperature value, T, (Boltzman exploration with decreasing temperature parameter). The temperature T is varied as a function of the learning episode according to (7).

$$T = \frac{1}{n^{\beta}}, \quad (7)$$

Where *n* is the number of episodes already performed during the current learning session, and β is a positive decay parameter specifying how fast T will exponentially decrease towards zero (a larger β value increases the chances of earlier exploitation). In the beginning of the session, when the agent has not gathered much information, a relatively high temperature encourages exploration of the state-space by allowing more non-greedy actions. As the learning session progresses, the temperature decreases, reducing the number of random actions, and allowing the agent to exploit the information already gathered and select actions that perform better.

D. Task Definition

Evaluation is conducted for a mobile robot motion planning problem in a three-dimensional grid-world of size 10×10×10 (1000 states). Two grid-world instances are considered, world I with a relatively low obstacle density (60 obstacle states), and world II with a higher density (100 obstacle states).

Fig. 2 shows one of the two grid-worlds. The starting and goal states are the light-gray cubes. The obstacles are shown as dark-gray cubes. The optimal (shortest) path from the starting state to the goal state includes 13 and 16 steps for worlds I and II, respectively.

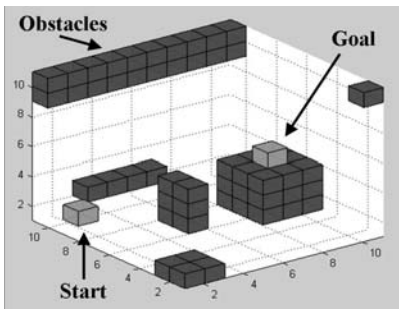

Fig. 2 Grid-world I

Also, cases of limited and full viewability are evaluated. The limited view case may arise in systems employing teleoperated guidance, when the human operator is located remotely, receiving visual feedback from the robot's operation area through a camera with limited area coverage.

In the limited case, the human can supply advice only when the robot is within the field of view. This field of view is defined for each of the grid-worlds.

The objective of the robot is to traverse from a starting state to a goal state through the shortest path, while avoiding obstacles. At each state (s_t), the robot can choose one of six actions (a_t) - up, down, left, right, forward, or backward. When the robot collides with an obstacle, reaches the goal or exceeds a maximum number of steps, the learning episode is stopped, and the robot is returned to the starting point. Reinforcements (r_t) are set as follows: the robot receives a positive reward of +1.5 units for reaching the goal, a negative reward of -1.0 units for colliding with an obstacle and a negative reward of -0.1 units for each step performed. The state of the system is the position of the robot defined by its three coordinate values.

E. Evaluation

Four different learning methods were employed and compared in simulations implemented in MATLAB: (i) fully-autonomous learning using a standard *Q*-learning algorithm (serving as base-line for the comparison), (ii) the CCRL algorithm, (iii) the IA method, and (iv) a combined method integrating the advice request rules of both CCRL and IA. All methods were evaluated using four environments – worlds I and II, each with full and limited views.

All of the learning sessions included 200 learning episodes, with a maximum of 200 steps allowed at each episode. For the collaborative algorithms, human skills, represented by τ, were varied from an expert advisor ($\tau = 0.01$) to a novice ($\tau = 1$).

The *first* simulation examined the base-line fully autonomous learning performance. Determination of the best decay factor β for the adaptive softmax temperature T was determined by a series of runs.

A *second* simulation is conducted to evaluate the performance of the CCRL algorithm, in which the same best value of β is used for the learner. The advisor's collaboration threshold Λ is varied from 0.01 (demanding small improvement in performance) to 0.9 (demanding significant improvement during the session). The threshold Ω, defining the number of occasions in which the results of learning with an advisor's aid are allowed to be worse than the results of the previous episodes, is varied from 1 to 7. For each combination of τ (the advisor's skill level), Λ and Ω, five simulation replications of 100 learning sessions each were performed. In all of the experiments the parameter K used for various calculations was set to 5.

