Adaptive Parameter EXploration (APEX): Adaptation of Robot Autonomy from Human Participation

Anqi Xu, Arnold Kalmbach, and Gregory Dudek

***Abstract*— The problem of *Adaptation from Participation* (AfP) aims to improve the efficiency of a human-robot team by adapting a robot's autonomous systems and behaviors based on command-level input from a human supervisor. As a solution to AfP, the *Adaptive Parameter EXploration* (APEX) algorithm continuously explores the space of all possible parameter configurations for the robot's autonomous system in an online and anytime manner. Guided by information deduced from the human's latest intervening commands, APEX is capable of adapting an arbitrary robot system to dynamic changes in task objectives and conditions during a session. We explore this framework within visual navigation contexts where the human-robot team is tasked with covering or patrolling over multiple terrain boundaries such as coastlines and roads. We present empirical evaluations of two separate APEX-enabled systems: the first, deployed on an aerial robot within a controlled environment, and the second, on a wheeled robot operating within a challenging university campus setting.**

I. INTRODUCTION

We define *Adaptation from Participation* (AfP) for a human-robot team as the problem of dynamically adjusting configuration parameters of an autonomous robot system (henceforth referred to as a *robot autonomy*) with the aim of maximizing the team's overall efficiency in terms of improved task performance and reduced human workload. We propose an online and anytime solution to AfP called *Adaptive Parameter EXploration* (APEX), which uses multiple competing parameter hypotheses that we refer to as *particles* to simultaneously explore the parameter space of the robot autonomy. Particles optimize their hypotheses using information from the human's latest intervening commands to search for configuration settings that can effectively handle the evolving task objectives and environmental conditions that occur during a session. We evaluate two APEX-enabled autonomies for terrestrial and aerial visual navigation tasks wherein human operators and autonomous robots collaborate to sequentially cover or patrol through different terrain boundaries such as shorelines and roadsides.

Human-robot teams have the potential to solve very challenging tasks as they combine the heightened dexterity and comprehensive planning capabilities of autonomous robots with the keen instincts and creative problem solving skills of humans. A key concern, however, is that it is often difficult for the operator to configure the robot autonomy by hand, e.g. when tuning controller gains, adjusting learning rates, etc. Such manual adjustments are especially challenging to perform during an active task session and also for systems whose parameters interact with one another. AfP serves as a powerful interaction paradigm by allowing non-expert users to train robots through mere participation in the team, thereby reducing the need to have designer-level grasp of the robot's internal workings.

The authors are with the School of Computer Science, McGill University, 3480 University Street, Montréal, QC, Canada H3A 2A7 {anqixu,akalmbach,dudek}@cim.mcgill.ca

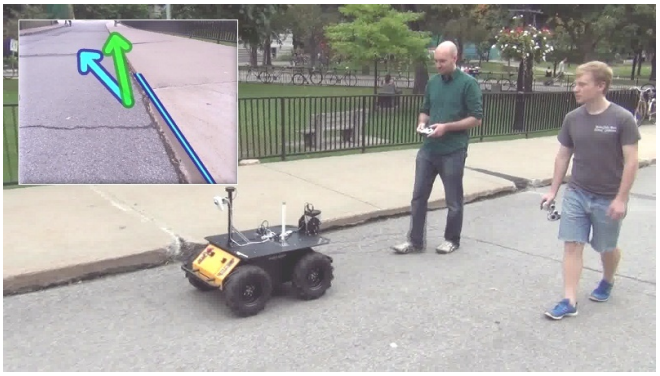

Fig. 1. Robot operator (left) collaborating with autonomous vehicle during boundary patrol field trial, supervised by experimenter (right). Inset: robot autonomy's camera view (not shown during field trial), overlaid with detected boundary (blue line), autonomy's steering command (blue arrow), and operator's intervening command (green arrow).

As an online solution to AfP, APEX is especially useful when task conditions and goals evolve over time. This is because the human's sole duty is to issue intervening commands when the robot misbehaves or when task objectives change. The robot can then learn from these interventions and adapt to match its behaviors to the evolving task on a moment-to-moment basis. In addition, APEX can also reconfigure the robot autonomy automatically after its physical configuration has been perturbed, for instance when its camera is re-positioned by an external agent.

Our contributions are centered around the novel problem of *Adaptation from Participation* (AfP), as well as our algorithmic solution, *Adaptive Parameter EXploration* (APEX). Sec. II begins with a brief survey on related research, which is followed by formulations of AfP (in Sec. III) and APEX (in Sec. IV). Sec. V describes two separate APEX-enabled robot autonomies, built on our prior work for visual navigation applications [1]. These systems provide the capability of performing boundary tracking tasks to both a wheeled robot and an aerial robot, each with distinct sensing capabilities. Sec. VI and VII present empirical evaluations comparing the efficiency of APEX-enabled autonomies against other common human-robot team setups through both a controlled user study and a series of field trials in a challenging university campus environment (see Fig. 1). Finally, our work is summarized in Sec. VIII.

