# Imitation Learning for Task Allocation

Felix Duvallet
Robotics Institute
Carnegie Mellon University
felixd@cmu.edu

Anthony Stentz
Robotics Institute
Carnegie Mellon University
tony@cmu.edu

***Abstract*—At the heart of multi-robot task allocation lies the ability to compare multiple options in order to select the best. In some domains, this utility computation is not straightforward, for example due to complex and unmodeled underlying dynamics or an adversary in the environment. Explicitly modeling these extrinsic influences well enough so that they can be accounted for in utility computation (and thus task allocation) may be intractable, but a human expert may be able to quickly gain some intuition about the form of the desired solution.**

**We propose to harness the expert's intuition by applying imitation learning to the multi-robot task allocation domain. Using a market-based task allocation method, we steer the allocation process by biasing prices in the market according to a policy which we learn using a set of demonstrated allocations that represent the expert's solutions (accounting for external influences). We present results in two distinct domains: a disaster response domain where a team of agents must put out fires that are spreading between buildings, and an adversarial game in which teams must make complex strategic decisions to score more points than the opposite team.**

## I. Introduction

The goal of any multi-robot task allocation mechanism is to maximize utility, an essential and unifying concept which represents an estimate of the system performance [1]. For example, a team of robots exploring an environment may wish to maximize the area observed while minimizing costs for traveling [2], or a team of searchers may want to maximize the likelihood of finding an evader while minimizing the time-to-capture [3]. In domains such as these, the utility metric is simple to express, and can easily be derived from the high-level goals.

In some domains, however, estimating utility is not as straightforward. There may exist rich underlying dynamics or an adversary team in the domain that is not fully modeled. Although forming a complete and explicit understanding of the world may be intractable or impossible, a human observer may have some previous experience or be able to quickly gain some intuitive understanding from observing the environment. Though this knowledge may be hard to articulate into an explicit algorithm, policy, or utility mapping, the expert will generally be able to recognize a good solution.

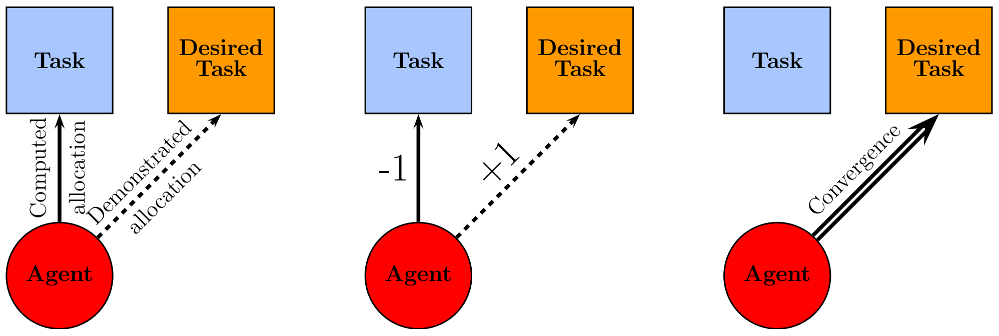
This domain expert may have an end-result behavior in mind when allocating tasks, but developing a hand-tuned utility mapping to produce it is a tedious and sometimes difficult process involving many iterations of policy tweaking and testing [4]. As the number of "knobs" to tune grows with increasing domain complexity, more and more potential policies must be validated, and this approach quickly grows intractable.

To address this problem, we apply imitation learning to the problem of multi-robot task allocation. A set of expert demonstrations provide desired allocations, and we generalize these expert examples to learn a utility mapping. Our approach is based on Maximum Margin Planning (MMP), a popular imitation learning framework which has successfully been used to learn utility mappings in other domains such as overhead imagery interpretation, footstep prediction, and 3D point cloud classification [4], [5]. Imitation learning provides an intuitive method for specifying a task allocation policy, and is much more efficient than manually crafting a cost function.

