Coactive Learning with a Human Expert for Robotic Information Gathering

Thane Somers and Geoffrey A. Hollinger

***Abstract*— We present a coactive algorithm for learning a human expert's preferences in planning trajectories for information gathering in scientific autonomy domains. The algorithm learns these preferences by iteratively presenting solutions to the expert and updating an estimated utility function based on the expert's improvements. We apply these algorithms, in the context of underwater data collection, using a pair of risk and reward maps. In simulated trials, the algorithm successfully learns the underlying weighting behind a utility map used by a human planning trajectories. We also present experimental trials demonstrating the algorithm using a temperature and depth monitoring task in an inland lake with an autonomous surface vehicle. This work shows it is possible to design algorithms for autonomous navigation with reward functions that capture the essence of a human's preferences.**

I. Introduction

When robotic vehicles collaborate with humans, true autonomy relies on the robot having a clear understanding of its goals and the tradeoffs it faces when making decisions. When a robot is assisting a human, the robot's goals must often mimic those of the human. One example of this is in planning trajectories for underwater robots performing scientific monitoring. The robot must autonomously navigate the environment while maintaining the same goals as a human scientist.

When planning trajectories for underwater gliders during such robotic data collection tasks [6, 17], a scientist implicitly balances several environmental variables, such as risk of collision (seen in Fig. 1), uncertainty in ocean currents, and the location of points of interest. While current planning algorithms can account for all of these variables, it is difficult to learn the correct tradeoffs between them [14]. In this work, we study applying a coactive learning algorithm to learn a human path planner's weighting of the variables involved in choosing a trajectory. In this way, we can create an autonomous system that generalizes to different problems while still capturing the scientist's expert knowledge and experience.

In this paper, we adapt the coactive learning algorithm [13] to learn a human expert's preferences in the domain of robotic path planning, modeled specifically after planning for underwater scientific data collection. The algorithm attempts to learn the expert's judgment of the utility of a set of paths. This learned utility function can then be used to create a path mimicing that which the human would have planned.

T. Somers and G. Hollinger are with the School of Mechanical, Industrial & Manufacturing Engineering, Oregon State University, Corvallis, OR 97330 USA, (e-mail: {somersth,geoff.hollinger}@oregonstate.edu).

This work was supported in part by the following grants: NSF IIS-1317815 and ONR N00014-14-1-0905.

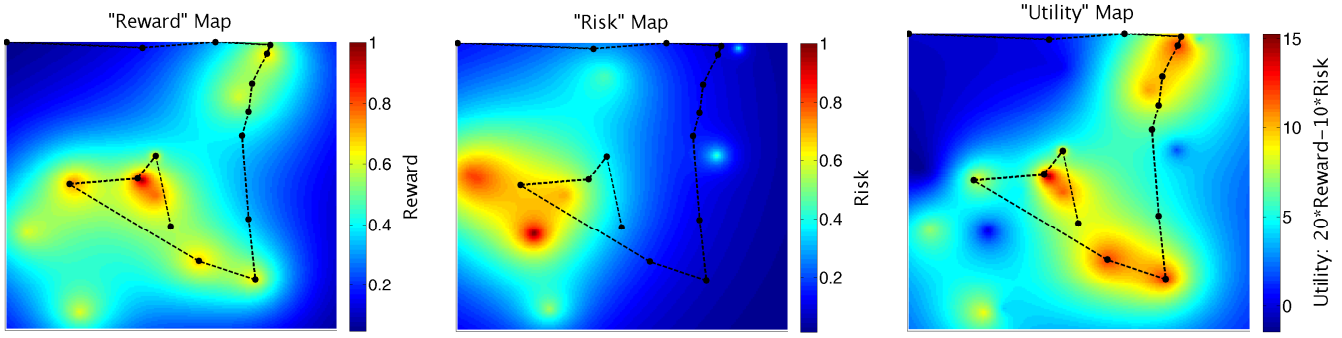
Fig. 1: An autonomous underwater glider [18] that was damaged by ship traffic. These kinds of accidents motivate the need for a better risk-aware path planners informed by human scientists.

