Multi-Robot Coordination for Space Exploration

Logan Yliniemi[†] and **Adrian Agogino**[✠] and **Kagan Tumer**[†]
[†]Oregon State University
[✠] University of California, Santa Cruz

Abstract

Teams of artificially intelligent planetary rovers have tremendous potential for space exploration, allowing for reduced cost, increased flexibility and increased reliability. However, having these multiple autonomous devices acting simultaneously leads to a problem of coordination: to achieve the best results, the they should work together. This is not a simple task. Due to the large distances and harsh environments, a rover must be able to perform a wide variety of tasks with a wide variety of potential teammates in uncertain and unsafe environments. Directly coding all the necessary rules that can reliably handle all of this coordination and uncertainty is problematic. Instead, this article examines tackling this problem through the use of coordinated reinforcement learning: rather than being programmed what to do, the rovers iteratively learn through trial and error to take take actions that lead to high overall system return. To allow for coordination, yet allow each agent to learn and act independently, we employ state-of-the-art reward shaping techniques. The article uses visualization techniques to break down complex performance indicators into an accessible form, and identifies key future research directions.

Introduction

Imagine for a moment that you're tasked with teleoperating (controlling with a joystick) a Mars rover as it navigates across the surface. You watch the feed from the on-board camera as the rover rolls along the surface, when you notice the terrain changing ahead, so you instruct the rover to turn. The problem? You're 6 minutes too late. Due to the speed-of-light delay in communication between yourself and the rover, your "monolithic" multi-million dollar project is in pieces at the bottom of a Martian canyon, and the nearest repairman is 65 million miles away (NAS 2003).

There are, of course, solutions to this type of problem. You can instruct it to travel a very small distance and re-evaluate the rover's situation before the next round of travel, but this leads to painfully slow processes that take orders of magnitude longer than they would on earth. The speed of light is slow enough that it hinders any attempts at interacting regularly with a rover on another planet.

But what if, instead of attempting to control every aspect of the rover's operation, we were able to take a step back and simply tell the rover what we're trying to find, and have it report back when it finds something we'll think is interesting? Giving the rover this type of autonomy removes the need for constant interaction, and makes the speed of light a moot point.

Hard-coding a procedure for handling all of the cases a rover could encounter while navigating — and the thousands of other tasks that a rover might have to undertake — is not a good option in these cases. The need for flexibility is key, and the onboard storage space is typically quite limited. Due to the large distances, communication lag, and changing mission parameters, any efforts in space exploration need to be extremely robust to a wide array of possible disturbances, and capable of a wide array of tasks. In short, as the human race expands its efforts to explore the solar system, artificial intelligence will play a key role in many high-level control decisions.

However, giving a rover that cost many man-years of labor and a multi-million dollar budget complete autonomy over its actions on another planet might be a bit unnerving. Space is a harsh and dangerous place; what if it isn't able to achieve the tasks it needs to? Worse, what if the rover finds an unpredicted and creative way to fail? These are legitimate concerns, worth addressing seriously.

One way to mitigate these concerns is to take the concept of a single traditional monolithic rover and broke it up into many pieces, creating a team of rovers, with one to embody each of these pieces. Each would be simple, and perform

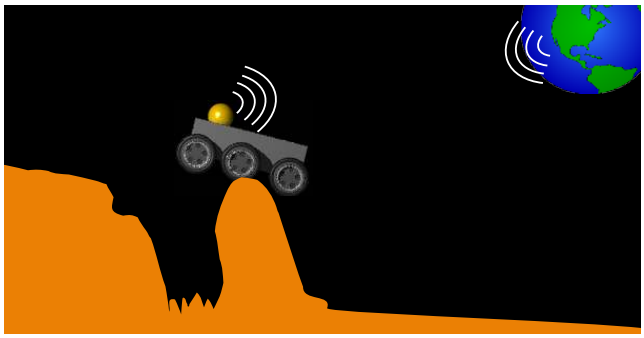

Figure 1: The speed-of-light communication delay makes artificial intelligence a necessity for space exploration.

just a few functions. Though each of the pieces is less effective individually than the monolithic rover, the sum of the pieces is greater than the whole in many ways.

First, any of the members of the team is significantly more expendable than the whole monolithic rover. This alleviates a large number of concerns, and opens many opportunities. If one rover does find a way to fail creatively, the remainder of the team is still completely operational. By the same token, the team of rovers can undertake more dangerous missions than the monolithic rover; if the dangerous conditions lead to the failure of one rover, the rest can complete the mission. Additionally, redundancy can be designed into the team for particularly dangerous or critical roles.

Beyond the disposability of the individual team members, there are other benefits to this team-based approach. Savings can be realized in construction, as each rover can be designed with parts from a lower-cost parts portion of the reliability curve. Similar savings are available in the design process, as a new team can be formed with some members that have been previously designed.

In addition, a team of rovers can have capabilities that a single monolithic rover cannot, like having presence in multiple locations at once, which is incredibly useful for planetary exploration. Ephemeral events can be simultaneously observed from separate locations (Estlin and et al 2010), even from the ground and from orbit simultaneously (Chien and et al 2011), which can make interpreting the situation significantly easier. Construction tasks that might be impossible for a single rover with limited degrees of freedom become much easier. Teams can survey areas separated by impassible terrain and share long-range communication resources (Chien et al. 2000).

