Co-evolutionary Perception-based Reinforcement Learning For Sensor Allocation in Autonomous Vehicles

Hamid R. Berenji, David Vengerov, Jayesh Ametha
Intelligent Inference Systems Corp.
333 West Maude Ave., Suite 105
Sunnyvale, CA 94086
(berenji, vengerov, jayesh14)@iiscorp.com

Abstract— **In this paper we study the problem of sensor allocation in Unmanned Aerial Vehicles (UAVs). Each UAV uses perception-based rules for generalizing decision strategy across similar states and reinforcement learning for adapting these rules to the uncertain, dynamic environment. A big challenge for reinforcement learning algorithms in this problem is that UAVs need to learn two complementary policies: how to allocate their individual sensors to appearing targets and how to distribute themselves as a team in space to match the density and importance of targets underneath. We address this problem using a co-evolutionary approach, where the policies are learned separately, but they use a common reward function. The applicability of our approach to the UAV domain is verified using a high-fidelity robotic simulator. Based on our results, we believe that the co-evolutionary reinforcement learning approach to reducing dimensionality of the action space presented in this paper is general enough to be applicable to many other multi-objective optimization problems, particularly those that involve a tradeoff between individual optimality and team-level optimality.**

I. Introduction

The problem of sensor allocation in teams of UAVs operating in uncertain, dynamic environments is very challenging. No existing centralized control approaches can provide a fully adequate solution, since at any point in time the space of possible future states of each UAV is very large, possibly continuous. Any centralized approach that keeps track of all UAV's states will be defeated by the "curse of dimensionality" – exponential explosion of possible states to consider as the number of UAVs increases. In addition, if the local environment of each UAV also contains a large number of objects (e.g. targets, obstacles, etc.) whose properties change over time, developing good sensor allocation policies even for a single UAV becomes a major challenge.

Moreover, the sensor allocation policy of each UAV should be aware of other nearby UAVs to reduce the chance of tracking unnecessarily the same targets. That is, at the individual level, UAV's sensors need to be dynamically allocated to tracking various targets. At the team level, each UAV can be treated as a composite sensor, and these sensors need to be allocated to different regions of the search space in proportion to the density and importance of targets there, in order to maximize the team efficiency.

We propose to address the above challenges by using perception-based reinforcement learning in conjunction with the distributed co-evolutionary multilevel learning framework described in [1]. Instead of using a single policy for decision-making that involves both sensor management and geographical UAV allocation together, this framework uses two different but complementary policies on individual and group levels that co-evolve toward a common goal.

In section II we describe the way in which rule-based systems in combination with the Computational Theory of Perceptions (CTP) can be used for evaluating various action alternatives available to autonomous decision makers. Section III presents equations of a reinforcement learning algorithm for tuning the coefficients of perception-based rules. In section IV we describe a mathematical programming formulation of the sensor allocation problem faced by a team of UAVs. A perception-based co-evolutionary reinforcement learning approach for solving this problem is described in section V. The experimental results are presented in section VI and section VII concludes the paper.

II. Computational Theory of Perceptions

Proposed by Lotfi Zadeh [3], the Computational Theory of Perceptions (CTP) is based on Computing with Words

(CW), where *granulation* plays a critical role for data compression. By granulating input variables and treating them as high-level *perceptions*, compact and broadly applicable inference rules can be defined. In the UAV domain, we will use the following perception-based rules for evaluating state-action pairs during UAV sensor allocation policies:

Rule i: IF s_1 is S_1^i and s_2 is S_2^i and ... and s_K is S_K^i THEN (Value is q^i),

where s_j is the jth attribute of the currently considered state-action pair, S_j^i is the corresponding fuzzy label and q^i is a tunable coefficient representing the value suggested for this state-action pair by rule i. Each fuzzy label can be represented by a membership function $\mu_{S_j^i}(s_j) : R \to R$ that maps its input into a degree to which this input belongs to the fuzzy category (linguistic term) described by the label. The weight of each rule will be computed using the product inference: $w^i(s) = \prod_{j=1}^{K} \mu_{S_j^i}(s_j)$. The final conclusion of a fuzzy rulebase function $f(s)$ with M rules is given by:

$$Q = f(s) = \frac{\sum_{i=1}^{M} q^i w^i(s)}{\sum_{i=1}^{M} w^i(s)}. \tag{1}$$

After defining the perception-based rules for UAV sensor allocation, we will use reinforcement learning to tune the rule recommendations q^i, corresponding to each perception.