In our approach, we have chosen to use a market-based task allocation mechanism. Though any allocation method can be used, markets have been shown to be fast, scalable, and yield solutions that are close to optimal [6]. Based on empirical evidence, markets converge quickly to locally optimal solutions for even complex problems [7]. A market-based task allocation mechanism is ideally suited for our imitation learning approach because prices represent the system utility very compactly and can easily be biased by the auctioneer by introducing certain types of incentives for task completion that are independent of the task's true reward.

We demonstrate our approach in two distinct simulated domains: a disaster response domain and an adversarial strategic game. In the disaster response domain, the expert's intuition about the underlying fire dynamics given the surrounding layout of buildings may be hard to articulate algorithmically, but the expert can quickly and easily demonstrate a response to the particular disaster. In the adversarial strategic game, the expert is able to learn (through repeated games) a strategy which can effectively counteract the opponent's. Harnessing the expert's experience, we observe them playing the game to collect training examples. In both domains we show that imitation learning can be used to generalize from expert demonstrations to learn a utility mapping for a complex task allocation environment.

We begin by reviewing related work in both task allocation and imitation learning (Section II). We introduce Maximum Margin Planning and market-based task allocation in Sections III and IV, and illustrate how both are used together to imitate an expert's task allocations by biasing prices in the



(a) Demonstration (b) Subgradient update (c) Learned Policy

Fig. 1. Intuitive visualization of one iteration of the Max Margin subgradient update step for task allocation described in Section III. Starting with some demonstrated and computed task allocations (1a), the reward (in feature space) is then decreased for the computed allocation and increased for the demonstrated allocation (1b). This process iterates until convergence (1c).

market. We illustrate experimental results (Section V) and provide an analysis (Section VI), then conclude and discuss potential future directions for this work (Section VII).

## II. Related Work

Learning is playing an increasingly important role as robotics systems become more complex. Within this large body of work, techniques that utilize imitation learning (also known as learning from demonstration) are of particular interest, as they benefit from the presence of an expert who can provide examples of the optimal (or desired) behavior [8]. Imitation learning has been used to develop an autonomous driving vehicle [9], improve overhead imagery interpretation [4], and develop helicopter controllers [10].

Market-based task allocation is also a well-studied mechanism that is used in many applications including mapping [2] and cooperative manipulation [11]. Though this approach has been used successfully to tightly coordinate robots in complex scenarios, the utility function the team is optimizing has so far been fairly straightforward to express and can be derived directly from the domain goals. For example, Kalra et al. use Hoplites (a market-based framework) to tightly coordinate robots in a security sweep domain, where the goal is to sweep through an environment while detecting any evaders. Accordingly, they use the market to reward agents for sweeping parts of the environment, and assess large penalties for violating the visibility constraint [12].

Learning has been applied to markets to influence trading. In an oversubscribed disaster-response scenario, Jones et al. enhance task allocation by modifying agent bids to better predict task penalties assessed for failing to complete tasks [13]. In their approach, learning was used to model the underlying causes for disparities between expected and actual rewards the agents receive, something which would be hard to model explicitly *a priori*. After collecting a large number of example allocations and their associated penalties, they use regression to improve future bidding by creating a model for the actual value the task given the agent's current schedule and other task features. Similarly, Schneider et al. learn (online) the opportunity cost for taking on tasks in a market-based multi-robot scenario [14].

Though these approaches have successfully applied supervised and reinforcement learning techniques to task allocation, we believe there are many applications where imitation learning is a superior approach for several reasons:

- Imitation learning explicitly utilizes an expert's knowledge. This expert may bring a deep and complex understanding of the domain, which can lead to better system performance much faster than if the system trained itself.
- The time and resources required to learn using reinforcement learning may not be acceptable in some real-world scenarios, where there is no training phase and poor performance is not acceptable at any time.
- An expert who is monitoring the system can detect changing conditions before a decrease in the total objective or utility, and therefore may be able to re-train the system before performance is greatly impacted.