However, the concerns that we must address expand rapidly once we start to consider the possibilities that arise with multiple rovers acting in the same area simultaneously. How do the rovers coordinate so that their efforts lead to the maximum amount of interesting discoveries? How does a rover decide between achieving a task on its own versus helping another rover that has become stuck? How does it decide between covering an area that's been deemed interesting, or exploring an area that hasn't received much attention? These are all issues that fall under the larger umbrella of multiagent artificial intelligence (or multiagent systems), which is a ripe area of modern research (Wooldridge 2008).

One technique that has proven useful within the multiagent systems community is that of "reward shaping" used in conjunction with reinforcement learning. In this paradigm, instead of the rovers being told what to do, they each individually "learn" what to do through an iterative process of trial and error [1]. In this process, each rover learns to maximize a reward function, measuring its performance. By carefully shaping the rewards that the rovers receive, we can promote coordination and improve the robustness of the learning process (Mataric 1994; Taylor and Stone 2009). Our goals in reward shaping are to balance two fundamental tensions in learning: 1) The rewards that the rovers are maximizing should be informative enough that they can promote coordination of the entire system, and 2) They should be simple enough that the rovers can easily determine the best actions to take to maximize their rewards. There are a number of obstacles that can make achieving this goal more difficult.

[1]Note the form learning used in this article is closely related to direct policy search and evolutionary algorithms. Shaped rewards and shaped evaluation functions work similarly between these methods.

Multiagent coordination is hard

Being able to automatically learn intelligent control policies for autonomous systems is an exciting prospect for space exploration. Especially within the context of a coordinated set of autonomous systems, we have the possibility of achieving increased capabilities while maintaining an adaptive and robust system. However, these "multiagent" systems are fundamentally different from other types of artificial intelligence in two ways: 1) We have to promote coordination in a multiagent system (See Figure 2), since agents learning by themselves may work at cross-purposes, and 2) We have to overcome increased learning complexity as the actions taken by other agents increase the difficulty that any particular agent has in determining the value of its actions with respect to a coordinated goal.

In space applications, this coordination will involve many issues like optimizing communication networks, maximizing scientific information returned from a set of sensors and coordinating power usage through shared power resources. As a guiding example, consider a group of autonomous rover agents set to explore an area of an extraterrestrial body. Their goal is to observe a series of points of interest, and gain as much knowledge about these points as possible on a team-wide level. This means that ideally each agent within the multiagent system will cooperate toward the common good, but how to do this is not immediately obvious. For example, it may not be readily apparent in practice that a rover

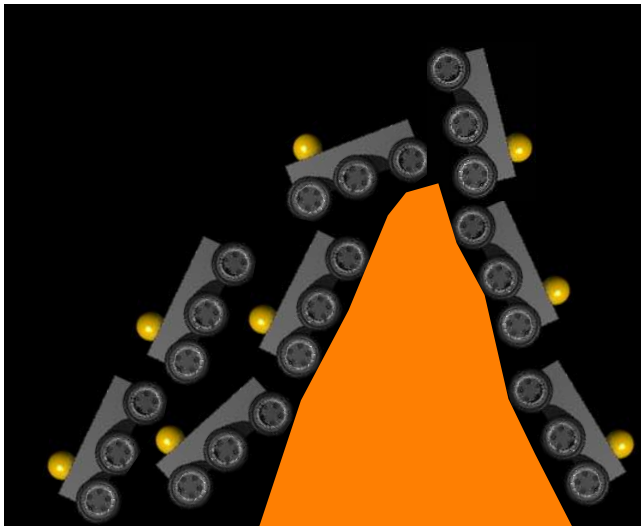

Figure 2: Effective single agent learning may lead to incompatible interactions in a multiagent setting. Repetitive exploration and congestion are common problems.

is actively observing a point that has been well-studied at an earlier point in time. The rover's actions of observing that point may be a very good choice, except that the other agents acting in the environment had already gleaned the necessary information from the point, making the action redundant.

Complex communication protocols or teamwork frameworks may offer a solution to this problem, but it might not be a practical one for space travel. Communication availability is limited, and failures of existing rovers or introduction of new rovers that weren't originally planned into the team are a realistic expectation for space exploration (Stone et al. 2013). Because of the large travel times and distances, and unpredictable and harsh environments, flexibility in implementation is key, and the solution must be robust to all sorts of disturbances.

This flexibility can be developed through the use of adaptive agent policies, which change over time to fit the situation the rover encounters. This creates a learning multiagent system, which allows the team to effectively deal with changing environments or mission parameters. A key issue in a learning multiagent system is the choice of the reward function that the agents use.