III. Reinforcement Learning Algorithm

In our simulations we have combined the Q-learning algorithm of Watkins [2] with perception-based fuzzy logic function approximation. In the original Q-learning algorithm, $Q(x, a)$ is defined as an expectation of a discounted sum of future rewards after taking action a in state x and following the optimal policy thereafter. Once the accurate Q-values have been computed for all state-action pairs, the optimal policy can be defined by taking in each state x the action a that maximizes $Q(x, a)$. In order to learn the optimal Q-values, the following equation is used at every time step, after taking action a_t in state x_t, receiving the reward r_t and moving to the next state x_{t+1}:

$$Q(x_t, a_t) = Q(x_t, a_t) + \alpha_t \delta_t, \tag{2}$$

where α_t is the learning rate and δ_t is given by:

$$\delta_t = r_t + \gamma \max_a Q(x_{t+1}, a) - Q(x_t, a_t), \tag{3}$$

with γ being the discounting factor. The above equation was proven to converge to optimal Q-values in finite state and action spaces if each action is chosen infinitely many times in each possible state and the learning rate is suitably decreased at every time step.

When the state space is very large or continuous, Q-learning cannot explore all possible states, and hence the Q-values need to be generalized across similar states. This is accomplished by using a function approximation architecture $Q(x, a, \Theta)$, where Θ is the matrix of tunable parameters, whose columns correspond to possible actions. That is, if the jth possible action was taken at time t, then the jth column of Θ is used to compute the approximation to $Q(x_t, a_t)$. The basic parameter updating rule used by discounted Q-learning for such an architecture is:

$$\Theta_t \leftarrow \Theta_t + \alpha_t \delta_t \nabla_{\Theta_t} Q(x_t, a_t, \Theta_t), \tag{4}$$

where δ_t is the Bellman error used in the corresponding learning rule for the look-up table case presented in equation (3). Notice that only the column of Θ corresponding to action a_t gets updated.

In the general version of discounted Q(λ)-learning, equation (4) becomes:

$$\Theta_t \leftarrow \Theta_t + \alpha_t \delta_t \sum_{\tau=T_0}^{t} (\gamma\lambda)^{t-\tau} \nabla_{\Theta_t} Q(x_\tau, a_\tau, \Theta_t), \tag{5}$$

where T_0 is the time when the current episode began.

The analytical expression for approximating the Q-value at any time t using a perception-based rulebase is:

$$Q(x, a, \Theta) = \sum_{k=0}^{K} \Theta_k(a) \mu_k(x), \tag{6}$$

where $\Theta_k(a)$ can be interpreted as the Q-value of taking the action a in the k-th fuzzy state s_k and $\mu_k(x)$ is the degree of membership of state x to s_k. If generalization over action space is used, then equation (6) still applies after changing $\mu_k(x)$ to $\mu_k(x, a)$ and learning only K parameters $\theta_k, k = 1, ..., K$ instead of K parameters for each possible action a. In this case, $\nabla_{\theta_t} Q(x_t, a_t, \theta_t)$ becomes the vector $(\mu_1(x_t, a_t), ..., \mu_K(x_t, a_t))^T$. Thus, equations (4) and (5) can now be rewritten component wise as:

$$\theta_k \leftarrow \theta_k + \alpha_t \delta_t \mu_k(x_t, a_t), \tag{7}$$

$$\theta_k \leftarrow \theta_k + \alpha_t \delta_t \sum_{\tau=T_0}^{t} (\gamma\lambda)^{t-\tau} \mu_k(x_\tau, a_\tau). \tag{8}$$

The above equations have a natural interpretation in the realm of fuzzy state aggregations: the Q-value of a fuzzy state-action pair (s_k, a) gets updated proportionally to its contribution to the Q-value of the state-action pair (x_t, a_t) in equation (6).