We use the term expert loosely, in many cases a few rounds of experience is enough for a human player to gather enough information to create a rough internal model of the underlying dynamics or adversary, and this intuition can be used to train the system.

## III. Maximum Margin Planning

Maximum Margin Planning is an imitation learning framework for structured prediction over the space of policies [4]. We denote by $\mu$ a particular set of allocations from the space of possible allocations $\mathcal{G}$. We can extract a feature vector $f \in \mathbb{R}^d$ for each possible task allocation and accumulate these features in a matrix $F$ (the product $F\mu$ represents the accumulation of features encountered by following the policy $\mu$ [4]). Each training example $i$ consists of a set of tasks, agents, and the expert allocation $\mu_i \in \mathcal{G}_i$. Each example also includes a loss field $l_i$, so that $l_i^T\mu$ quantifies how bad a policy $\mu$ is compared to the demonstrated policy $\mu_i$. Written more concisely, our training data set is $\mathcal{D} = \{(F_i, \mathcal{G}_i, \mu_i, l_i)\}_i^N$.

Our goal then is to learn a set of weights so that each demonstrated allocation is better than every other possible allocation for the same scenario:

$$w^T F_i \mu_i \geq w^T F_i \mu + l_i^T \mu \qquad \forall i, \mu \in \mathcal{G}_i \tag{1}$$

The loss field $l_i$ in this inequality requires that the demonstrated allocation be better than every other allocation by some margin during training, and improves testing performance [4].

If the constraint (1) holds for all allocations in $\mathcal{G}_i$, it must hold for the best allocation [15], thus the only constraint we need to consider is:

$$\mu_i^* = \arg\max_{\mu \in \mathcal{G}_i} \left[ \left( w^T F_i + l_i^T \right) \mu \right] \tag{2}$$

which allows us to rewrite the constraints from equation (1) as

$$w^T F_i \mu_i \geq \max_{\mu \in \mathcal{G}_i} \left[ w^T F_i \mu + l_i^T \mu \right] \qquad \forall i \tag{3}$$

This presents us with the following quadratic program [4]:

$$\begin{array}{cl} \min\limits_{w, \xi_i \in \mathbb{R}_+} & \frac{\lambda}{2} \|w\|^2 + \frac{1}{N} \sum\limits_{i=1}^{N} \xi_i \\ \text{s.t.} & w^T F_i \mu_i + \xi_i \geq \max\limits_{\mu \in \mathcal{G}_i} \left[ w^T F_i \mu + l_i^T \mu \right] \quad \forall i \end{array} \tag{4}$$

where $\xi_i$ are slack terms and $\lambda$ is a regularization constant. While solving this quadratic program is difficult, we note that at the optimum, the slack variables will exactly equal the constraint violation [16], and they can thus be included in the objective function:

$$c(w) = \frac{\lambda}{2} \|w\|^2 + \frac{1}{N} \sum_{i=1}^{N} \left( \max_{\mu \in \mathcal{G}_i} \left[ \left( w^T F_i + l_i^T \right) \mu \right] - w^T F_i \mu_i \right) \tag{5}$$

Though this objective is convex, it is not differentiable. We can, however, optimize it efficiently by using the subgradient method and a fast task allocation solver [4]. The subgradient of Equation (5) is:

$$g_w = \lambda w + \frac{1}{N} \sum_{i=1}^{N} \left( F_i \mu_i^* - F_i \mu_i \right) = \lambda w + \frac{1}{N} \sum_{i=1}^{N} F_i \Delta^w \mu_i \tag{6}$$

where $\mu_i^* = \arg\max_{\mu \in \mathcal{G}_i} \left[ \left( w^T F_i + l_i^T \right) \mu \right]$ is the solution to the loss-augmented problem, and $\Delta^w \mu_i = \mu_i^* - \mu_i$ computes the difference in feature accumulations between the computed and demonstrated allocations (under the current policy $w$).