We present a novel addition to the coactive learning algorithm to account for the variability in the quality of solutions provided by the human. This allows the algorithm to reliably learn the human's preferences in 10-15 updates. We then perform field trials to demonstrate that this makes the algorithm robust and efficient enough for a human planner to easily use in real time.

The remainder of this paper is organized as follows. We begin by discussing related work in the fields of coactive learning, human-robot interaction, and learning by demonstration (Section II). We then present the coactive learning algorithm and the modifications needed to allow it to accurately capture a human's preferences (Section III). The outcomes of human-in-the-loop simulations run on each algorithm are presented to show the benefit of the proposed approach (Section IV). We then present the results of field trials done using our algorithm on an Autonomous Surface Vehicle (Section V). Finally we conclude with a discussion of the applications of this research and possible future directions for study.

II. Related Work

Much of the previous work on solving the problem of allowing humans to teach robots has focused on finding ways for the robot to effectively mimic the human. Researchers have studied a variety of problems such as planning driving trajectories [12] and autonomous helicoptor flight [1]. However, most learning from demonstration problems assume that the expert is providing optimal feedback, which is often impossible to achieve. For example, in informative path planning problems [5, 9], the human cannot easily find the optimal path, but can quickly choose which paths they prefer. In our work, we account for this limitation on human



(a) Reward map representing the value of traveling in a particular area.

(b) Risk map showing the risk of traveling in a region.

(c) Utility map generated from a weighted sum of the risk and reward maps. Here, the target weights of risk and reward are -10 and 30 respectively.

Fig. 2: An example path and utility field generated using the proposed algorithm after one trial. Only the utility map shown in (2c) is presented to the expert during simulated trials. The black line represents the robot's path through the environment after the expert has made local improvements to it. Here, risk and reward integrated along the path are the features used in the utility function. The proposed algorithm learns the underlying weighting between the features using coactive learning.

performance by allowing the human to merely present a preference for a solution. In this case, the optimal solution is never needed.

Several forms of coactive learning algorithms with theoretical bounds have been studied. These include regret bounds on the perceptron coactive learning algorithm [13] and cost bounds on the cost sensitive perceptron and passive agreesive algorithms [4]. However, these bounds still assume optimal or locally optimal feedback and have not been tested directly with human experts.

Much of the work done on coactive learning algorithms has studied problems where both the expert and the learner are computer programs, which solve and improve the solution using different methods [4] [13]. A few studies of using the coactive learning algorithm online with humans have been made, most notably in learning trajectories for robotic arms [8]. They show that the robot can successfully learn from the human's iterative, sub-optimal improvements and that the coactive algorithm performs better than other learning algorithms. Our algorithm builds upon these ideas upon this by providing increased resistance to errors made by the human expert and short learning times.

This work dovetails nicely with current trends in adaptive sampling algorithms. These algorithms attempt to choose path goals that maximize information gain, minimize prediction uncertainty or minimize risk [20], [10] while minimizing the cost of performing the tour. In these algorithms, there is an implicit tradeoff between the various goals. Our work allows the robot to easily learn those tradeoffs.

A preliminary version of our algorithm was presented in our prior workshop paper [19].

III. Coactive Learning Algorithm

We first present the basic perceptron coactive learning algorithm. We build upon previous work on adapting the coactive algorithm to noisy environments [11] and present a novel approach to dealing with sub optimal updates made by the human expert.

A. Perceptron Coactive Learning Algorithm

The perceptron coactive learning algorithm attempts to learn an expert's utility function, $U(\langle x, y\rangle) \rightarrow \mathbb{R}$, for judging a candidate solution y for a given problem x (as in [4]). We assume that the expert's utility function can be approximated as a weighted linear function of the features of the candidate solution: $\hat{U}(\langle x, y\rangle) = \vec{w}^\top \vec{\phi}(\langle x, y\rangle)$. The ultimate goal of the algorithm is to learn the parameters $\vec{w}$ that match the expert's method for judging the utility of a solution.