II. RELATED WORK

Adaptation from Participation (AfP) is closely related to the problem of *Learning from Demonstration* (LfD), in that the objective of LfD is for a robot to learn behaviors from demonstrations provided by a human or another robot expert with superior task knowledge [2]. LfD has been studied extensively in the literature, and is also commonly referred to as Inverse Reinforcement Learning [3], Apprenticeship Learning [4], Robot Programming by Demonstration (PbD) [5], and imitation learning [4], [6].

Abbeel and Ng [4] presented a solution to imitation learning using the framework of Markov Decision Process without Rewards (MDP\R), and demonstrated the ability for a robot to acquire complex behaviors such as highway driving by observing humans. This approach does not require the user to specify explicitly the rewards or penalties of each particular action towards being a competent driver, which is similar to the goal for AfP of removing the need for operators to manually configure parameters of a robot autonomy.

Billard *et al.* [5] provided a survey of different solutions and systems for Robot Programming by Demonstration. The authors indicated that a key motivation for PbD is to eliminate the tedious task of manually programming behaviors for robots, akin to the goals of AfP addressed in this work.

AfP also builds on previous research in the domain of shared autonomy, which refers to the shared control of a robotic platform between a human operator and an autonomous agent. Human-robot teams using shared autonomy control have been empirically shown to achieve better task performance than both a fully tele-operated system and a fully autonomous one [7], [8]. AfP aims to further increase the efficiency of shared autonomy systems by enabling the robot to adapt its behaviors based on the human's participation, similar to the interaction method used in LfD.

Dogged Learning (DL) [9] is a closely related interaction paradigm that also aims to combine concepts from LfD and shared autonomy. DL builds an autonomous robotic agent using an online LfD approach, and then arbitrates between commands from this agent, from the human demonstrator, and optionally from a reactive controller, through a common measure of confidence. A main focus of this work is to develop a generic algorithm that learns a mapping from sensor inputs to control outputs in a *task-independent* manner. In contrast, AfP aims to improve the cumulative efficiency of an existing *task-specific* robot system by repeatedly adjusting its parameters.

Dragan and Srinivasa [10] established a unifying formalism of *policy blending* for shared autonomy systems, where action policies from a human expert and a planning algorithm are combined to produce optimal behaviors. Using policy blending, the authors developed a robot manipulator planner that generates trajectories by estimating the *intent* of human-demonstrated motions. Our APEX algorithm similarly aims to adapt to the task intents conveyed through a sequence of human interventions, while ignoring the effects of moment-to-moment noise found within individual command signals.

APEX's parameter adaptation approach bears similarities to the works of Vanier and Bower [11], who investigated the performance of several optimization methods for automatically training Artificial Neural Networks (ANN). Bergstra and Bengio [12] also studied the problem of hyper-parameter optimization for ANNs, and showed that a random search method is more efficient at finding static optimal hyper-parameters, compared to both grid-based exhaustive search and manual tuning by experts. APEX extends the scope of the parameter optimization problem to the broader case where the robot's optimal parameter settings change over time.

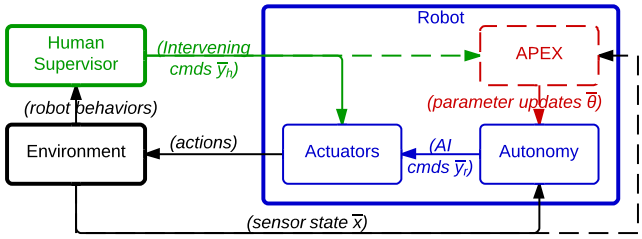

Fig. 2. Block diagram for a supervisor-worker human-robot team, including our solution to *Adaptation from Participation* (AfP), namely the *Adaptive Parameter EXploration* (APEX) module (shown in dashed lines).

III. FORMALISM

In this section we characterize the main elements of our human-robot interaction (HRI) framework, which forms the basis for the formulation of the *Adaptation from Participation* (AfP) interaction paradigm and computational problem.

A. Interaction Context

We are interested in HRI scenarios where an autonomous robot works in tandem with a human operator towards an assigned task. In this context, the "robot autonomy" consists of *sensing*, *planning*, and *control* modules responsible for producing autonomous behaviors for the robot. Under normal operations, the robot autonomy is responsible for handling the bulk of the work while the human supervises the task progress. In contrast, when a challenging scenario causes the robot to misbehave, the supervisor can choose to take over control and correct these mistakes, until the robot is capable of performing on its own again. Given this supervisor-worker relationship, we assume that the human's commands will *always supersede* those generated by the robot autonomy.

The main constituents of our HRI setting are shown in Fig. 2. The robot autonomy $\mathbb{A}$ repeatedly senses and processes the state of the world $\overline{x}$ in order to generate command outputs $\overline{y_r}$. For our visual navigation contexts, the output is in the form of a heading command $\bar{y} \in [0^\circ, 360^\circ)$. These commands are also affected by the current system parameter settings $\overline{\theta}$, and furthermore the internal state of the autonomy $\bar{s}$ may be modified in the process; therefore the robot autonomy can be viewed as $\overline{y_r}, \bar{s} \leftarrow \mathbb{A}(\overline{x}, \overline{\theta}, \bar{s})$. Separately, the human supervisor can issue intervening commands $\overline{y_h}$ at any time to the robot's actuators directly; commands generated by the robot autonomy $\overline{y_r}$ are only sent to the actuators when the supervisor is not intervening, i.e. $\overline{y_h} = \varnothing$.