How to judge a reward function

A multiagent learning system depends on a way to measure the value of each agent's behavior. For instance, did a particular sensor reading give additional scientific value? Did a particular message sent efficiently use the communications channel? Did a particular rover movement put the rover in a good location and not interfere with the actions of another rover? This measurement is called a reward function, and changing what form the reward function takes is the science of "reward shaping" (Chalkiadakis and Boutilier 2003; Guestrin, Lagoudakis, and Parr 2002; Hu and Wellman 1998; Mataric 1998; Stone and Veloso 2000; Tumer, Agogino, and Wolpert 2002; Wolpert and Tumer 2001). An agent will seek to solely increase its reward function, forsaking all other concerns, so it is important that it has two specific properties.

First, the reward function must be "sensitive" to the actions of the agent (Wolpert and Tumer 2001). An agent taking good actions should receive a high reward, and an agent taking poor actions should receive a lower reward. In an unpredictable, stochastic, or multiagent environment, there are other factors affecting the reward that the agent will receive. An ill-developed reward function will allow these random factors to insert a large amount of "noise" into the "signal" offered by the reward function, and as the signal-to-noise ratio decreases, so does the agent's performance.

Second, the reward function must be "aligned" with the overall mission that the agent team must achieve (Wolpert and Tumer 2001). That is, an agent that increases its own reward should simultaneously be increasing the system performance. A lack of alignment can lead to situations such as the Tragedy of the Commons (Hardin 1968; Crowe 1969), wherein a group of rationally self-concerned agents lead to a drop in system performance due to working at cross-purposes. That is, agent A does what it perceives in its own best interest, as does agent B; in some way, their actions deplete their shared environment, and lead to both agents being worse off than they would be had they cooperated for the communal good.

Both of these properties — sensitivity and alignment — are critical to multiagent systems. An agent must be able to clearly discern what it has done to earn a high reward, and continuing to earn that high reward must be in the best interest of the system as a whole. This is especially the case in space applications, because the large distances and communication restrictions introduced by limited bandwidth, limited power, or line of sight lead to time prevents outside intervention if the system performance were to go awry. In fact, even identifying a that a problem exists within the system is challenging: space and extra planetary exploration is a complex and difficult problem, and it might not be easy to immediately diagnose when agents aren't achieving their full potential.

In this work we show one approach to diagnosing potential system performance issues through visualizing the sensitivity and alignment of various reward structures in a simple and straightforward manner.

Classic approaches to coordination via reward shaping

There are three classic approaches to solving complex multiagent systems. Each has specific advantages and drawbacks.

Robot totalitarianism (centralized control) First, consider a centralized system in which one agent is making all necessary decisions for the entire system as a whole, and all other agents are merely following orders. The advantages here are that perfect coordination is possible, and the pieces of the system as a whole will cooperate to increase system performance. This typically works well for small systems consisting of just a few agents (Sutton and Barto 1998). However, such a centralized system can fall prey to complexities such as communication restrictions, component failures — especially where a single point of failure can stop the entire system — and simply the difficulty of simultaneously solving a problem for hundreds or thousands of agents simultaneously. In most realistic situations, this is simply not an option.

Robot socialism (global or team reward) Next, consider a system in which each agent is allowed to act autonomously in the way that they see fit, and every agent is given the identically same *global* reward, which represents the system performance as a whole. They will single-mindedly pursue improvements on this reward, which means that their efforts are directed toward improving system performance, due to this reward having perfect *alignment*. However, because there may be hundreds or thousands of agents acting simultaneously in the shared environment, it may not be clear what led to the reward. In a completely linear system of n agents, each agent is only responsible for $1/n$ of the reward that they all receive, which can be entirely drowned out by the $(n-1)/n$ portion for which that agent is not responsible. In a system with 100 agents, that means an agent might only

have dominion over 1% of the reward it receives! This could lead to situations in which an agent chooses to do nothing, but the system reward increases, because other agents found good actions to take. This would encourage that agent to continue doing nothing, even though this hurts the system, due to a lack of *sensitivity* of the reward.

Robot capitalism (local or perfectly learnable reward) Finally, consider a system in which each agent has a *local* reward function related to how productive it is, itself. For example, a planetary rover could be evaluated on how many photographs it captures of interesting rocks. This means that its reward is dependent only on itself, creating high *sensitivity*. However, the team of rovers obtaining hundreds of photographs of the same rock is not as interesting as obtaining hundreds of photographs of different rocks, though these would be evaluated the same with a local scheme. This means that the local reward is not *aligned* with the system level reward.

Summary Each of the reward functions has benefits and drawbacks that are closely mirrored in human systems. However, we are not limited to just these reward functions; as we mentioned before, an agent will single-mindedly seek to increase its reward, no matter what it is, whether or not this is in the best interest of the system at large. Is there, perhaps, a method that could be as *aligned* as the global reward, while as *sensitive* as the local reward, while still avoiding the pitfalls of the centralized approach?

Difference rewards

An ideal solution would be to create a reward that is aligned with the system reward while removing the noise associated with other agents acting in the system. This would lead agents toward "doing everything they can to improve the system's performance". Such a reward in a multi rover system would reward a rover for taking a good action that coordinates well with rovers that are close to it, and would ignore the effects of distant rovers that were irrelevant.