In a *third* simulation, IA was implemented to solve the navigation problem under consistent assumptions. The width parameter Ψ is varied from 0.1 (representing a learner which is rarely uncertain) to 1.5 (representing a learner that asks for aid quite frequently). Each value of Ψ is evaluated by performing five simulation replications, each containing 100 learning sessions.

A *fourth* simulation evaluated the performance of a combined method, in a combined algorithm integrating both CCRL and IA. In this algorithm, advice is solicited only when both advice request tests (2) and (5) are passed, meaning the learner's rate of improvement is unsatisfactory and the learner is unsure of its choices in its current state. Again, for each combination of τ, Λ, Ω and Ψ five replications of 100 sessions were performed.

Performance was evaluated with the following measures: (i) *AR* (advice requests) - average number of requests for advice during the learning session (used only for CCRL, in which advice is requested for a whole episode), (ii) *HS* (helped steps) – average number of steps performed using advice during the learning session (used for the IA method in which advice is requested per step, and for the comparison), (iii) *SP* (success percentage) – average percentage of learning sessions reaching the optimal solution (minimal path length) and (iv) *Score* – weighted normalized scoring based on the *HS* and *SP* measures (described in a following section).

F. Results and Discussion

Autonomous learning performance was examined for various values (0.5 to 1.1) of the Boltzman temperature decay parameter β. The β values that achieved the best results were 1.3 and 0.7, for worlds I and II, respectively. The fully autonomous learning achieved *SP* values (average percent of sessions converging to optimum) of 63 and 37 percent for worlds I and II, respectively. The reason for the lower success percentage in world II lies in the fact that it has a higher obstacle density, and hence it is harder to reach the goal. When applying CCRL the same optimal β values were used for the robot learner. With an expert advisor ($\tau = 0.01$), the results improve drastically as expected. The learning achieves 99 and 96 percent SP for worlds I and II (for th4e full view case), respectively. This translates to an improvement of 36 and 59 percent from the autonomous learning results.

When setting low values of Λ, the robot is expected to achieve less improvement in each episode, hence requests less advice. As Λ rises, high improvement is required and the robot asks for help more frequently. Figures 3 and 4 show the results for the *limited view case* of world I, with and without advice assessment capabilities, respectively. The collaboration threshold is $\Lambda = 0.05$ and advisors of various skill levels are represented by τ.

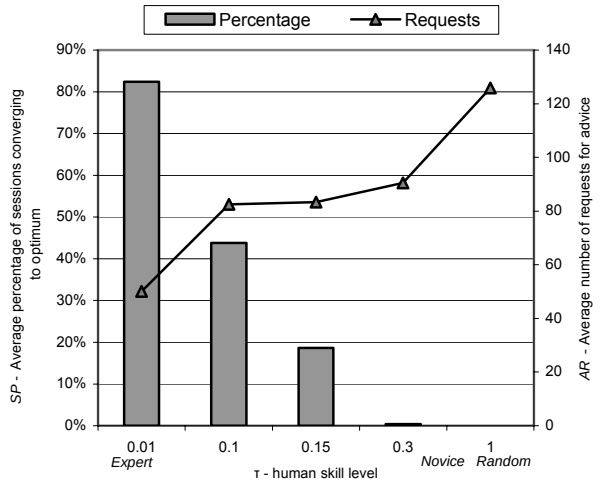

Fig. 3 CCRL: Collaborative learning without advice assessment

On the one hand, the learning agent, sensing its performance is not sufficient, asks for human aid. On the other hand, when the human is not an expert, the advice may not bring the desired improvement in performance, and even cause deterioration. The learning agent keeps asking for help because it does not improve, and the help deteriorates its performance. The situation enters a "vicious cycle" from which there is no escape, resulting in very low performance. As expected, when human skill level is low, the agent requests more help, and the performance deteriorates rapidly. This is especially seen when advice is random.