IV. Problem Formulation

We have developed a mathematical programming framework, which allows finding sensor allocation policies that are optimal not only for individual UAVs but also for the multi-UAV team as a whole. The objective of each UAV is to choose actions in consecutive time periods t = 0, 1, 2, ... so as to maximize the expected value of the discounted sum of future rewards:

$$\max_{a_t, t=0,1,\ldots} E[\sum_{t=0}^{\infty} \gamma^t r_t(s_t, a_t)] \tag{9}$$

subject to the constraint on the sequence of states s_t:

$$s_{t+1} = f(s_t, a_t), \tag{10}$$

where γ is the discounting factor and $r_t(s_t, a_t)$ is the reward received at time t in state s_t after taking the action a_t.

We chose a very general reward function, which reflects simultaneously many of the problem complexities that we would like the team of UAVs to optimize. The reward received by the kth UAV for tracking all targets within its sensor range (e.g. its field of vision) after having aligned itself with target j is given by:

$$r_{kj} = \sum_{n=1}^{N} (\frac{V_n}{1+d_{kn}^2})(\frac{\frac{1}{1+d_{kn}^2}}{\sum_{m=1}^{M} \frac{1}{1+d_{mn}^2}}), \tag{11}$$

where N is the total number of targets within its field of vision once the sensors are pointed at target j, M is the number of UAVs tracking target n, d_{kn} is the distance between UAV k and target n, and V_n is the value of target n. The above form of the reward function allows UAVs to learn the sensor allocation policy that tends to track targets that have higher values, that are closer to the UAV, targets that are bunched together, and that do not cause increased competition among UAVs for rewards.

The action a_t of each UAV needs to optimize simultaneously two aspects of the reward function. On the one hand, the UAV needs to be close to the high-valued targets in order to maximize the first term of the reward function, $\frac{V_n}{1+d_{kn}^2}$. On the other hand, the team-level optimality of the multi-UAV sensor allocation requires that UAVs do not duplicate each other's efforts by allocating all of their sensors to the single highest-valued target while leaving other targets unattended. The second component of the reward function, $\frac{\frac{1}{1+d_{kn}^2}}{\sum_{m=1}^{M} \frac{1}{1+d_{mn}^2}}$, attempts to prevent such a behavior by rewarding each UAV more for tracking targets that do not have many other UAVs around them. Therefore, the action of each UAV has two components: individual decision of choosing the target with which to align the sensors and the team-level decision of where to position itself in the metric space inhabited by other UAVs. Because of the different physical nature of these components, different parts of the state s_t will be most relevant for making each decision.

To complete the mathematical programming formulation of the multi-UAV sensor allocation policy, the state transition function f needs to be specified in equation (10). Due to the complexity of the considered problem, this function cannot be expressed analytically. However, it can be simulated by following the motion strategy of all UAVs as well as by simulating appearance and disappearance of targets in the search area. Fortunately, the simulation-based description of the state transition targets is sufficient for reinforcement learning algorithms to learn how to iteratively improve the actions in order to maximize the reward function $r_t(s_t, a_t)$.

V. Solution Methodology

In order to make UAVs learn policies that are both tractable and robust to changes in the number of targets or UAVs observed, we developed an approach allowing each UAV observe only the most relevant parts of the complete state vector (involving information about all targets and all UAVs) at each time step t when making the individual sensor allocation decision. Also, we developed a potential-field feature extraction method that can be used by the team-level motion strategy [5]. These methods are general in nature and can be used in many different problem scenarios.