A way to represent this analytically is to take the global reward $G(z)$ of the world z, and subtract off everything that doesn't have to do with the agent we're evaluating, revealing how much of a difference the agent made to the overall system. This takes the form

$$D_i(z) = G(z) - G(z_{-i}) \tag{1}$$

where $G(z_{-i})$ is the global reward of the world without the contributions of agent i, and $D_i(z)$ is the difference reward.

Let us first consider the alignment of this reward. $G(z)$ is perfectly aligned with the system reward. $G(z_{-i})$ may or may not be aligned, but in this case, it doesn't matter, because agent i (whom we are evaluating) has no impact on $G(z_{-i})$, by definition. This means that $D_i(z)$ is perfectly aligned, because all parts that agent i affects are aligned: agent i taking action to improve $D_i(z)$ will simultaneously improve $G(z)$.

Now, let us consider the sensitivity of this reward. $G(z)$ is as sensitive as the system reward, because it is identical. However, we remove $G(z_{-i})$ from the equation; that is, a large portion of the system — on which agent i has no impact on the performance — does not impact $D_i(z)$. This means that $D_i(z)$ is very sensitive to the actions of agent i, and includes little noise from the actions of other agents.

Difference rewards are not a miracle cure. They do require additional computation to determine which portions of the system reward are caused by each agent. However, it is important to note that it is not necessary to analytically compute these contributions. In many cases, a simple approximation that serves to remove a large portion of the noise caused by using the system-level reward gains significant performance increases over using the system reward alone.

Although in this paper we focus on the continuous rover domain, both the difference reward and the visualization approach have broad applicability. The difference reward used in this paper has been applied to many domains, including data routing over a telecommunication network (Tumer and Wolpert 2000), multiagent gridworld (Tumer, Agogino, and Wolpert 2002), congestion games such as traffic toll lanes (Tumer and Wolpert 2004a; 2004b; Wolpert and Tumer 2001) and optimization problems such as bin packing (Wolpert, Tumer, and Bandari 2004) and faulty device selection (Tumer 2005).

Continuous rover domain

To examine the properties of the difference reward in a more practical way, let us return to our example of a team of rovers on a mission to explore an extraterrestrial body, like the moon or Mars. We allow each rover to take continuous actions to move in the space, while receiving noisy sensor data at discrete time steps (Agogino and Tumer 2004).

Points of interest (POIs) Certain points in the team's area of operation have been identified as points of interest (POIs), which we represent as green dots. Figure 4 offers one of the layouts of POIs that we studied, with a series of lower-valued POIs located to the left on the rectangular world, and a single high-valued POI located on the right half. Because multiple simultaneous observations of the same POI are not valued higher than a single observation in this domain, the best policy for the team is to spread out: one agent will

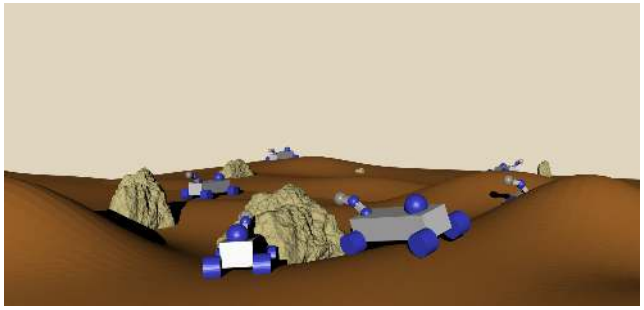

Figure 3: Artist's rendition of a team of rovers exploring various points of interest on the Martian surface.

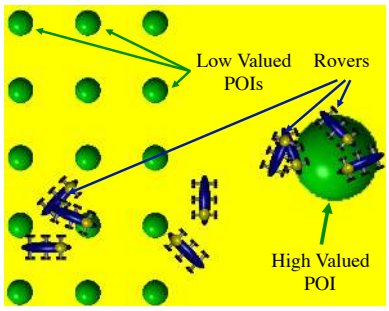

Figure 4: A team of rovers observing a set of points of interest (POIs). Each POI has a value, represented by its size here. The team will ideally send one rover to observe the large POI on the right closely, while the rest spread out in the left region to observe as many small POIs as possible.

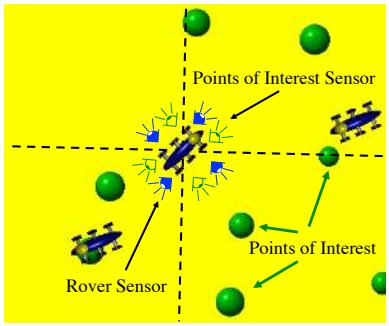

Figure 5: Rover sensing diagram. Each rover has 8 sensors: four rover sensors and four POI sensors that detect the relative congestion of each in each of the four quadrants that rotate with the rover as it moves.

closely study the large POI, while the remainder of the team will cover the smaller POIs on the other side.

Sensor model We assume that the rovers have the ability to sense the whole domain (except in the results we present later marked with PO for partial observability), but even so, using state variables to represent each of the rovers and POIs individually results in an intractable learning problem: there are simply too many parameters. This is also why a centralized controller does not function well in this case. We reduce the state space by providing 8 inputs through the process illustrated in Figure 5. For each quadrant, which rotates to remain aligned with the rover as it moves through the space, the rover has a 'rover sensor' and a 'POI sensor'. The rover sensor calculates the relative density and proximity of rovers within that quadrant and condenses this to a single value. The POI sensor does the same for all POIs within the quadrant.