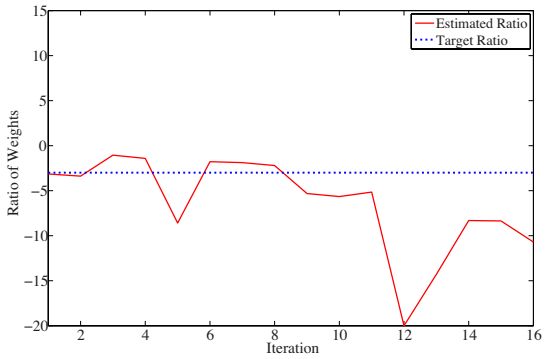
On each update of the coactive learning algorithm, the algorithm creates a candidate solution y_t based on its current estimate $\hat{U}$ of the expert's utility function and presents that solution to the expert. The expert has a set of operations, $\mathbb{O}$, that can be applied to the solution to improve it: $\mathbb{O}_i \in \mathbb{O} : \langle x, y\rangle \rightarrow \langle x, y'\rangle$. In path planning for instance, these operations might involve altering the trajectory. The cost for the update C_t is equal to the number of operations the expert applies to improve the solution. The learning algorithm then adjusts $\hat{U}$ based on the difference in parameters between y_t and y'.

Algorithm 1: CoactiveLearningUpdate (problem x_t, learning algorithm's solution y_t, improved solution y', cost C_t)

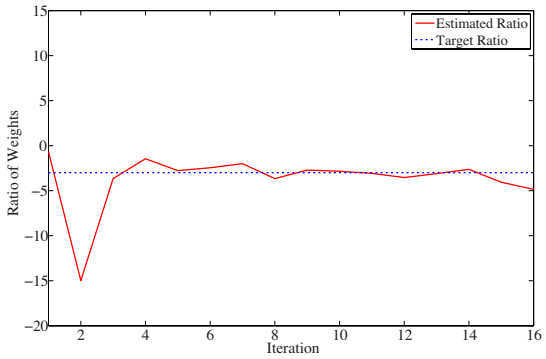
if $C_t > 0$ **then**
  $\vec{\Delta}_t := \vec{\phi}(\langle x_t, y'\rangle) - \vec{\phi}(\langle x_t, y_t\rangle)$
  $\vec{w}_{t+1}^\top = \vec{w}_t^\top + \lambda_t \vec{\Delta}_t$
end

Algorithm 1 shows how the weights w are updated. If the expert has improved the proposed solution, the difference in parameters $\vec{\Delta}$ between the proposed and improved solutions is calculated. This difference is then scaled by the learning rate and added to the previous estimated weights to find the new estimated weights.

Several variations of the coactive learning algorithm have been proposed. In [4], Goetschalckx and Tadepalli examine adjusting the learning rate λ. In addition to the above perceptron (PER) algorithm with a constant learning rate,



(a) An example of the estimated ratio of weights over a trial of the perceptron coactive learning algorithm.



(b) An example of the estimated ratio of weights over a trial of the histogram coactive learning algorithm.

Fig. 3: Example plots of how the estimated weight changes over a trial in simulations of the perceptron and histogram learning algorithms. Note that the perceptron algorithm initialy tracks the target weights relatively well until a sub-optimal update from the human expert throws it off. Comparatively, the histogram method converges on the target ratio, discounting an initial sub-optimal update.

they also study a passive aggressive (PA) algorithm, which adjusts lambda to ensure the solver's most recent mistake is corrected, and a cost-sensitive perceptron (CSPER) algorithm where the learning rate is proportional to the number of operations, C_t, applied.

Assuming that the expert provides a locally optimal solution, they prove an upper bound on the effort required by the expert. With T being the number of update steps, the upper bound is $O(1/\sqrt{(T)})$ for the PER and PA algorithms and a bound of $O(1/(T))$ for the CSPER algorithm. In [13], a lower bound of $O(1/\sqrt{(T)})$ on the algorithm's regret is shown, assuming the expert provides an optimal solution.