Our subgradient update rule then becomes:

$$w_{t+1} = w_t - \alpha g_w \tag{7}$$

for a learning rate $\alpha$. Intuitively, this gradient update rule increases the reward (in feature space) on the demonstrated allocation ($\mu_i$) and decreases the reward (again in feature space) on the chosen allocation ($\mu_i^*$). Note that when the demonstrated allocations matches the computed allocations, $\Delta^w \mu_i$ is zero. A simplified visualization of the Max Margin update for task allocation is shown in Figure 1.

## IV. Market Based Task Allocation

The market-based approach for task allocation models robots as self-interested agents and the team of robots as an economy [17]. Each robot estimates their fitness for a task and encodes this utility in a price which can be understood by all other agents on the team. Since all revenue is generated from accomplishing team-wide goals, maximizing an agent's individual profit yields better overall team performance [6]. Furthermore, since agents and the auctioneers only reason about (and transmit) price information, task allocation is decentralized, fast, and robust, making it ideally suited for allocating tasks in dynamic environments.

### *A. MMP and Market Based Task Allocation*

In order to steer the task allocation mechanism towards the demonstrated allocations, we introduce a bias term into bids, so that profit used for bidding is now [13]:

$$\text{profit} = \text{reward} - \text{cost} + \text{bias}. \tag{8}$$

This biasing term uses the learned feature weighting vector $w$ (see Section III):

$$\text{bias}_k = w^T f_k \tag{9}$$

for features $f_k$ associated with task $k$. Note that we are not restricted to use a linear combination of the features [4].

## V. Experimental Results

We performed imitation learning on task allocations from two differing domains: a disaster response domain in which the goal is to minimize the total damage to all buildings after a disaster event occurs, and an adversarial strategic scenario where the goal is to maximize points scored while minimizing the adversary's score. Both are difficult domains for task allocation, exhibiting complex underlying dynamics or an adversary team, and require a considerable amount of reasoning to make an informed allocation.

### *A. Disaster Response*

The disaster response scenario models a world immediately following a disaster. A number of buildings of different sizes exist in a simulated world which is affected by a disaster event, igniting a number of them as a result. A team of responders must and attend to the fires. Fires spread between buildings and fires may randomly appear (caused by the disaster's aftershocks). The goal of the response team is to minimize the damage to buildings. We note that the team is oversubscribed, as there are many more tasks than agents available, and thus allocations must be made carefully. Although the agents know which buildings are on fire, they do not know which buildings are most likely to combust in the future. Determining these fire dynamics would involve reasoning about the distribution of buildings (buildings are more likely to combust if they are closer to fires) and their past (fires spread more the longer they burn). However, an expert may already have some understanding of the fire dynamics and an intuition for the best response. For example, traveling to a small fire which is near a large cluster of unaffected buildings may be preferable to traveling to a building which may be closer but is isolated.

*1) Demonstrations:* The demonstrations are hand-picked examples of common scenarios encountered for allocation. They are identified or created by the expert as exhibiting a qualitatively good allocation strategy. The system in this domain was trained using six examples, each consisting of an instance of tasks, agents, and allocations for the team.

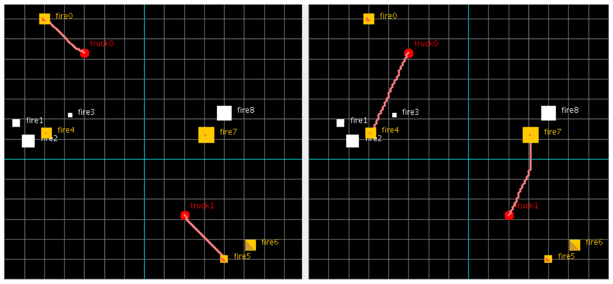


Fig. 2. Simple example showing the default (left) vs. learned (right) strategies. Under the learned policy, the leftmost agent travels further to reach a cluster of tasks, preventing fire spread. The rightmost agent goes for a large building instead of a small building which is almost burned down.