Motion model We model the continuous motion of the rovers at each finite timestep as shown in Figure 6. We maintain the current heading of each rover, and at each timestep the rovers select a value for dy and dx, where the value of dy

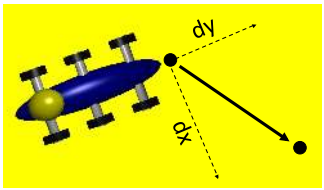

Figure 6: Rover motion model. At each timestep, each rover determines a continuous dy value to represent how far it moves in the direction it is facing, and a dx value determining how far it turns. Its heading at the next timestep is the same as the vector $dx + dy$.

represents how far forward the rover will move, and dx represents how much the rover will turn at that timestep. The rover's heading for the next timestep is represented as the direction of the resultant vector $(dx + dy)$, shown as the solid line in Figure 6.

Policy search The rovers use multi-layer perceptrons (MLP) with sigmoid activation functions to map the eight inputs provided by the four POI sensors and four rover sensors through ten hidden units to two outputs, dx and dy, which govern the motion of the rover. The weights associated with the MLP are established through an online simulated annealing algorithm that changes the weights with preset probabilities (Kirkpatrick, Gelatt, and Vecchi 1983). This is a form of direct policy search, where the MLPs are the policies.

Reward structures

We present the visualizations for alignment and sensitivity of four reward structures in this work:

- The perfectly learnable local reward, P_i, is calculated by considering the value of observations of all POIs made by agent i throughout the course of the simulation, ignoring the contributions that any other agents had to the system.
- The global team reward, T_i, is calculated by considering the best observation the team as a whole made during the course of the simulation.
- The difference reward, D_i, is calculated similarly to the perfectly learnable reward P_i, with the exception that if a second agent j also observed the POI, agent i is only rewarded with the difference between the quality of observations. Thus, if two agents observe a POI equally well, it adds to neither of their rewards, because the team would have observed it anyway. If an agent is the sole observer of a POI, they gain the full value of the POI observation.
- The difference reward under partial observability, $D_i(PO)$, is calculated in the same manner as D_i, but with restrictions on what agent i can observe. Each rover evaluates itself in the same way as D_i, but because of the partial observability, it is possible that two rovers will be observing the same POI from opposite sides, and neither will realize that the POI is doubly observed (which does not increase the system performance), and both will credit

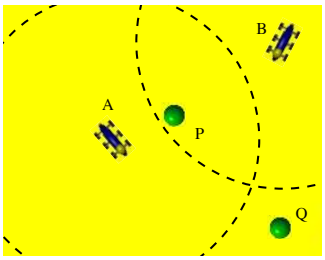

Figure 7: Rovers under partial observability of range denoted by the dotted line. Both rover A and rover B can sense and observe POI P, but cannot sense each other. In the $D_i(PO)$ formulation, they both would calculate that theirs was the only observation. Additionally, neither rover has any knowledge of POI Q.

themselves. Likewise, each rover cannot sense POIs located outside of its observation radius. This is represented in Figure 7.

Visualization of reward structures

Visualization is an important part of understanding the inner workings of many systems, but particularly those of learning systems (Agogino, Martin, and Ghosh 1999; Bishof, Pinz, and Kropatsch 1992; Gallagher and Downs 1997; Hinton 1986; Hoen and G. Redekar 2004; Wejchert and Tesauro 1991). Especially in costly space systems we need additional validation that our learning systems are likely to work. Performance simulations can give us good performance bounds in scenarios that we can anticipate ahead of time. However, these simulations may not uniformly test the rovers in all situations that they may encounter. Learning and adaptation can allow rovers to adapt to unanticipated scenarios, but their reward functions still have to have high sensitivity and alignment to work. The visualization presented here can give us greater insight into the behavior of our reward functions. Our visualizations can answer important questions such as how often we think our reward will be aligned with our overall goals and how sensitive our rewards are to a rover's actions. Through visual inspection we can see if there are important gaps in our coverage, and we can increase our confidence that a given reward system will work reliably.

The majority of the results presented in this work show the relative sensitivity and alignment of each of the reward structures. We have developed a unique method for visualizing these, which is illustrated in Figure 8. We use the sensor information from the rover (left) to determine which of the spaces we will update (right). The alignment or sensitivity calculation (Agogino and Tumer 2008) is then represented by a symbol that takes the form of a '+' or '-' sign; the brighter the shade of the spot, the further from the average. A bright '+', then, represents a very aligned or very sensitive reward and a bright '-' represents an anti-aligned or very non-sensitive reward for a given POI and rover density, in the case of Figure 9. We also present these calculations projected onto a specific case of the actual space that the