However, using the algorithm with a human expert breaks these assumptions. The solutions provided by the human are unlikely to be locally optimal and could even be unintentionally misleading. Furthermore, only the human's incorrect updates to a near optimal solution change the learned weights. This causes the weights to oscillate as update steps are performed. One way to mitigate this issue is to present suboptimal candidate solutions half of the time, allowing the learned weights to be reinforced [11]. We incorporate the essence of this solution as our learning algorithm does not create perfect solutions while further extending it to specifically remove the effect of incorrect or erroneous updates.

B. Proposed Histogram Algorithm

The baseline perceptron algorithm is sensitive to suboptimal updates made by the human expert. To overcome this limitation, we developed an alternate algorithm to reduce the effect of these suboptimal updates.

Our algorithm takes all previous improved weights into account when determining the new estimated weights. A histogram of the new, $\vec{w}_t$ and previously improved weights, $\vec{w}_1 ... \vec{w}_{t-1}$, for each feature is created. A normal distribution is fitted to the histogram. The center of the normal distribution is taken as the new estimated weight, $\vec{w}_{t+1}$, for each respective feature. This method excludes outliers and prevents new updates from completely changing the estimated weights. In this way, the algorithm is able to continuously converge on the human experts weightings, even when multiple incorrect estimated weights are included.

For a small number of updates it can be difficult to reliably fit a reasonable distribution to the data. In this case the $median(\vec{w}^\top)$ provides a good estimate of the new weights.

IV. Comparative Simulations

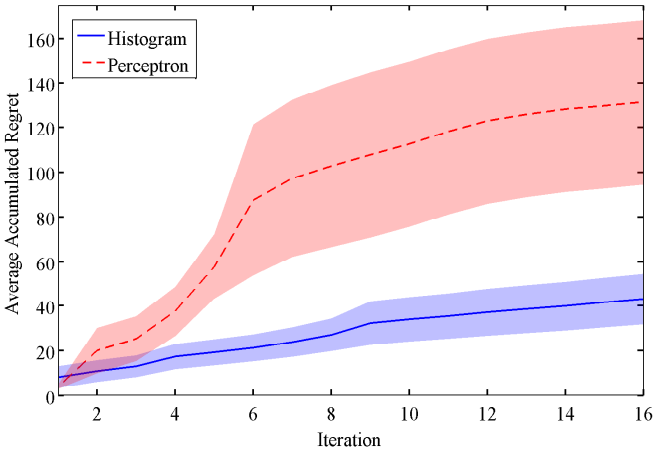


Fig. 4: The regret (sum of deviation between target weight and estimated weight) for each algorithm averaged over 20 trials. The standard error of the mean for the averages are shown. The histogram coactive learning algorithm accumu- lates regret more slowly.

The problem we examine consists of several components: a planned trajectory of waypoints, a target vector $\vec{w}^\top$ of feature weights for the learning algorithm to learn, and maps representing the value of those features in a region. These feature maps could represent real world variables, such as temperature or PH, or abstract features, such as a "risk" feature representing the cost of traveling in a given region or a "reward" feature that represents the quality and value of information gained by traveling in a given area [16, 15].

For our simulation, we assume that the expert's utility function is linearly composed of two features: the risk the robot incurs and the information it gains during its tour [7]. The total risk and total information for a path are found by