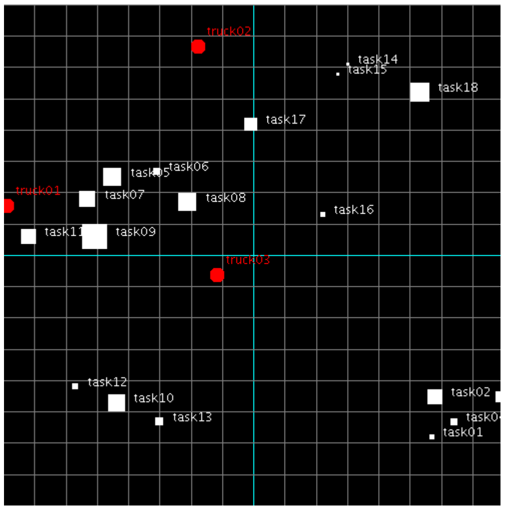


Fig. 3. A randomly generated world for performance testing with three agents (red circles) and 18 buildings (solid white squares).

2) *Task Features:* Each task in the environment has a set of features that encapsulates discriminative information about each individual task and its relation to other tasks in the world. We list below the features that are computed per task (omitting normalizing information):

- Fire: Whether a building is on fire or not.
- Size: Building size.
- Health: Current building health.
- Closest Task Neighbor: Proximity to next-closest task.
- Closest Agent Neighbor: Proximity to closest agent.
- Blurred building sizes: A Gaussian blurring representing the density of buildings in the neighborhood.
- Blurred building health: The same density measure as above, but with each building weighted by its current health.
- Constant: Constant term (1.0).

The first features depend only on the current state of each individual building. The distance to the closest neighbor feature provides a biasing for tasks which are closer to other buildings. The Gaussian features are a "smoothed" map of the building sizes (weighted by the health for blurred building health). The features listed here were chosen for their saliency and ability to encode the information an expert might use to make allocation decisions.

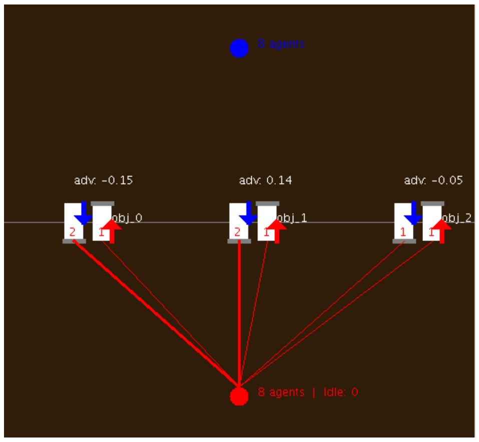


Fig. 4. Strategic domain before the game begins. Three passageways have Offensive and Defensive agents from the red team allocated to each. The opposite (blue) team's allocation is hidden until after the game begins.

3) *Results:* A side-by-side comparison of the default and learned policies applied to the same sample environment are shown in Figure 2. The default policy weights all tasks equally during task allocation (and thus attends to the closest fire). The expert's strategy of attending to tasks in clusters is evident under the learned policy.

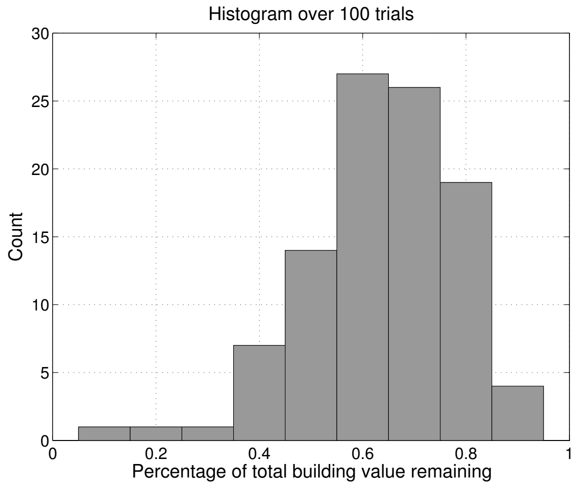
We also evaluated the learned policy on a random instance of the disaster domain with three agents and 18 buildings (show in in Figure 3). One hundred trials were run for 500 time steps each, with a different random initialization of the starting fires for each trial. We established a metric of percentage of total building value remaining (sum over all buildings of size $\times$ health).