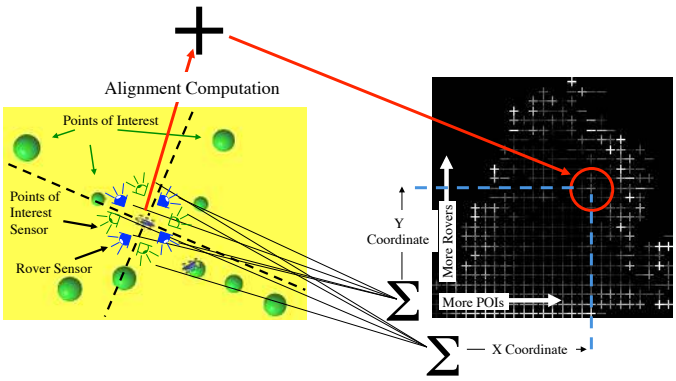

Figure 8: Illustration of the visualization calculation process. We use sensor data to determine which spot in the state space a circumstance represents, and place a marker in that location that represents whether the reward scores highly (bright +), near random (blank) or lowly (bright −).

rovers move through in Figure 10. A more general version of this technique projects onto the principal components of the state space, which is more thoroughly explored in other work (Agogino and Tumer 2008).

Sensitivity and alignment analysis

A reward with simultaneously high alignment and sensitivity will be the easiest for agents to use to establish high-performing policies. Figure 9 presents the visualization for each of the reward structures. Notice that the perfectly learnable reward P_i does indeed have high sensitivity across the space, but has low alignment with the global reward in most of the center areas, which correspond to a moderate concentration of rovers and POIs. This area near the center of the visualization represents circumstances that the rovers find themselves in most often (Agogino and Tumer 2008).

The team reward T_i, on the other hand, is very aligned throughout the search space, but is extremely lacking in sensitivity (denoted by the many '-' signs throughout the space).

The difference reward D_i is both highly aligned and highly sensitive throughout the search space. When we reduce the radius at which D_i can sense other rovers and POIs, the visualization from the $D_i(PO)$ row indicate that the sensitivity remains strong everywhere, but there is a slight drop in alignment throughout the space.

So, it would appear that difference rewards (D_i) offer benefits over other rewards, even with partial observability ($D_i(PO)$), but what does this mean in a more practical sense? To address this, we created Figure 10, which projects the same type of alignment into the actual plane in which the rovers are operating.

The left figure presents the alignment for the perfectly learnable reward P_i, and the indicated region is anti-aligned with the system-level reward. That is, even though traveling across this region would be beneficial to the team (because traveling across this region is required to reach the large POI on the right), the rovers that find themselves in this area of the space are actively penalized.

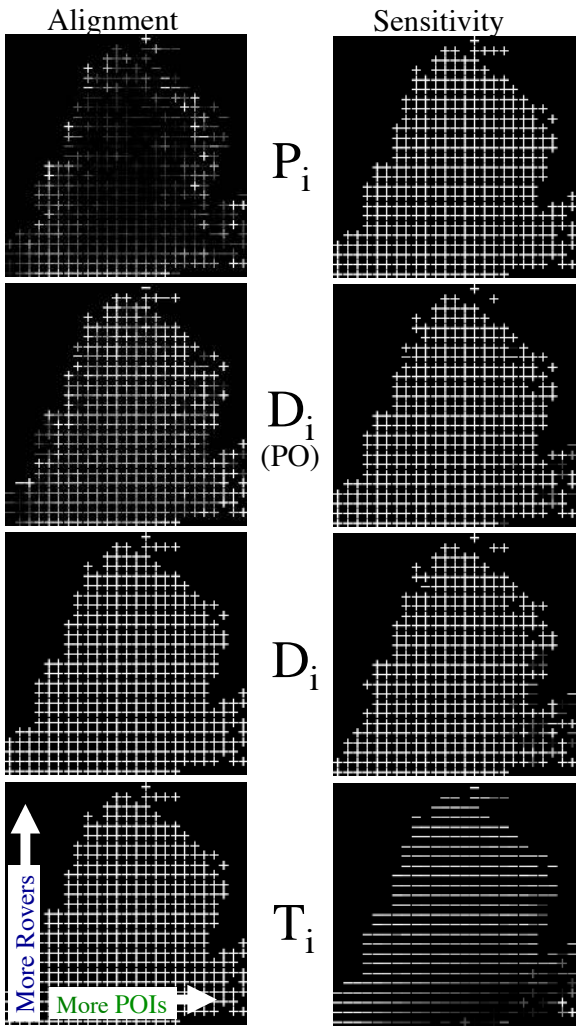

Figure 9: Alignment and sensitivity visualization for the four reward types, projected onto a 2-dimensional space representative of the state space. Note that the perfectly learnable reward P_i has low factoredness through most of the space, and the team reward T_i is extremely non-sensitive through most of the space, while both instances of the difference reward maintain high performance by both metrics.

The figure on the right presents the alignment for the difference reward under observation restrictions $D_i(PO)$, which is qualitatively different within the highlighted regions: $D_i(PO)$ builds two "aligned bridges", which allow the rovers to pass through the highlighted region without being penalized while they travel to the large POI on the right. Furthermore, the other parts of the highlighted region are not anti-aligned with the system reward meaning that the rovers are not penalized for traveling through this space, they merely do not increase their reward while there.