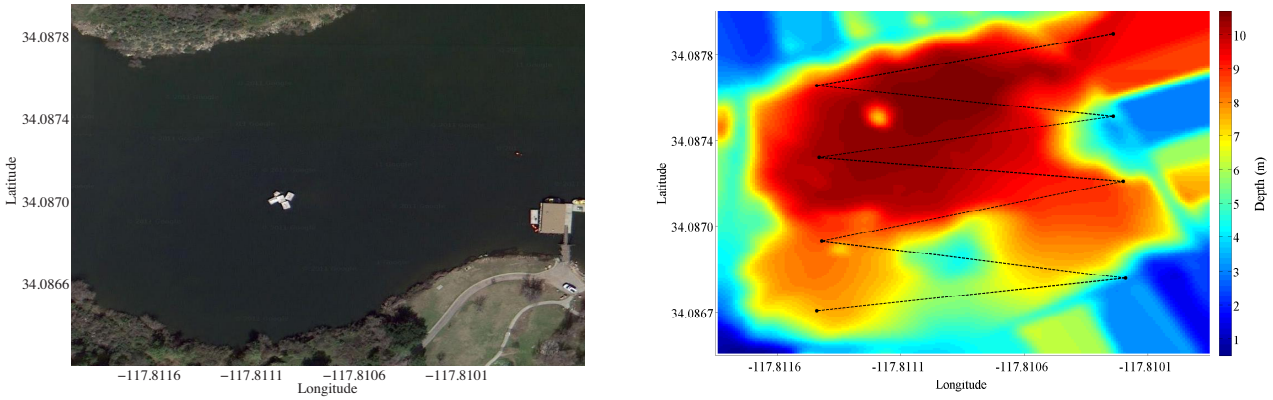


Fig. 5: An aerial view of the test area at Puddingstone Reservoir **(left)** with the corresponding depth map **(right)**. A simple baseline lawnmower path is shown over the depth map. The lawnmower path is a commonly used path for guiding AUVs in ecological monitoring missions. The depth map was created by interpolating between depth points measured on a dense survey of the area.

finding the line integral of each respective feature map along the path.

To test the algorithm's ability to learn a human expert's weighting, the expert is presented with a path overlaid on a map of the utility at each location in a region, as shown in in Fig. 2. The utility map is generated by weighting these risk and reward maps by their respective target weights and summing them. Maps of risk and reward are generated as a random sum of Gaussians. For all tests, we use target weights of -10 and 30, for risk and reward respectively. This represents a stronger preference for gathering information than avoiding risk. Since the human expert is optimizing the path based on a map of utility calculated using the target weights, we can test how effectively the learning algorithm finds these target weights.

In our tests, the algorithm used a simple greedy information gathering path planner to generate candidate paths. It is modeled on observations of our behavior as we plan paths, in which we find a short path connecting areas of high utility. First, our planner finds the peaks of a utility map made from temperature and depth maps weighted by the learned weights. Then, using a locally optimal traveling salesman problem solver [2], it connects the peaks using a path that minimizes the inverse of the utility along the path. Thus, the planner finds a short path while still maximizing the utility of that path.

At each update, the expert improves the path by moving one of the points of the path. The change in the path's information and risk are calculated and used in the coactive learning update to update the learning algorithm's estimate of the expert's utility function. A new map and path is generated for each coactive update.

We conducted 20 trials each for the the baseline perceptron and histogram algorithms. Each trial consisted of performing 16 updates based solely on the provided utility maps.

As shown in Fig. 4, the histogram algorithm accumulates regret more slowly than the perceptron coactive learning algorithm. Additionally, it also smoothly converged towards a set of estimated weights as each update shifts the histogram only slightly. As seen in Fig. 3, the perceptron algorithm is still highly susceptible to suboptimal updates made by the human, even after many iterations. This is because each update is valued equally and the algorithm cannot compare the current update to previous feedback. However, the histogram algorithm learns what the optimal weighting is and is able to ignore or reduce the effects of suboptimal updates.