B. Adaptation from Participation

A common shortcoming among human-robot teams is that in challenging scenarios where the robot repeatedly makes mistakes, the human's interactions degenerate from high-level supervision to full-time tele-operation. The *Adaptation from Participation* (AfP) paradigm addresses this concern by re-tuning the robot autonomy during periods of human interventions in order to improve the robot's task performance over time and alleviate the need for constant human intervention. Also, when task requirements change during a session, AfP enables the robot to dynamically adjust its settings and behaviors to match the updated conditions.

Before formalizing AfP as a computational problem, we first assume that efficiency metrics for task performance $\mathbb{P}$ and for human intervention $\mathbb{I}$ are available. Whereas the former metrics are typically application-specific, a standard measure for human intervention given our interaction context is to compute the fraction of time instances where the human is not intervening.

The objective of AfP is to optimize the robot autonomy's parameters $\overline{\theta}$ throughout periods of constant human interventions towards *greater team efficiency* by simultaneously improving the task performance $\mathbb{P}$ and reducing the frequency of human interventions $\mathbb{I}$. Conceptually, we assume that for any given task there exists a set of optimal parameter values $\left\{\overline{\theta^*}\right\}$ that can result in the best possible robot behaviors. These optimal parameter values, however, are usually unobserved, and furthermore can be altered by changes in the world or to the robot's configuration. By assuming that the human supervisor can react to these changes, AfP aims to maintain an up-to-date $\overline{\theta^*}$ by deducing from the supervisor's intervening commands $\overline{y_h}$.

To solve the AfP problem, we must balance between the optimization objectives $\mathbb{P}$ and $\mathbb{I}$. In certain domains, the cost of human interventions naturally translates into an equivalent penalty for task performance; for example, both autonomous productivity and manual interventions can be quantified in terms of monetary gains and losses in a manufacturing setting. Our empirical evaluations in Sec. VI and VII address this multi-objective issue in a statistical manner.

Despite their many similarities, AfP differs from *Learning from Demonstration* (LfD) in some key aspects that highlight its novelty. Firstly, AfP is designed for highly *dynamic situations* where task objectives and conditions change unpredictably and repeatedly over the course of a single session; in contrast, each LfD system has a *single, stationary* task objective that it is trying to learn from the human's demonstrations. In addition, the purpose of LfD is for a robot agent to *learn* a *novel* set of task behaviors, whereas AfP instead aims to improve the performance of an *existing* robot autonomy by *adaptively* toggling between its range of behaviors. Although AfP does not extend the behavioral capabilities of an existing autonomy, it can be used to augment a sophisticated autonomous robot system and be readily deployed to solve challenging, real-world tasks.

IV. ADAPTIVE PARAMETER EXPLORATION

We now present *Adaptive Parameter EXploration* (APEX): an online and anytime algorithm for adapting a robot autonomy's parameters during periods of interaction where the human supervisor is constantly issuing commands. APEX continuously searches the parameter space of the robot autonomy for better configurations using a multi-hypothesis approach, where each evolving parameter hypothesis is referred to as a *particle*. In the time between two consecutive sensor updates, each particle optimizes its parameter values based on training exemplars consisting of matched pairs of sensor readings and the human's intervening commands, within a recent period of time. The results of this optimization process on their own are short-sighted and susceptible to noise, therefore when adopting the winning particle's hypothesis APEX uses several measures to enforce temporal consistency and account for the long-term history of parameter evolutions.

A. Applicability and Prerequisites

APEX is capable of tuning parameters for a wide variety of different autonomous systems. An important characteristic of a subset of these systems is the ability to analytically compute the inverse autonomy mapping $\overline{\theta^*} = \mathbb{A}^{-1}(\bar{x}, \bar{s}, \overline{y^*})$, i.e. a separate program for computing the *exact* parameter settings $\overline{\theta^*}$ matching a desired output command $\overline{y^*}$, sensory inputs $\bar{x}$, and the autonomy's prior state $\bar{s}$. In most setups, the robot autonomy is sufficiently complex such that obtaining $\mathbb{A}^{-1}$ is infeasible, or even ill-posed. We refer to systems without an inverse autonomy mapping as *black-box autonomies*, and those with an accessible $\mathbb{A}^{-1}$ as *white-box autonomies*.

APEX's generic design allows it to optimize different parameter types, including bounded continuous parameters, ordinal discrete parameters, and categorical parameters. It is important to provide sensible parameter ranges when deploying APEX, especially to ensure that the autonomy's outputs do not saturate for nearby parameter values. Nevertheless, system designers are free to specify loose parameter ranges that satisfy the above condition, since APEX's particles are designed to lock onto optimal regions in the parameter space.