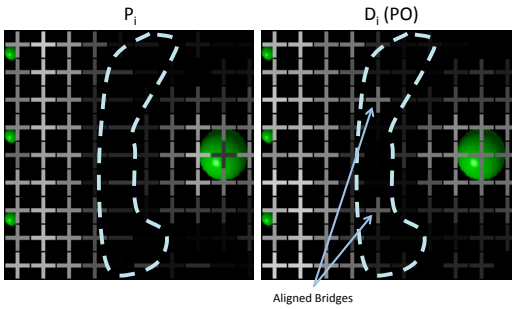

Figure 10: Alignment visualization for the perfectly learnable reward P_i, and the difference reward under partial observability, $D_i(PO)$, projected onto the actual plane the rovers operate within.

System performance

We present system-level performance in Figure 11, which represents the final system reward after training (y-axis) for teams of rovers trained on various rewards (line type), within different experiments with varying restrictions on observation radius (x-axis). Points to the left represent performance under extreme observation restrictions, and points to the right represent near-full observability. The visualizations performed in Figures 9 – 10 correspond to full observability for all rewards except $D_i(PO)$, which corresponds to the D_i reward at a communication radius of 10 units in Figure 11.

The benefits in sensitivity and alignment offered by the difference rewards D_i does result in increased system performance, as shown by the rightmost portion of Figure 11. This reward leads to high-performing systems of rover agents with very successful policies. The global shared team reward T_i is capable of making some increases over a local policy under full observability, but still falls short the difference reward.

The remainder of Figure 11 presents a result based on the final system performance attained by agent teams operating with different rewards under restricted communications. Agents trained on the difference reward D_i are robust to a reduced communication radius, which could easily happen in cases of a dust storm, craggy landscape, or partial sensor failures. Agents using the perfectly learnable reward P_i are not affected by these restrictions, as the actions of other agents don't affect their policies.

Agents trained on the team or global reward T_i show an interesting phenomenon, however. Agents operating with a large communication radius are able to perform well as a team, and as this communication radius is reduced, so is the quality of the discovered policies — this much is expected. However, as the observation radius is decreased further, experimental runs with very low observation radii actually perform slightly better than those with moderate observation powers. This suggests that a little bit of knowledge about the location of other rovers is actually a bad thing. This can be explained: as the observation radius is reduced, agents trained on the team reward will behave more selfishly, like rovers using P_i, simply because they cannot sense the other

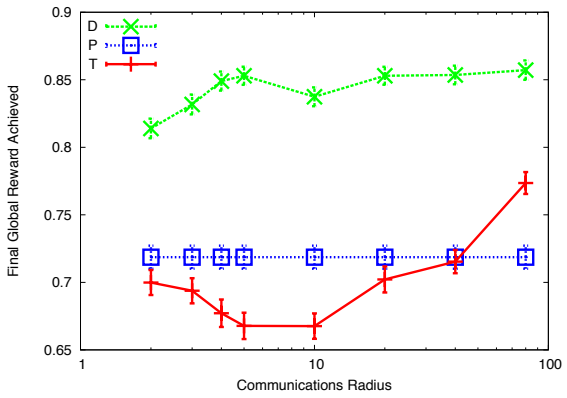

Figure 11: Final performance attained vs. communication radius for the different reward structures. Difference rewards maintain robust performance, but team rewards lose significant performance under restricted communication.

rovers in the area, thus the gap between their performance decreases as the restrictions mirror this case.

Conclusions

Space exploration creates a unique set of challenges that must be addressed as we continue our expanding our reach in the solar system. One approach for dealing with these challenges is through the use of reinforcement learning with reward shaping. Care must be taken in any use of reward shaping: a solution that works with a small number of agents will not necessarily scale up in an expected fashion, and might lead to catastrophic system-level results. The readily obvious team reward and perfectly learnable reward both lead to poor results due to their low sensitivity and alignment, respectively. There is a need for local-level rewards that can be carried out quickly and efficiently that will scale into favorable results at the broader system level.

Difference rewards are an effective tool for this by encouraging multiagent coordination by their guaranteed alignment with the system objective, as well as their high sensitivity to local actions. They maintain high learnability throughout the state space, while offering perfect alignment with the system-level reward. This results in benefits that can be readily visualized within the space in which a team of rovers works, creating "bridges of high reward" that rovers can cross in between sparse POIs, and increasing overall system performance over a personal or team-based reward.

These properties in tandem with the robustness to various types of change within the environment show that their use in space exploration applications is an ideal fit. The capability of using a difference reward to encourage agents to do their best to help the team at whatever task is assigned allows for a team that can quickly and deftly adjust when mission parameters change. This can be as mundane as a sensor failing, or as dramatic as a complete mission reassignment.

While developing more sophisticated technologies for sensing more about the environment in a more efficient manner is a useful step forward, for multiagent space exploration, the key problem remains as "What should the agents do to work together?". This persists as a fertile motivating question for future research.

References

Agogino, A., and Tumer, K. 2004. Efficient evaluation functions for multi-rover systems. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2004)*, 1–12.