For a baseline comparison in our evaluation, we ran each trial twice: once with the learned policy and once with the default policy. In the head-to-head trials, the learned policy outperformed the default policy in 63 of 100 tests. Histograms of the remaining building value at the end of the trials are shown in Figure 5. The mean value for the learned policy was higher overall (64% compared to 53%), and this policy produced a much larger proportion of instances where more than half of the starting health remained. These results are statistically significant ($p = 5 \times 10^{-6}$).

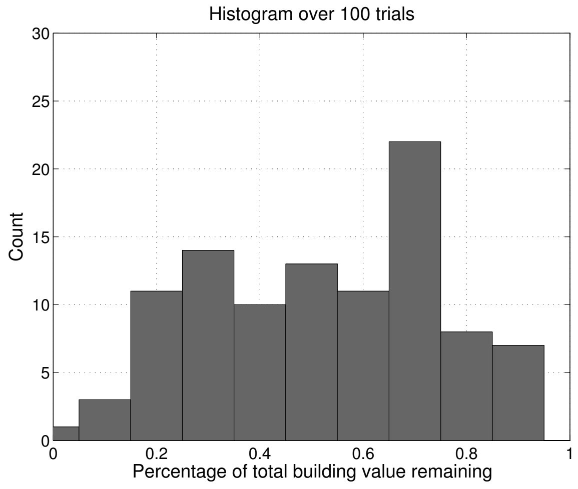
## B. *Adversarial strategic game*

The adversarial domain has two teams playing against (and reasoning about) each other in a zero-sum game. Both teams have an equal number of agents starting in their respective bases, and each team's goal is to maximize the number of agents that reach the opposite base while minimizing the number of adversary agents reaching their base. A number of passageways connect the two teams, and before the game begins the teams must allocate their agents, either as offense or defense (these become their strategic task allocations). The only difference between the different passages is that each has a different "strategic advantage" associated to them, a value which indicates how likely the offense (or defense) is to win a one-on-one matchup.

Neither team is aware of the other's allocation as they



(a) Learned Strategy



(b) Default Strategy

Fig. 5. Comparative results in the disaster response domain between the learned policy and a default policy over 100 trials.

allocate tasks, and the game begins after both teams complete their allocations. The outcome of all matchups (one team's offense vs. the other team's defense) is decided using a stochastic process which favors the offense (or defense) proportionally to its strategic advantage value (with a slight defensive bias). This process repeats itself for each passageway until there are no remaining agents on one team. The remaining offensive agents from all teams advance and score one point each, whereas the remaining defensive agents score no points. After the game is played out the allocation strategies of both teams are revealed as well as the final score of the game. This short game can be played many times with a different number of passageways and strategic advantage distributions.

Obviously, each team's allocation strategy plays a big role in their eventual score. A team can choose to be more defensive and risk scoring fewer points, while allowing a smaller number of points to be scored on them. Similarly, a team may choose to allocate more agents to defend a passage where the offensive advantage is high, or they can instead re-allocate their agents to offense if they believe no opponents will be allocated to this task.

Through repeated games, a human player can learn an intuitive opponent model and hence develop a strategy they use to play the game. This strategy may be a combination of many complex factors and may be hard to explicitly articulate, much less code into an algorithm or allocation policy. However, we can use imitation learning to generalize a policy from the allocations the expert makes.