In order to extract the most relevant information from the high-dimensional state vector, we used the potential field approach for compactly encoding information about location of multiple objects. That is, since the presence of each object – target or UAV – is important only in its local neighborhood, we treated it as a potential charge, whose value decays with the squared distance from it. With this view, we found the following two variables to be sufficient for learning the proper direction of change in the individual sensor allocation component of the action a_t by UAV k:

1) $x[1] = \sum_{n=1}^{N} \frac{V_n}{1+d_{kn}^2}$
2) $x[2] = \sum_{m=1}^{P} \frac{1}{1+d_{jm}^2}$,

where N is the total number of targets within UAV's field of vision once the sensors are pointed at target j, P is the total number of other UAVs and d_{jm} is the distance between target j and UAV m. The variable $x[1]$ represents the sum of potentials of all targets that the UAV expects to track if it aligns its sensors with the jth target, while $x[2]$ represents the sum of potentials of all other UAVs near the jth target.

Using the same potential field approach, we found the following two variables to be sufficient for learning the proper direction of change in the team-level motion component of the action a_t:

1) $y[1]$ = "Target potential"
2) $y[2]$ = "UAV potential"

The gradient of the "Target potential" determines for each UAV the direction of motion leading to the greatest concentration of targets. All else being equal, this should be the preferred direction of motion. The gradient of the "UAV potential" determines for each UAV the direction of motion leading to the greatest concentration of other UAVs. All else being equal, this should be the least preferred direction of motion, so as to cause least competition for available targets. The team-level motion strategy for each UAV will help it to tradeoff the target and the UAV potentials at future locations in various circumstances.

More specifically, a target j contributes the following amount to the target potential at location i in the world: $P_{ij} = \frac{V_j}{1+d_{ij}^2}$, where V_j is the value of the jth target and d_{ij} is the distance between the target and the considered location. A similar formula holds for computing the UAV field, except that the potential sources are other UAVs and $V_j = 1$ is the value assigned to each UAV. Different values may be assigned to different UAVs in case of a heterogeneous set of UAVs that have different capabilities. The variable x_1 is the sum of potential of all targets at the considered location for the UAV, and x_2 is the sum of potential of all other UAVs. A graphical representation of this tile world is shown in Fig. 1, with darker locations having a higher target potential.

The decision variables used by UAVs are assigned to a number of fuzzy categories depending on the level of granularity intended for it. For example, in the simplified case, we can have two fuzzy categories, SMALL and LARGE. Each state variable will be SMALL to a certain degree and LARGE to a certain degree, according to the value of these linguistic categories at each point in space. In our simulations, we chose a simple linearly decreasing functions for SMALL and a linearly increasing function for LARGE.

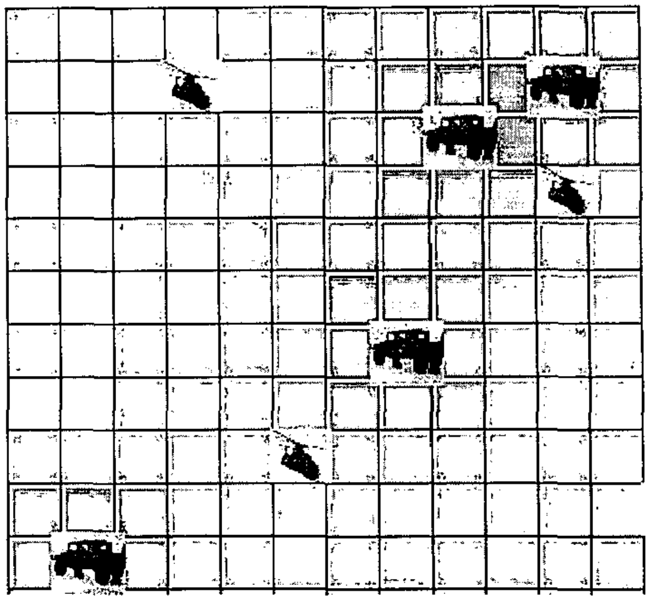
Fig. 1. A potential surface model of the UAV domain, with darker locations having a higher target potential.