B. APEX Algorithm

The main pipeline loop of APEX adds routines both before and after processing the latest sensor data $\bar{x}$ through the robot autonomy $\mathbb{A}$, as illustrated by Algorithm 1. Each APEX particle i has an associated *long-term cost* $\mathscr{C}_i$, which keeps track of the consistency of the performance of its searched parameter results over time. Whenever a new piece of sensor data becomes available, APEX first pauses the optimization process for all particles, updates $\mathscr{C}_i$ based on the quality of each particle's latest hypothesis, and incorporates parameter values from the lowest-cost, *winning particle* back into the main pipeline. Next, the robot autonomy $\mathbb{A}$ processes the sensor input $\bar{x}$ using the updated parameter settings $\overline{\theta}$ to produce a new control command $\overline{y_r}$. Finally, during periods of manual interventions the particles are resumed to search for better parameter values; otherwise, they are reset during periods of autonomous control.

Algorithm 1 APEX's main pipeline loop

$\mathscr{C}_i \leftarrow 0 \ \forall i$
loop
  wait for incoming sensor update $\overline{x}$
  if *particles are optimizing parameters* $\overline{\theta_i}$ **then**
    for all *particles* i **do**
      pause optimization
      update long-term cost $\mathscr{C}_i$
    end for
    $i^* \leftarrow argmax_i(\mathscr{C}_i)$ *// choose winning particle*
    update main pipeline's parameters $\overline{\theta}$ *with* $\overline{\theta_{i^*}}$
  end if
  $\overline{s'} \leftarrow \overline{s}$ *// save prior autonomy state*
  $\overline{y_r}, \overline{s} \leftarrow \mathbb{A}(\overline{x}, \overline{\theta}, \overline{s'})$
  if $\overline{y_h} \neq \varnothing$ **then**
    store latest exemplars $\{\overline{x}, \overline{s'}, \overline{y_h}\}$
    resume optimization for all particles
  else
    $\mathscr{C}_i \leftarrow 0 \ \forall i$
  end if
end loop

Between two sensor updates, each particle i continuously optimizes its parameter hypothesis $\overline{\theta_i}$ using a sequence of the W most recent training exemplars $\{\overline{x_w}, \overline{s'_w}, \overline{y_{h,w}}\}_{w=1..W}$. Each exemplar consists of a sensor state $\overline{x}$, a prior state for the robot autonomy $\overline{s'}$, and a command from the supervisor $\overline{y_h}$. The following mean squared cost objective is used:

$$cost\left(\overline{\theta_i}\right) = \frac{1}{W}\sum_{w=1}^{W}\left\|\overline{y_{h,w}} - \mathbb{A}\left(\overline{x_w}, \overline{\theta_i}, \overline{s'_w}\right)\right\|^2$$

For black-box autonomies, particles perform numerical optimization by repeatedly querying copies of the autonomy $\mathbb{A}$ using different hypotheses $\overline{\theta_i}$ each time. On the other hand, particles interacting with a white-box autonomy can directly solve for the best parameter values $\overline{\theta^*}$ using $\mathbb{A}^{-1}$, through a single-step least squares formulation.

Since APEX is an anytime algorithm [13], it can update the autonomy's parameter settings whenever new sensor data becomes available, by integrating the hypothesis from the *winning particle* i^* at those time instances. Unfortunately, it would be short-sighted to choose the winning particle by the $cost\left(\overline{\theta_i}\right)$ of its latest hypothesis directly, since these costs are measured against the W most recent training exemplars only. Instead, these momentary costs are updated into the particles' long-term cost attributes $\mathscr{C}_i$, which are then used to identify the winning particle i^* based on its historical success at consistently finding low-cost configurations in the past:

$$\mathscr{C}_i \leftarrow \gamma\mathscr{C}_i + cost\left(\overline{\theta_i}\right)$$

This enforces temporal consistency and reduces the likelihood of oscillating between different winning particles in successive iterations, which can lead to undesired and discontinuous control. When updating $\mathscr{C}_i$, the hyper-parameter $\gamma \in [0,1]$ discounts previously accumulated costs, and its value reflects how much historical consistency matters for a given application context.

Once the winning particle i^* is determined, its optimized hypothesis is used to update the parameter values for the main pipeline. Whereas discrete parameters are assigned values from the winning particle directly, continuous parameters $\overline{\theta^c}$ are smoothed using a learning rate $\alpha \in (0,1]$:

$$\overline{\theta^c} \leftarrow \overline{\theta^c} + \alpha\left(\overline{\theta^c_{i^*}} - \overline{\theta^c}\right)$$

This hyper-parameter also mitigates the short-sightedness of optimizing based on the W latest training exemplars only.

C. Particle Types

APEX uses a combination of four different types of particles to effectively explore the parameter space:

- *local search particles* employ a gradient-based search method to iteratively find numerical approximations to locally optimal parameter values;
- *random restart search particles* randomly sample initial values within the specified parameter ranges, and then perform iterative local search;
- *inverse optimal search particles* use the inverse autonomy mapping $\mathbb{A}^{-1}$ to solve for optimal parameter values directly through an analytical least squares formulation;
- *persistence particles* duplicate the winning particle from the previous loop iteration, and do not perform any further parameter optimizations.