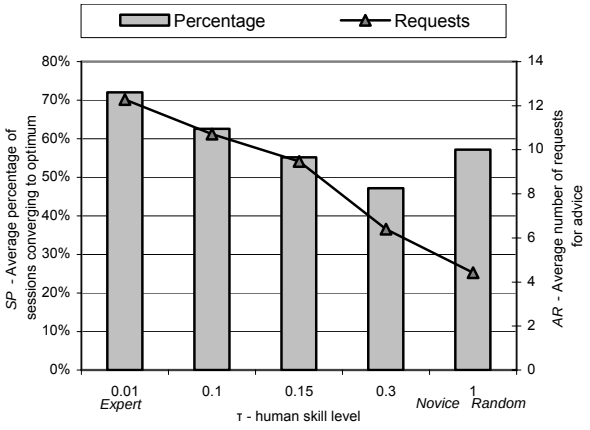

Fig. 4 CCRL: Collaborative learning with advice assessment

Fig. 4 shows results for the same environment (world I, limited view) with a collaboration threshold $\Lambda = 0.05$ and an advice assessment capability threshold $\Omega = 1$, again using the aid of advisors with various skill levels. The introduction of advice assessment capabilities helps break this "vicious cycle" and improves performance. This is true with the exception of the case of the expert where results are slightly worse in comparison to those achieved without the advice assessment capability (72% vs. 82%). The reason is that the agent sometimes misjudges the advice and rejects it. Even though the advisor is considered an expert, with low probability some actions are non optimal. In these cases the actions chosen autonomously by the agent (who reverted to fully autonomous learning) are worse than those suggested by the advisor, leading to worse performance. The increase of the SP on the right of the graph in Fig. 4 deserves explanation. This is the case when random actions are proposed, which are quickly detected by the robot, who then stops asking for advice early in the session. The result is a 53 percent success rate slightly below the autonomous value of 63 percent. The worse performance appears when the advisor's expertise is midway ($\tau = 0.3$), not very good and not very bad, so it takes time for the robot learner to notice that the advice is not good enough and to stop asking for it.

When setting Ω there is a trade-off: when Ω is low, poor skilled advisors would be quickly recognized and discarded, but experts might be misjudged and unjustly discarded as well. When Ω is high there is a smaller chance of discarding an expert, but it also takes longer time for the robot to identify worse skilled advisors, and the prolonged use of their advices obstructs the learning. Hence, when the advisor is skilled a use of high values of Ω achieves the best performance, while when the advisor has limited or no skills lower values of Ω result in better learning.

When employing IA, the width parameter influences the learning as previously described - with a small width, the learner is rarely uncertain, asking little advice, while with a large width, the learner asks for aid quite frequently. When introducing less competent advisors, IA suffers from the same problem described for CCRL – bad advice leaves the robot uncertain, leading it to ask for more advice, making it even more uncertain, and so on. The difference from CCRL is that IA does not endow the robot with the advice assessment capability that enables the robot to cope with such advisors.

G. Comparative Analysis

When comparing the methods it is important to notice that unlike CCRL, in IA assistance is triggered per step and not for the entire episode. Therefore, the performance measure used for comparison is *HS*, the number of steps in the entire session performed using advisor assistance.

The comparison is problematic since we have a multi-objective problem with two performance measures, *SP* (success percentage) and *HS* (helped steps). The preferable case, higher performance with the cost of many interruptions to the advisor, or less interruption with inferior performance, depends on the specific application. A way to address this difficulty is to base the comparison on a weighted normalized scoring. The two performance measures receive a weight corresponding to their relative importance. The results of a specific combination of collaboration parameters (Λ, Ω and Ψ) are normalized and combined to provide standard score for comparable analysis. The combined score is calculated according to (8).

$$Score = W_1 * \frac{SP_i - \min(SP)}{\max(SP) - \min(SP)} + W_2 * \frac{\max(HS) - HS_i}{\max(HS) - \min(HS)} \quad (8)$$
$$W_1 + W_2 = 1$$

Where W_1 and W_2 are the weights assigned to *SP* and *HS*, respectively. SP_i is the average percentage achieved using the *i*-th combination of collaboration parameters and HS_i is the average number of helping steps used with that combination. The calculation is designed in a way that will result in the highest score of 1 when the evaluated combination achieves the highest *SP*, using the lowest *HS*. Lower SP_i or higher HS_i will reduce the score.