If only two fuzzy labels are used for both the individual sensor allocation decision and the team-level motion planning decision, then the following rules will be used by each UAV (for evaluating the utility of aligning its sensors with each of the targets or for evaluating the utility of moving in each of the possible major directions):

IF (s_1 is SMALL) and (s_2 is SMALL) then (q^1)
IF (s_1 is SMALL) and (s_2 is LARGE) then (q^2)
IF (s_1 is LARGE) and (s_2 is SMALL) then (q^3)
IF (s_1 is LARGE) and (s_2 is LARGE) then (q^4),

where s stands for the individual decision variable x or team-level decision variable y as explained in the previous section. The final utility of each target is computed by combining the recommendations of individual rules using equation (1). The values $q^1, ..., q^4$ are tuned by equation (8) with $T_0 = 0$ for both policies. Since the input variables x and y combine information about the state of the UAV and the action being considered, they in effect implement a generalization over the action space. This allows learning only a single vector of parameters, common to all actions.

The overall learning procedure of a UAV is as follows. First, the UAV selects its next location by choosing the one with the highest Q-value with probability p_1 and choosing any other location at random with probability $1 - p_1$. After arriving at the new location, the UAV chooses its next target toward which its sensors should be pointed. The target with the highest Q-value (from the individual sensor allocation policy) is chosen with probability p_2 and any other target is chosen at random with probability $1 - p_2$. Once the motion and rotation phases have been accomplished, the UAV computes its one-step reward according to equation (11) and uses it for updating the Q-values of both fuzzy rulebases according to equation (8). After updating the Q-values, the UAV selects a new location and sensor alignment direction, and the cycle repeats.

The potential field motion planning strategy described above is fully distributed and robust to any changes in the environment. The decisions of each UAV change gradually as the environment changes without the need for a complete "re-planning" of classical planning strategies. Also note that the individual and team sensor allocation policies are affecting each other's environment by changing the relationship between actions and expected rewards. However, since both UAV allocation policy and sensor allocation policy have the same common goals: tracking higher valued targets, closer targets, targets surrounded by other nearby targets, and tendency to reduce competition with other UAVs, it is expected that they will be able to tune themselves in a co-evolutionary manner.

VI. Experimental Results

Player-Stage software [6] was used for simulating our problem setup. A bounded environment with no physical obstacles was chosen for clarity of our results. We used a simple 2D square-shaped environment of length 2 units with no physical obstacles. In our experimental setup, 3 UAVs move in this environment, attempting to track 6 moving targets. Targets have different values represented by their color. Size of a UAV and targets is 0.05 units and 0.025 units respectively.

The targets use sonar sensors to detect UAVs around them. If a UAV comes closer than a pre-selected minimum threshold distance to any target, the target moves in the exact opposite direction from the UAV in order to avoid it. The UAVs can move over each other and over the targets in the 2D simulator.

Each UAV has the following set of simulated sensors:

1) Sony EVID30 pan-tilt-zoom camera set to a range of 60 degrees, with ACTS – a fast color segmentation program for identifying colors of targets coming in the camera's range (the target's color gets translated into the target's value)
2) SICK LMS-200 laser rangefinder for measuring distance to other targets or UAVs
3) GPS device for exactly locating its own position in the environment with respect to a fixed reference point.

In the beginning of our simulation process, antecedent labels for the fuzzy rules used by each UAV had to be created. In order to accomplish this, a simple simulation was used, where the UAVs and targets move in the environment, and data corresponding to the state variables is collected. The UAVs tend to avoid other UAVs, while targets tend to avoid UAVs and each other. Based on the range and distribution of data obtained, the fuzzy categories LOW and HIGH were identified for each state variable. Since two state variables were used by the individual sensor allocation policy as well as by the UAV motion policy, 4 rules were created for each of them.

A sensible set of initial Q-values was assigned to the policies, and performance of the UAVs measured as the average reward was evaluated. The antecedent labels were then manually tuned to improve the average reward, while ensuring that all rules are triggered to similar cumulative levels for effective learning during the training phase. This tuning could have also be done using our co-evolutionary reinforcement learning algorithm at the expense of adding extra complexity to the problem.