For black-box autonomies, we prescribe a combination of *local search particles* and *random restart search particles* to solve for globally optimal parameter settings. Each *local search particle* randomly samples its local parameter neighborhood for a promising gradient direction, and then explores along this gradient in order to find configurations that minimize $cost\left(\overline{\theta_i}\right)$. These local search efforts are complemented by non-local explorations in the parameter space through the use of *random restart search particles*. In contrast, white-box autonomies are paired with *inverse optimal search particles* to find the best parameter settings directly, without having to perform iterative numerical optimization. Finally, both black-box and white-box autonomies use a *persistence particle* to enforce temporal consistency by ensuring that the winning particle's parameter hypothesis in the current loop iteration is at least as good as the previous iteration's settings.

The search methods described above are designed specifically to optimize continuous or ordinal parameters. APEX uses a separate strategy to search amongst categorical parameters by instantiating one search particle for each combination of all discrete parameter values. This way, the optimal categorical parameter settings are determined automatically when selecting the winning particle on each iteration.

V. SYSTEMS

In this section we present two autonomous systems for boundary tracking tasks, targeting an aerial vehicle as well as a wheeled robot. Both autonomies are equipped to identify terrain boundaries from camera frames, although the controllers that translate detected boundaries into steering commands are distinct on each system. Both the aerial and wheeled robots operate at fixed velocities, therefore

the robot autonomy and the human supervisor are solely responsible for regulating each vehicle's heading direction. In both setups, the supervisor intervenes by holding the analog joystick on a gamepad towards a desired heading, and returns control back to the robot autonomy by releasing the joystick.

A. Autonomous Boundary Tracker

Vision-based navigation tasks are appealing for human-robot teams because humans are naturally inclined to solve them robustly and without effort, whereas they also necessitate sophisticated autonomous solutions (e.g. [14], [15]). Our boundary tracking algorithm is an extension of previous work [1], which was originally deployed on a fixed-wing aerial vehicle equipped with a downward-facing camera.

Our boundary tracking algorithm has two key parameters that allow it to detect and follow diverse classes of terrain boundaries. The *boundary type* $T_b \in \{Edge, Strip\}$ parameter can be configured to track edge-like boundaries such as coastlines, or strip-like boundaries such as roads. In addition, different image features can be employed to segment terrain patches within camera frames by adjusting the *clustering type* parameter $T_c \in \{Hue, Grayscale, HSV\}$. Further details on the image processing algorithm are discussed in [1].

B. Aerial Boundary Tracker System

For the aerial vehicle, the boundary tracker's output is connected to a Proportional-Derivative (PD) controller. This allows the system to cope with aggressive steering needs in situations where the boundary frequently changes shape. We deployed this aerial robot autonomy on a simulated platform that generates downward-looking camera view from a holonomic aerial robot, based on static satellite footage.

During a task session, the user is presented with a graphical interface showing real-time camera feed from the aerial vehicle. As depicted by the inset of Fig. 1, the latest tracked boundary line and steering command generated by the autonomy are overlaid on each camera frame, in order to provide feedback of the autonomy's internal state. This feedback is useful even during periods of manual intervention, since the operator can visually gauge when the robot is capable of resuming the boundary tracking task.

We employed APEX to adapt the boundary tracking algorithm's discrete parameters T_b and T_c, as well as the coefficients of the PD controller K_p and K_d. We also provided loosely estimated ranges for the latter parameters in order to allow the autonomy to adapt to different steering styles.

C. Terrestrial Boundary Tracker System

Our boundary tracking algorithm was also extended to control a wheeled platform with a tilted, front-facing camera, which can be seen in Fig. 1. When using a tilted camera, the detected boundary line in a given frame no longer maps directly onto a heading command. Instead, we compute the intersection χ between the boundary line and the bottom of the image, as well as the slope ϕ of the line, and prescribe the following parametric mapping for generating headings:

$$y_r = M_1\chi + M_2\phi + M_3$$

This mapping represents a linear approximation of the projected boundary position on the vehicle's ground plane, and includes two scaling factors M_1, M_2 for the boundary line's intercept and slope terms. In addition, the additive parameter M_3 relates to the desired distance to follow a given side boundary. This mapping is best suited for when the image contains only the ground plane and not the horizon, thus we remove the top H_0 percentage of each camera frame prior to running through the boundary tracking algorithm.

In contrast to the aerial boundary tracker setup, the human supervisor operates alongside the wheeled autonomous robot, without a graphical display for feedback. The user has greater situational awareness from this third-person view, although the autonomy's state is also less transparent. Anecdotally, we observed that users adopted a greater tendency to switch to tele-operation when the robot misbehaved momentarily.

For this terrestrial system, we configured APEX to adapt the pre-processing image cutoff parameter, H_0, the boundary tracker's discrete parameters, T_b and T_c, as well as the various mapping parameters M_1, M_2, and M_3. Sensible ranges for all parameters were estimated empirically.