*1) Demonstrations:* The demonstrations used in this domain are much denser than those used in the disaster response scenario. They are extracted from recording the allocations by the user during game play (after a training period where they can learn about the opponent). We record how many agents were assigned to each task as offense and defense, but since our goal is to match the expert's strategy and not necessarily maximize the score (which is dependent on stochastic elements), we do not record the outcome of the particular game.

*2) Task Features:* Because the space of allocation policies in the strategic domain is much larger, we require more features to express the full range of potential policies. We use 20 features, based on the properties of each strategic task as well as its relation to the other tasks in the game (ex: the strategic advantage and its relationship to the advantages of all tasks). We also consider the current allocation properties (such as the mean and variance of the number of allocated agents per task, the total number of allocated tasks, and the percentage of offensive and defensive agents). Allocations are made sequentially and task features are computed before each auction, so initial agent allocations have an effect on future allocations.

*3) Results:* To evaluate the strategic domain we generated two strategies the adversary could pick from: a "blitz" strategy and a "sneaky" strategy. The blitz strategy allocates all agents to be on offense for the single task with the highest offensive advantage, trying to maximize the number of agents that reach the opposite base. The sneaky strategy takes a different approach: it attempts to spread out its agents on tasks with lower strategic advantages (still allocating all agents on offense). Our objective is to use imitation learning to learn policies which can effectively counter both of these by maximizing the score (more concretely: maximize points scored minus points scored on). For these trials, each game was an instance of three passageways with random strategic advantages, and two teams with 8 agents each; the maximum and minimum scores possible are thus +8 and -8 respectively.

We allowed the expert about a dozen games to learn about the opponent's strategy by playing the game manually. We then recorded the expert's allocations to use as training examples. For each opponent policy, we recorded 16 demonstrations against that adversary and ran MMP for 500 iterations. Using these learned policies, we ran 200 trials against their respective opponents. The scores of the learned policies were compared against those of a hand-tuned strategy which was generated manually. The results are summarized in Table I.

We successfully learned two policies to counter the blitz and sneaky adversaries; the blitz counter-strategy improves the mean score slightly and the minimum score considerably, representing a better worst-case allocation policy. The sneaky strategy was easier to counter (as the opponents were easier to defend against), and the mean, min, and max scores all improved dramatically.

TABLE I
RESULT AGAINST TWO OPPONENTS

| Opponent Policy | Counter-strategy | Mean | Min | Max |
|---|---|---|---|---|
| Sneaky | Best tuned | -0.075 | -5.0 | 3.0 |
| | Learned | 1.11 | -2.0 | 7.0 |
| Blitz | Best tuned | -0.28 | -4.0 | 3.0 |
| | Learned | 0.29 | -1.0 | 2.0 |

## VI. Analysis

These results show that imitation learning can be used in complex task allocation domains. The expert in the disaster response domain was able to impart an overall team strategy by picking demonstrations which they considered to be best responses given knowledge of the domain. This did not require the expert to explicitly write down a policy or hand-tune a complicated reward function. After a policy was learned, head-to-head trials showed that this learned policy was more successful at retaining overall building value.

In the strategic domain, the human player was able to learn about the opponent's strategy in a few trials, and counter it by adjusting their allocation policy. A number of demonstrations were then recorded from this counter-strategy, and a policy was learned which proved to be effective against the same opponent.

Using imitation learning (as opposed to reinforcement learning) allows us to explicitly make use of the expert's experience and knowledge about the domain, and learn a policy without an extended exploration phase (in which performance may be arbitrarily bad) or requiring extensive time or computational resources.

## VII. Conclusions and Future Work

We have successfully applied imitation learning to the problem of allocating tasks in a complex multi-robot setting. Informally, it learns a task utility function which biases prices in the market and makes the demonstrated allocations optimal. Imitation learning provides an intuitive way to utilize the expert's knowledge about the environment, while requiring only a small set of demonstrations and no manual tuning of high-dimensional reward functions.