When comparing, one can seek the combination receiving the highest score for a specific advisor skill level, or the combination that shows the most robust performance, dealing well with all levels of human expertise (here the final score for a specific combination is an average of the scores received for the various human skill levels).

Tables I and II show the best scores achieved by each of the three methods (CCRL, IA and combined), for the full and limited view cases of world I.

TABLE I
SCORES FOR FULL VIEW CASE

Advisor (τ)	Method	Parameters	*SP*	*HS*	*Score*
Expert (0.01)	CRL	Λ=0.05, Ω=1	99%	292.2	0.96
	IA	Ψ=0.7	98%	268.6	0.96
	Combined	Λ=0.3, Ω=1, Ψ=1	100%	96.8	1.00
Moderately expert (0. 1)	CRL	Λ=0.3, Ω=1	67%	825.2	0.72
	IA	Ψ=1	99%	200.5	0.98
	Combined	Λ=0.3, Ω=1, Ψ=1	98%	156.2	0.98
Limited skills (0.3)	CRL	Λ=0.9, Ω=1	11%	593.6	0.47
	IA	Ψ=1.3	96%	747.9	0.87
	Combined	Λ=0.05,Ω=1, Ψ=1	80%	623.7	0.81
Random (1.0)	CRL	Λ=0.9, Ω=1	55%	271.4	0.74
	IA	Ψ=0.1	62%	915.1	0.68
	Combined	Λ=0.05,Ω=1, Ψ=0.3	61%	308.8	0.77
All levels (average)	CRL	Λ=0.3, Ω=1	55%	595.4	0.69
	IA	Ψ=0.7	84%	988.9	0.77
	Combined	Λ=0.05,Ω=1, Ψ=1	82%	290.9	0.88

TABLE II
SCORES FOR LIMITED VIEW CASE

Advisor (τ)	Method	Parameters	*SP*	*HS*	*Score*
Expert (0.01)	CRL	Λ=0.05, Ω=7	80%	135.3	0.85
	IA	Ψ=1.3	82%	153.9	0.86
	Combined	Λ=0.05,Ω=7,Ψ=0.3	73%	51.5	0.84
Moderately expert (0. 1)	CRL	Λ=0.2, Ω=1	66%	25.5	0.78
	IA	Ψ=1.3	80%	171.9	0.82
	Combined	Λ=0.05,Ω=1,Ψ=0.3	67%	15.9	0.8
Limited skills (0.3)	CRL	Λ=0.05, Ω=1	59%	39.9	0.69
	IA	Ψ=0.1	60%	136.2	0.62
	Combined	Λ=0.5,Ω=1,Ψ=0.3	60%	19.2	0.72
Random (1.0)	CRL	Λ=0.01, Ω=1	63%	28.8	0.74
	IA	Ψ=0.1	63%	149.6	0.63
	Combined	Λ=0.5,Ω=1,Ψ=0.3	61%	18.9	0.73
All levels (average)	CRL	Λ=0.3, Ω=1	62%	27.9	0.73
	IA	Ψ=0.1	63%	121.4	0.66
	Combined	Λ=0.05,Ω=1,Ψ=0.3	63%	17.4	0.74

Equal weights are assigned to each performance measure ($W_1 = W_2 = 0.5$). The scores are compared and ranked using one-way ANOVA (F-test) and Tukey's HSD test, demanding 95% confidence level. Significant best scores are marked in gray. When two scores are marked it means they are significantly better than the third, but there is no significance between them. When none is marked it means there is no significant difference.