VI. CONTROLLED USER STUDY

We carried out a user study to evaluate the efficiency of our aerial boundary tracking robot within a controlled setting. The study is structured as a *coverage game*, where the objective is to fly over as much of a designated boundary course as possible within a fixed 3-minute time limit.

A. User Study Setup

This evaluation aims to compare the efficiency of adaptive robot autonomies using APEX against other common types of human-robot teams. Specifically, the study entails four coverage sessions with the following autonomy settings:

- `APEX BB`: a black-box APEX-enabled autonomy that employs 6 *local search particles*, one for each combination of discrete parameter values, as well as 2 *random restart search particles* and a *persistence particle*;
- `APEX WB`: a white-box APEX-enabled boundary tracker that uses 6 *inverse optimal search particles* and a *persistence particle* to adapt parameters;
- `CONST`: a non-adaptive tracker with empirically-tuned parameter values by an expert designer;
- `MANUAL`: a baseline setting where the boundary tracking algorithm is disabled, and where the supervisor must rely solely on tele-operation.

In addition to being a baseline, `MANUAL` also reflects extreme circumstances where the robot autonomy constantly fails.

The study begins with a brief tutorial that explains the setup, properties of the adaptive autonomous systems being evaluated, and the study procedure. This is followed by an extended practice coverage session to familiarize users with the interface and with the interaction process of collaborating with an adaptive boundary tracking robot. The data collection phase of the study consists of 4 coverage sessions of 3 minutes each, using the aforementioned robot autonomies in a random order following a counterbalanced design.

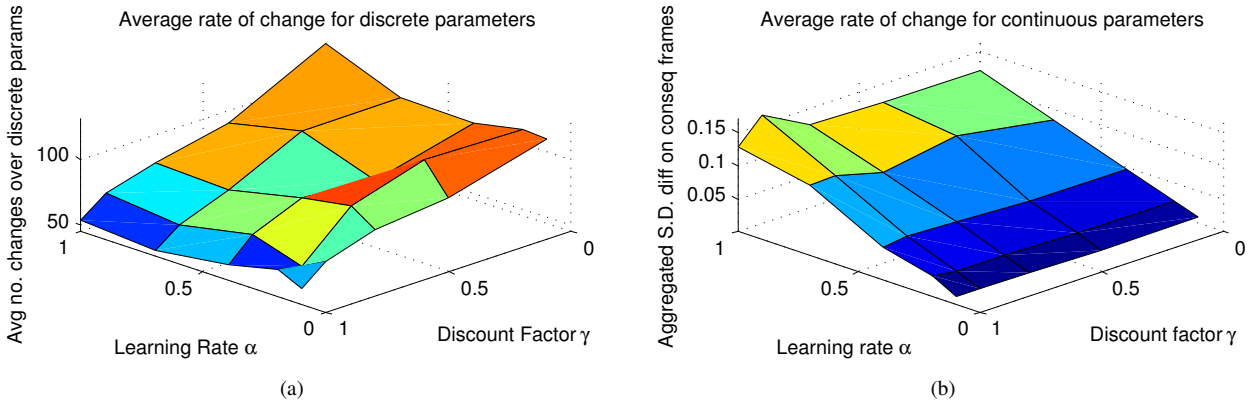

Fig. 3. Average rate of change for discrete parameters (a) and for continuous parameters (b), aggregated over 2 expert users' data. The rate of change for discrete parameters is correlated to the discount factor γ, whereas the rate of change for continuous parameters is correlated to the learning rate α.

Participants are provided with a graphical interface that integrates the aerial robot's camera view, a separate mini-map displaying the vehicle's position and the designated coverage course, as well as the current session's coverage score and remaining time. The mini-map allows users to monitor their coverage task progress, although the zoomed view of the camera feed is necessary to adequately steer the robot.[1]

The designated flight trajectory incorporates three different terrain boundaries, comprising of a wide *Highway* section with many potential boundaries visible at any given time, a narrow *Forest Path* with significant tree cover, and a highly curved *Coastline* segment. The *Coastline* section is particularly difficult for both the human users and the robot autonomy to track, since it requires aggressively reactive steering in order to accommodate to the changing and varied forms of many small inlets and peninsulas.

B. Evaluation Metrics

This study employs multiple metrics to evaluate the task efficiency of each robot autonomy setting. The *coverage score* corresponds to the amount of the designated course covered by the aerial robot's camera view, and thus quantifies the aggregated task performance within the boundary segments throughout each session. Since participants were informed about the exact trajectory of boundaries to follow, the *mean distance to ground truth* pose reflects moment-to-moment task progress. Separately, the *AI failure ratio* measures the fraction of time steps where the autonomy encountered an exceptional clause internally and terminated without producing a sensible command, for instance when the segmentation sub-module failed to identify any boundaries in the image. This metric quantifies the reliability and robustness of each robot autonomy configuration. Similarly, the *supervisor intervention ratio* records the fraction of time steps having intervening commands present, and captures the amount of active human workload incurred during each boundary segment and each coverage session. Finally, we solicit *user preference* ratings through a bounded numerical scale after each coverage session, in order to gauge participants' subjective assessments on the efficiency and adaptability of the various robot autonomy settings.