Using imitation learning can enable us to influence the task allocation process in other ways than improving the performance of the team. When many solutions exist that are equally-valid, we can use imitation learning to impart a high-level strategy to the team, in effect performing preference elicitation. Future work will focus on this preference elicitation, and field trials involving human-robot teams, as an economy is not straightforwardly defined when robot costs and human preferences mix.

## VIII. Acknowledgments

The authors gratefully acknowledge Chris Skonieczny for his contributions to the disaster response domain. We have appreciated valuable discussions with Drew Bagnell, Geoff Gordon, David Silver, Paul Scerri, and Geoff Hollinger. This work was supported in part by ONR under MURI grant ”Reasoning in Reduced Information Spaces” (no. N00014-09-1-1052) and by the National Science Foundation under a Graduate Research Fellowship.

## References

[1] B. P. Gerkey and M. J. Mataric, “A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems,” *The International Journal of Robotics Research*, vol. 23, no. 9, September 2004.
[2] R. Zlot, A. Stentz, M. B. Dias, and S. Thayer, “Multi-robot exploration controlled by a market economy,” in *IEEE International Conference on Robotics and Automation*, 2002.
[3] B. Gerkey, S. Thrun, and G. Gordon, “Visibility-based pursuit-evasion with limited field of view,” *The International Journal of Robotics Research*, vol. 25, no. 4, pp. 299–315, 2006.
[4] N. D. Ratliff, D. Silver, and J. A. Bagnell, “Learning to search: Functional gradient techniques for imitation learning,” *Autonomous Robots*, pp. 1–36, 2009.
[5] D. Munoz, N. Vandapel, and M. Hebert, “Directional associative markov network for 3-d point cloud classification,” in *Fourth International Symposium on 3D Data Processing, Visualization and Transmission*, 2008.
[6] S. Koenig, C. Tovey, M. Lagoudakis, V. Markakis, D. Kempe, P. Keskinocak, A. Kleywegt, A. Meyerson, and S. Jain, “The power of sequential single-item auctions for agent coordination,” in *Proceedings of the National Conference on Artificial Intelligence*, vol. 21, no. 2, 2006.
[7] E. G. Jones, M. B. Dias, and A. Stentz, “Time-extended multi-robot coordination for domains with intra-path constraints,” in *Robotics: Science and Systems*, 2009.
[8] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, “A survey of robot learning from demonstration,” *Robotics and Autonomous Systems*, vol. 57, no. 5, 2009.
[9] D. A. Pomerleau, “ALVINN: An Autonomous Land Vehicle In a Neural Network,” in *Advances in Neural Information Processing Systems*, 1989.
[10] A. Coates, P. Abbeel, and A. Y. Ng, “Apprenticeship learning for helicopter control,” *Communications of the ACM*, vol. 52, no. 7, July 2009.
[11] B. Gerkey and M. Mataric, “Sold!: auction methods for multirobot coordination,” *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 758–768, October 2002.
[12] N. Kalra, D. Ferguson, and A. Stentz, “Hoplites: A Market-Based Framework for Planned Tight Coordination in Multirobot Teams,” in *International Conference on Robotics and Automation*, 2005.
[13] E. G. Jones, M. Dias, and A. Stentz, “Learning-enhanced market-based task allocation for oversubscribed domains,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007.
[14] J. Schneider, D. Apfelbaum, D. Bagnell, and R. Simmons, “Learning opportunity costs in multi-robot market based planners,” in *International Conference on Robotics and Automation*, 2005.
[15] B. Taskar, C. Guestrin, and D. Koller, “Max-margin Markov networks,” in *Advances in Neural Information Processing Systems (NIPS)*, 2003.
[16] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, “Maximum margin planning,” *International Conference on Machine learning - ICML ’06*, pp. 729–736, 2006.
[17] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz, “Market-Based Multirobot Coordination: A Survey and Analysis,” *Proceedings of the IEEE*, vol. 94, no. 7, July 2006.