Agogino, A. K., and Tumer, K. 2008. Analyzing and visualizing multiagent rewards in dynamic and stochastic environments. *Journal of Autonomous Agents and Multi-Agent Systems* 17(2):320–338.

Agogino, A.; Martin, C.; and Ghosh, J. 1999. Visualization of radial basis function networks. In *Proceedings of International Joint Conference on Neural Networks*.

Bishof, H.; Pinz, A.; and Kropatsch, W. G. 1992. Visualization methods for neural networks. In *11th International Conference on Pattern Recognition*, 581–585.

Chalkiadakis, G., and Boutilier, C. 2003. Coordination in multiagent reinforcement learning: A bayesian approach. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-03)*.

Chien, S., and et al, J. D. 2011. Combining space-based and in-situ measurements to track flooding in thailand. *Geoscience and Remote Sensing*.

Chien, S.; Barrett, A.; Estlin, T.; and Rabideau, G. 2000. A comparison of coordinated planning methods for cooperating rovers. *International Conference on Autonomous Agents*.

Crowe, B. L. 1969. The tragedy of the commons revisited. *Science* 166:1103–1107.

Estlin, T., and et al, S. C. 2010. Coordinating multiple spacecraft in joint science campaigns. *i-SAIRAS*.

Gallagher, M., and Downs, T. 1997. Visualization of learning in neural networks using principal component analysis. In *International Conference on Computational Intelligence and Multimedia Applications*, 327–331.

Guestrin, C.; Lagoudakis, M.; and Parr, R. 2002. Coordinated reinforcement learning. In *Proceedings of the 19th International Conference on Machine Learning*.

Hardin, G. 1968. The tragedy of the commons. *Science* 162:1243–1248.

Hinton, G. 1986. Connectionist learning procedures. *Artificial Intelligence* 40:185–234.

Hoen, P., and G. Redekar, V. Robu, H. L. P. 2004. Simulation and visualization of a market-based model for logistics management in transportation. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 1218–1219.

Hu, J., and Wellman, M. P. 1998. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *Proceedings of the Fifteenth International Conference on Machine Learning*, 242–250.

Kirkpatrick, S.; Gelatt, C. D. J.; and Vecchi, M. P. 1983. Optimization by simulated annealing. *Science* 220:671–680.

Mataric, M. J. 1994. Reward functions for accelerated learning. In *Machine Learning: Proceedings of the Eleventh International Conference*, 181–189.

Mataric, M. J. 1998. Coordination and learning in multi-robot systems. In *IEEE Intelligent Systems*, 6–8.

NASA. 2003. *Mars Exploration Rover Launches*.

Stone, P., and Veloso, M. 2000. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots* 8(3):345–383.

Stone, P.; Kaminka, G. A.; Kraus, S.; Rosenschein, J. R.; and Agmon, N. 2013. Teaching and leading an ad hoc teammate: Collaboration without pre-coordination. *Artificial Intelligence*.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Taylor, M. E., and Stone, P. 2009. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*.

Tumer, K.; Agogino, A.; and Wolpert, D. 2002. Learning sequences of actions in collectives of autonomous agents. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 378–385.

Tumer, K., and Wolpert, D. H. 2000. Collective intelligence and Braess' paradox. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, 104–109.

Tumer, K., and Wolpert, D., eds. 2004a. *Collectives and the Design of Complex Systems*. New York: Springer.

Tumer, K., and Wolpert, D. 2004b. A survey of collectives. In *Collectives and the Design of Complex Systems*. Springer. 1,42.

Tumer, K. 2005. Designing agent utilities for coordinated, scalable and robust multi-agent systems. In Scerri, P.; Mailler, R.; and Vincent, R., eds., *Challenges in the Coordination of Large Scale Multiagent Systems*. Springer.

Wejchert, J., and Tesauro, G. 1991. Visualizing processes in neural networks. *IBM Journal of Research and Development* 35:244–253.

Wolpert, D. H., and Tumer, K. 2001. Optimal payoff functions for members of collectives. *Advances in Complex Systems* 4(2/3):265–279.

Wolpert, D. H.; Tumer, K.; and Bandari, E. 2004. Improving search algorithms by using intelligent coordinates. *Physical Review E* 69:017701.

Wooldridge, M. 2008. *An introduction to multiagent systems*. Wiley.

Logan Yliniemi is a graduate research assistant at Oregon State University. He is pursuing his Ph.D. in Robotics. His research focuses on credit assignment in multiagent systems and the interactions between multiagent systems and multi-objective problems.

Dr. Adrian Agogino is a researcher with the Robust Software Engineering Group in the Intelligent Systems Division at NASA Ames Research Center, employed through the University of California, Santa Cruz. His research interests are in the fields of machine learning, complex learning systems and multiagent control.

Dr. Kagan Tumer is a professor of robotics and control at Oregon State University. His research interests are control and optimization in large autonomous systems with a particular emphasis on multiagent coordination. Applications of his work include coordinating multiple robots, optimizing large sensor networks, controlling unmanned aerial vehicles, reducing traffic congestion and managing air traffic.