The aforementioned metrics quantify distinct aspects of the efficiency in our human-robot teams, and are all essential in forming a sufficient evaluation of APEX with respect to the objectives of AfP. Unfortunately, it is very challenging to combine these quantities together into a single score without performing additional empirical analyses to determine the per-metric normalization constants and the relative weightings of each metric. We address this concern by instead computing *qualitative* orderings of the four robot autonomy settings, separately for each metric and for each user. These orderings are then aggregated statistically using the Kemeny-Young voting method [16], which computed the optimal aggregated ranking based on the frequency of pairwise ranking comparisons across different robot autonomy settings.

C. Selection of APEX's Hyper-parameters

The rate of adaptation for APEX-enabled autonomies are regulated by the learning rate α and the discount factor γ hyper-parameters. To ensure that APEX adapts the autonomy at an adequate rate, values for these hyper-parameters were computed from empirical analyses by two expert operators prior to the user study. Concretely, the experts completed coverage sessions using the `APEX BB` setting for a $\{5 \times 5\}$ sampling of the hyper-parameter values. The resulting metric scores were then aggregated statistically to reveal $\alpha = 0.4$ and $\gamma = 0.4$ as the highest ranked hyper-parameter setting.

Fig. 3 shows that α and γ independently control the rates of adaptation for the continuous and discrete parameters of the aerial boundary tracking system. The association between γ and the adaptation rate of discrete parameters is due to the instantiation of individual particles for each combination of discrete parameter values. Since γ regulates the importance of each particle's historical search performance, smaller values will result in myopic selections with more frequent changes in the discrete parameters' values.

[1]Please refer to the video accompaniment for visual illustrations of user interface and coverage course. (HD version: `bit.ly/ICRA2014APEX`)

TABLE I

AGGREGATED RANKINGS OF DIFFERENT ROBOT AUTONOMY SETTINGS FOR COVERAGE SESSIONS IN CONTROLLED USER STUDY

Segment	Best	Second	Third	Worst
Session-wide	CONST	APEX BB	APEX WB	MANUAL
Highway	CONST	APEX WB	APEX BB	MANUAL
Forest Path	CONST	APEX WB	APEX BB	MANUAL
Coastline	APEX WB	APEX BB	CONST	MANUAL

D. User Study Results

We recruited 15 participants for this controlled study, comprising of graduate students, researchers, and professors working on robotics research, at the School of Computer Science at McGill University. Metric scores were computed for each coverage session, and also for each of the three boundary segments individually. These scores were then aggregated as rankings to establish the overall ordering of the four robot autonomy settings, as shown in Table I.

During the segments of *Highway* and *Forest Path*, the autonomy with static parameters, CONST, out-performed both adaptive systems. Since the parameters of the aerial boundary tracker were sufficiently straight-forward to conceptualize and tune, it is not surprising that the static, hand-optimized settings used by CONST resulted in performance comparable to the adaptive variants for relatively simple environments. In contrast, the CONST robot autonomy failed to adapt to the dynamic boundary conditions during the *Coastline* segment, and resulted in worse performance than either of the two APEX-enabled autonomies.

These results also show that APEX WB out-ranked APEX BB within each of the three boundary segments. This can be attributed to the fact that APEX BB uses gradient-based numerical solvers to iteratively find better parameter values, whereas APEX WB can compute the exact best parameter values for the least squares inverse optimal solution, thus producing higher quality configurations overall. Nevertheless, APEX BB is favored over APEX WB on the session-wide scale, which suggests that the adaptive performance of the two robot autonomies were comparable in general. Finally, although certain expert participants achieved higher task performance using manual tele-operation, MANUAL was statistically the least preferred setting across all segments and at the session-wide scale among our study's population.

In summary, our results empirically demonstrated that APEX is capable of properly adapting system parameters of an aerial boundary tracking robot based on commands from users with minimal knowledge of the autonomy's internal logic. In addition, the efficiency of APEX-enabled adaptive autonomies were comparable to, and sometimes even out-performed, a boundary tracking configuration with efficient static parameters that were painstakingly hand-tuned.

VII. FIELD ASSESSMENT

In addition to the controlled evaluation of our APEX-enabled adaptive aerial robot, we also assessed our adaptive terrestrial boundary following autonomy in a field deployment setting. Participants in this field study were asked to interact with a wheeled robot having a tilted front-facing camera, and to complete boundary patrolling tasks on McGill University's busy downtown campus.

A. Field Study Setup

This field assessment compares the efficiency of our *APEX BB* adaptive autonomy to an expert-tuned static boundary tracker, *CONST*, as well as a fully tele-operated setup, *MANUAL*. *APEX WB* was omitted because continuous parameters of this autonomy are intertwined with its discrete parameters, which prevents the *inverse optimal search particles* from analytically computing exact solutions for continuous parameter settings.