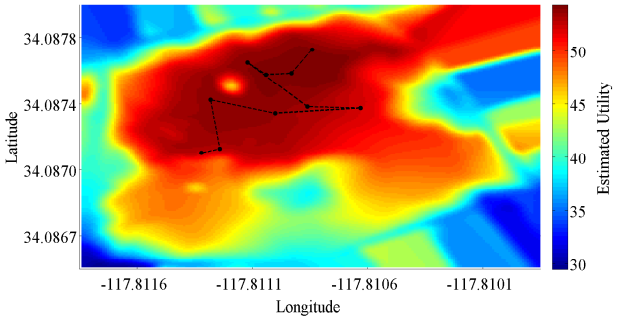
V. Field Trials

We performed a series of trials on the algorithm using a YSI EcoMapper autonomous underwater vehicle in a lake ecology monitoring scenario. These propeller-driven AUVs are able to maintain speeds of $2m/s$ for up to 10 hours. As such, they are often used in ecological monitoring and oceanographic research missions [3]. They have a wide range of sensors. These include conductivity, temperature, and depth sensors for ecological monitoring and a Doppler Velocity Log and GPS unit for localization. Missions of waypoints for the AUV to follow are uploaded wirelessly using the standard 802.11 wireless protocol. Our field trials were conducted in an inlet of Puddingstone Reservoir in San Dimas, California (Lat. 34.08, Lon. -117.81).

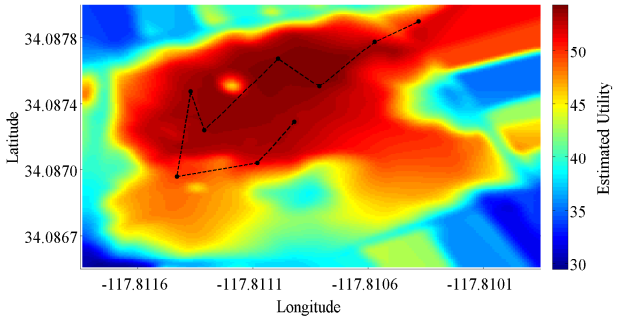
The goal in these experiments was to demonstrate that the coactive learning algorithm was robust enough for use in an integrated field environment and to show that the resulting integrated feature weights of paths planned by the human and by the algorithm were the same in a real world scenario. To do this we trained our algorithm to plan paths based on the temperatures and depths along the path.

For each trial, we began by using our coactive learning algorithm to train an information gathering planner our planning preferences. We then ran a dense lawnmower pattern over the inlet to establish a map of temperature and depth for the planning algorithm to use. Using the learned utility function, the planner than ran the AUV on a path attempting to maximize the utility of the sensed information. Due to the depth of the inlet and to simplify the experiments, the AUV was used on the surface using 2D trajectories.

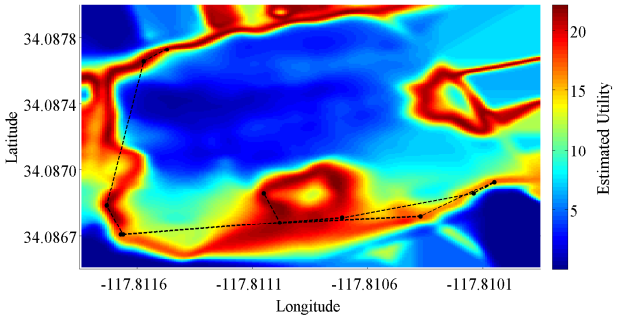
One limitation of these experiments is that depth and temperature are not independent features. Deeper locations



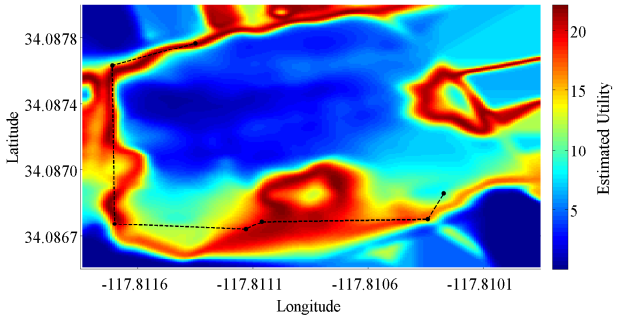
(a) Our algorithm's planned path maximizing depth while minimizing temperature.



(b) A human's planned path maximizing depth while minimizing temperature.



(c) Our algorithm's planned path targeting a depth of 6 meters and a temperature of 27° C.



(d) A human's planned path targeting a depth of 6 meters and a temperature of 27° C.

Fig. 6

are often colder because it takes longer for solar heating to warm them. As such, it is not always possible to find a path matching a given ratio of depth and temperature. For example, if maximizing depth is weighted highly with minimizing temperature having a smaller weight, it is likely the planned path will appear to satisfy both features equally.