The TD(λ) updating of fuzzy rulebase parameters based on equation (8) was used, with θ_k being the consequent label parameters q^k in accordance with the fuzzy rulebase presented in the previous section. A training run had a fixed duration of 30 minutes, which translates into about 1500 steps for one UAV. The UAVs and targets were placed in the environment randomly at the start of each simulation. After every 60 seconds, they were re-randomized to ensure that all possible states have been visited adequately. Both co-evolutionary policies used the same one step reward function, as given by equation (11).

During learning, the UAVs used and updated a common policy for individual sensor allocation and a common policy for team-level UAV allocation. Such a coopera-

	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 0.9$
Before learning	1.1	1.1	1.1
After learning	2.55	2.52	2.25

TABLE I
AVERAGE REWARD OF THE FINAL POLICY LEARNED BY UAVS.

tive learning method for RL agents can potentially speed up the learning process by more than a factor of N if N agents are participating [4]. The learning proceeded from a completely uninformed situation where all Q values are set to 0. In the beginning, each UAV used high exploration probabilities p_1 and p_2. However, over time both p_1 and p_2 were decreased geometrically by a factor ϵ, and each UAV tended to choose actions that have the highest Q value.

Table I shows the values of the average reward received by the UAVs during the testing phase for various values of the TD-parameter λ. The table also shows the values of the average reward for the initial policy that used Q-values equal to 0 – choosing all actions with equal probability. The following values for the key parameters were used:

- Learning rate α for both policies = 0.75
- Discounting factor γ = 0.9
- Exploration decay rate $\epsilon = 0.998$

As Table I shows, the UAVs significantly improved their performance as a result of learning with our co-evolutionary algorithm. The decrease in performance for higher values of λ is most likely caused by the fact that as the time separation between actions and rewards increases, the connection between them decreases faster than in a single-policy reinforcement learning due to the presence of a second policy, which is also changing with time.

The performance improvement recorded in Table I demonstrates the effectiveness of the two methods we used for reducing the dimensionality of the state space: evaluating targets one at a time for individual sensor allocation and using potential fields for team-level decision-making. These results also demonstrate the benefits of the co-evolutionary learning between the individual and the team-level policies in reinforcement learning implementation.

VII. CONCLUSIONS

In this paper, we studied a co-evolutionary approach for sensor allocation in UAVs. The Perception based Reinforcement Learning (PRL) algorithm was proposed for the joint optimization of individual sensor allocation policy and the team motion policy. Experimental results demonstrated the benefits of the two-level learning organization. We conclude that co-evolutionary PRL seems to be a very promising approach which needs to be further investigated in the field of Unmanned Aerial Vehicles and other areas.

ACKNOWLEDGMENT

This work was partially supported by the ONR grant N00014-02-M-0265 for Dr. Allen Moshfegh.

REFERENCES

[1] A. Vengerov, (2002). "Toward integrated pattern-oriented and case based design framework in complex multi-agent system development for e-business environment." Proceedings of the Fifth Annual International Conference of American Society of Business and Behavioral Sciences (ASBBS), London.
[2] Christopher J. C. H. Watkins. *Learning from delayed rewards.* PhD thesis, King's College, Cambridge, UK, 1989.
[3] L. Zadeh, (1999) "From computing with numbers to computing with words – from manipulation of measurements to manipulation of perceptions," *IEEE Transactions on Circuits and Systems*, vol. 45, pp. 105-119.
[4] Hamid R. Berenji, David A. Vengerov. (2000) "Advantages of cooperation between reinforcement learning agents in difficult stochastic problems." In proceedings of the 9th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE '00).
[5] David A. Vengerov, Hamid R. Berenji, Alexander B. Vengerov (2002) "Adaptive coordination among fuzzy reinforcement learning agents performing distributed dynamic load balancing," In proceedings of the 11th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE '02).
[6] Brian P. Gerkey, Richard T. Vaughan, Kasper Stoy, Andrew Howard, Maja J Mataric and Gaurav S Sukhatme. "Most valuable player: A robot device server for distributed control," In proceedings of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), pages 1226-1231, Wailea, Hawaii, October 2001.
[7] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, MIT Press, 1998.
[8] D. Bertsekas and J. Tsitsiklis. *Neuro-dynamic programming*, Athena Scientific, 2000.