Overall, the combined method achieves the best results for both worlds and both view cases. It does so for most of the advisor skill levels separately, and for the average case, demonstrating robustness in dealing with various levels of advisors. These results are achieved since the robot asks for aid only when it really requires it, under conditions of uncertainty and deficiency in performance, and stops asking for it when it is not beneficial.

When considering CCRL and IA, IA performs better when assisted by skillful advisors (τ = 0.01, 0.1) and in average in the full view cases, while CCRL achieves better results (equivalent to those of the combined method) with lesser skilled advisors (τ = 0.3, 1) and in average in the limited cases. This can be attributed to the advice assessment capability employed in CCRL, allowing it to deal better with less competent advisors.

IV. CONCLUSION

A cognitive collaborative reinforcement learning algorithm (CCRL) that incorporates an advisor into the learning process is developed to improve supervised learning. The CCRL algorithm allows a RL learner to intelligently decide whether and when to solicit advice from an advisor, by endowing it with the capabilities to evaluate its performance and to assess the value of the advice. When assisted by highly skilled advisors the agent learns to use them frequently to improve its performance. When dealing with less skilled advisors it learns to discard bad advice and switch to autonomous learning. Tests were made on several instances of a mobile robotic motion planning problem. The CCRL algorithm and especially the combined method (CCRL with IA) achieved better results than the base-line fully autonomous learner and the learner employing IA in many learning scenarios, proving the expediency of the endowed cognitive capabilities. Furthermore, this paper also suggests a new method for representing various advisor skill levels, allowing the evaluation of collaboration algorithms under realistic conditions of imperfect guidance. Although there are other robotic machine learning algorithms, we selected RL as a convenient platform to demonstrate the advantages of using outside advice to speed the learning process. The incorporation of the CCRL algorithm into other machine learning methods is left for future work. In addition for tasks outside of the grid-world, the optimal solution can be defined by any objective or reward measure in lieu of the number of steps taken.

ACKNOWLEDGEMENT

This research was partially supported by the Paul Ivanier Center for Robotics Research and Production Management,

and by the Rabbi W. Gunther Plaut Chair in Manufacturing Engineering, Ben-Gurion University of the Negev.

REFERENCES

[1] C. J. C. H. Watkins, "Learning from Delayed Rewards," *Ph.D. dissertation, Psychology Dept.*, Cambridge University, 1989.
[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, Cambridge, MA: MIT Press, 1998.
[3] T. G. Dietterich, "Hierarchical reinforcement learning with the maxq value function decomposition," *Journal of Artificial Intelligence Research*, 1999, vol. 13, pp. 227-303.
[4] V. N. Papudesi and M. Huber, "Learning from reinforcement and advice using composite reward functions," *in Proc. 16th Int. FLAIRS Conf.*, pp. 361-365, St. Augustine, FL, 2003.
[5] L. Mihalkova and R. Mooney, "Using active relocation to aid reinforcement," *in Proc. 19th Int. FLAIRS Conf.*, Florida, 2006,
[6] U. Kartoun, H. Stern, and Y. Edan, "Human-robot collaborative learning system for inspection," *IEEE Int. Conf. on Systems, Man, and Cybernetics*, pp. 4249-4255, Taipei, Taiwan, 2006.
[7] V. U. Cetina, "Supervised Reinforcement Learning Using Behavior Models," *IEEE Computer Society 6th Int. Conf. on Machine Learning and Applications, Cincinnati*, Ohio, USA, 2007.
[8] C. Breazeal and A, Thomaz, "Learning from Human Teachers with Socially Guided Exploration," *IEEE Int. Conf. on Robotics and Automation*, Pasadena, CA, USA, 2008.
[9] J. A. Clouse, "An introspection approach to querying a trainer," *technical report 96-13*, University of Massachusetts, Amherst, MA, 1996.
[10] M. A. Goodrich, R. D. R. Olsen, J. W. Crandall and T. J. Palmer, Experiments in adjustable autonomy," in Proceedings of the IJCAI Workshop on Autonomy, Delegation and Control: Interacting with Intelligent Agents, 2001, pp. 1624-1629.