We began by running a loose lawnmower pattern, as shown in Fig. 5. These are often used on ecological monitoring missions as they are easy to set up, so they formed a relevant baseline for our tests. The resulting ratio of depth to temperature integrated along the path was 1.157, significantly different than the human's targeted ratio. Additionally, this path is significantly longer than any of the targeted human or computer planned paths. The commonly used lawnmower pattern spends significant amounts of time areas of low utility.

We then taught the algorithm to strongly maximize the depth while minimizing the temperature and ran the mission shown in Fig. 6a. The measured ratio of -2.48 closely matched the learned ratio of -2.47. It also closely matched the measured ratio from the human planned mission in Fig. 6b of 2.45.

We also taught the algorithm to target specific depths and temperatures. We targeted depths close to 6 meters and temperatures very close to 27 degrees Celsius, weighting the depth more strongly. We again ran a computer planned and a human planned mission, shown in Figs. 6c and 6d. The algorithm learned a weight ratio of 2.56. The measured utility ratios were 1.564 for the computer's path and 1.495 for the human's path. While these are not exactly the same, they are still quite close. Additionally, the achievable ratio was limited by the correlation of the temperature and depth in the lake.

Finally, we tried to train the algorithm to follow the 6 meter depth contour by strongly preferring paths at that depth. The algorithm learned a weight of 19.34 for the utility of sampling a 6 meter depth and a weight of only 1.97 for the utility gained from sampling a 27.7 degrees Celcius temperature. As shown in Fig. 7, the algorithm successfully planned a route that closely follows the 6 meter depth contour line.

For each trial, we compared the ratio of the temperature and depth features sensed along the path for the human and algorithm planned paths to the learned weights. Even with our rudimentary path planner, the ratios matched well, with just a small amount of variability due to inaccuracies in path following and changing water temperatures. This shows that in a real-world mission, the algorithm is able to plan paths that follow the same preferences as a human expert's.

VI. Conclusion and Future Directions

We successfully applied our histogram coactive learning algorithm to path planning using a human expert, showing that the algorithm can learn and mimic a human expert's priorities. Over several trials, using a set of target weights, we found that the algorithm's estimated weights would converge on the target weights in a reasonable amount of time for use with a human expert. We also demonstrated the algorithm in field trials, showing that the algorithm can quickly and easily

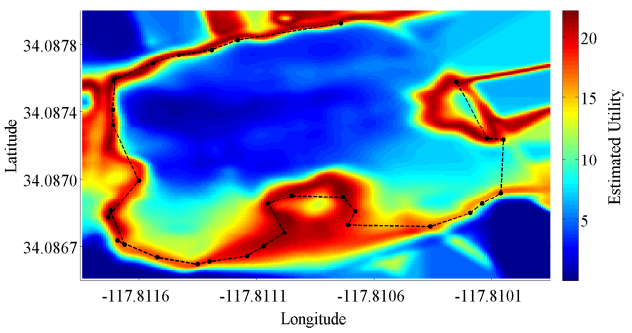


Fig. 7: A contour following path generated using our coactive learning algorithm. The algorithm was trained by consistently preferring paths around 6 meters of depth. It learned a weight of 19.34 for depths around 6 meters while ignoring other depths and temperature features with weights of less than 2.

learn the human's preferences and plan paths that perform similarly to human planned paths.

Further work needs to be done in testing the algorithm on a range of human experts to comprehensively evaluate the use of coactive learning for learning human preferences. Other path parameters should be added in order to more closely match the human's intentions and account for possible path parameters. Ultimately, we hope to be able to learn a human's preferences in trajectory planning without complete knowledge of the underlying parameters used.

Acknowledgment

The authors thank Robby Goetschalckx, Alan Fern, and Prasad Tadepalli from Oregon State University for their insightful comments. Further thanks go to Gaurav Sukhatme from the University of Southern California for providing access to the Ecomapper vehicle in the experiment.

References

[1] Pieter Abbeel, Adam Coates, and Andrew Y Ng. "Autonomous helicopter aerobatics through apprenticeship learning". In: *The International Journal of Robotics Research* 29.13 (2010), pp. 1608–1639.

[2] David L Applegate. *The traveling salesman problem: a computational study*. Princeton University Press, 2006.

[3] Rob Ellison and Mike Cook. "Cost-Effective Automated Water Quality Monitoring Systems Providing High-Resolution Data in Near Real-Time". In: *World Environmental and Water Resources Congress 2009 @ Great Rivers*. ASCE. 2009, pp. 2272–2281.

[4] Robby Goetschalckx, Alan Fern, and Prasad Tadepalli. "Coactive Learning for Locally Optimal Problem Solving". In: *AAAI*. 2014, pp. 1824–1830.

[5] Geoffrey Hollinger and Gaurav Sukhatme. "Sampling-based robotic information gathering algorithms". In: *International Journal of Robotics Research* 33.9 (2014), pp. 1271–1287.

[6] Geoffrey Hollinger et al. "Underwater Data Collection Using Robotic Sensor Networks". In: *IEEE Journal on Selected Areas in Communications* 30.5 (2012), pp. 899–911.

[7] Geoffrey A Hollinger and Gaurav S Sukhatme. "Trajectory Learning for Human-robot Scientific Data Collection". In: *Proc. International Conference on Robotics and Automation (ICRA)*. 2014, pp. 6600 –6605.

[8] Ashesh Jain et al. "Learning trajectory preferences for manipulators via iterative improvement". In: *Advances in Neural Information Processing Systems*. 2013, pp. 575–583.

[9] Andreas Krause and Carlos Guestrin. "Submodularity and its Applications in Optimized Information Gathering". In: *ACM Transactions on Intelligent Systems and Technology* 2.4 (2011), 32:1–32:20.

[10] Kian Hsiang Low, John M Dolan, and Pradeep K Khosla. "Information-Theoretic Approach to Efficient Adaptive Path Planning for Mobile Robotic Environmental Sensing." In: *ICAPS*. 2009, pp. 233–240.

[11] Karthik Raman et al. "Stable coactive learning via perturbation". In: *Proceedings of The 30th International Conference on Machine Learning*. 2013, pp. 837–845.

[12] Nathan D Ratliff, James Kuffner, and Andrew Ng. "Learning to search: structured prediction techniques for imitation learning". PhD thesis. Carnegie Mellon University, 2009.

[13] Pannaga Shivaswamy and Thorsten Joachims. "Online Structured Prediction via Coactive Learning". In: *International Conference on Machine Learning (ICML)* (2012), pp. 1431–1438.

[14] David Silver, J Andrew Bagnell, and Anthony Stentz. "Learning from demonstration for autonomous navigation in complex unstructured terrain". In: *The International Journal of Robotics Research* 29.1 (2010), pp. 1565–1592.

[15] Amarjeet Singh, Andreas Krause, and William Kaiser. "Nonmyopic Adaptive Informative Path Planning for Multiple Robots". In: *Proceedings of Twenty-First the International Joint Conference on Artificial Intelligence*. 2009, pp. 1843–1850.

[16] Amarjeet Singh et al. "Efficient planning of informative paths for multiple robots". In: *IJCAI*. 2007, pp. 2204–2211.

[17] Ryan Smith et al. "Persistent ocean monitoring with underwater gliders: Adapting sampling resolution". In: *Jouirnal of Field Robotics* 28.5 (2011), pp. 714–741.

[18] Ryan Smith et al. "USC CINAPS Builds Bridges: Observing and Monitoring the Southern California Bight". In: *IEEE Robotics and Automation Magazine* 17.1 (2010), pp. 20–30.

[19] Thane Somers and Geoffrey A Hollinger. "Coactive Learning with a Human Expert for Robotic Monitoring". In: *Workshop on Robotic Monitoring at Robotics Science and Systems*. 2014.

[20] David R Thompson et al. "Spatiotemporal path planning in strong, dynamic, uncertain currents". In: *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE. 2010, pp. 4778–4783.