Following a briefing on the field study and a free-roam practice session, participants performed boundary patrol sessions through a fixed-length course while collaborating with the three different robot autonomy settings following a counterbalanced design. The specified course involved patrolling three distinct boundaries at different distances, and comprised specifically of a *Footpath*, a *Grass-side* sidewalk, and the *Curb* of a long and curved stretch of road.

We empirically tuned the two hyper-parameters of the APEX algorithm starting from the user study's settings, to reflect the slower pace of our wheeled robot. The final hyper-parameter values were chosen as $\alpha = 0.2$ and $\gamma = 0.7$.

B. Evaluation Metrics

Since our wheeled robot lacked an accurate outdoor localization system, we modified the *mean distance to ground truth* metric to represent the average angular distance between the recorded per-frame headings and ground truth values. These ground truth headings were generated post-hoc using optimized parameter values that were computed from hand-chosen exemplar frames for each patrol segment. In addition, the *total durations* for each fixed-length patrol segment and session were computed to assess the overall task performance for this setup. Similar to the user study, further evaluation metrics included the *AI failure ratio*, the *supervisor intervention ratio*, and the *user preference* ratings. The aggregated rankings were once again computed using the Kemeny-Young voting scheme.

C. Field Study Results

We recruited 7 participants for the field evaluation of our APEX-enabled wheeled robot. All of the participants had previously completed our controlled user study, and therefore were accustomed to the process of sharing vehicular control with our autonomous boundary tracking system.

Results in table II reveal that the *APEX BB* adaptive system consistently out-ranked the two other human-robot teams in all three patrol segments, as well as on the session-wide scale. In addition, the individual session-wide metric scores in Fig. 4 show that APEX BB scored highest for nearly all metrics and users. Furthermore, the relative preferences between the CONST and MANUAL sessions alternated under different metrics, which resulted in the reversal of their ranks on the session-wide scale.

In particular, the *AI failure ratio* and *supervisor intervention ratio* metrics ranked APEX BB consistently higher

TABLE II

AGGREGATED RANKINGS OF DIFFERENT ROBOT AUTONOMY SETTINGS FOR PATROL SESSIONS IN FIELD STUDY

Segment	Best	Second	Worst
Session-wide	APEX BB	MANUAL	CONST
Footpath (@ 1.5 ft)	APEX BB	CONST	MANUAL
Grass-side (@ 1.0 ft)	APEX BB	CONST	MANUAL
Curb (@ 0.5 ft)	APEX BB	CONST	MANUAL

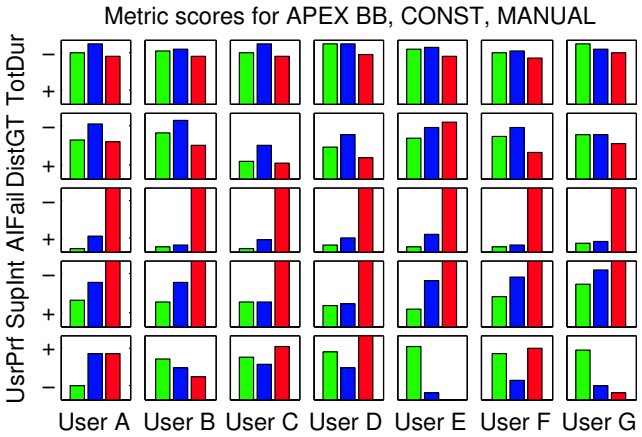

Fig. 4. Individual session-wide results in the field study, showing scores for the *total duration* (TotDur), *mean distance to ground truth* (DistGT), *AI failure ratio* (AIFail), *supervisor intervention ratio* (SupInt), and *user preference* (UsrPrf) metrics. Smaller scores are preferred on all metrics except for UsrPrf. Bars in each subplot are ordered by autonomy settings: APEX BB (green), CONST (blue), and MANUAL (red).

than CONST for our terrestrial boundary tracking system. The reduced efficiency of the statically-configured autonomy in CONST was primarily due to its sub-optimal parameter values given difficulties in conceptualizing the mapping parameters M_1, M_2, M_3, and also due to the inherent necessity for adaptive parameters to accommodate the different patrol distances. Both of these concerns were addressed by the automated parameter adaptation solution in APEX BB.

In summary, the empirical findings for both the controlled user study and the field study indicate that our *Adaptive Parameter EXploration* algorithm enabled efficient interactions between human supervisors and adaptive robot autonomies, for two distinct systems with different parameters. Results further demonstrated that APEX-enabled autonomies out-performed both expert-tuned statically-configured autonomies, as well as fully tele-operated setups.

VIII. CONCLUSION

In this work, we introduced *Adaptation from Participation* (AfP), both as a streamlined interaction paradigm for collaborative human-robot teams, as well as a computational problem. AfP shares many commonalities with the related problem of *Learning from Demonstration*, yet its distinctiveness arises due to the formulation of *human-assisted, online parameter adaptation to changing task objectives and conditions*. We implemented AfP using the *Adaptive Parameter EXploration* (APEX) algorithm, and described instantiations of APEX on two distinct robot autonomies within both aerial and terrestrial contexts. We also presented empirical evaluations of these APEX-enabled autonomies, which demonstrated higher team efficiency through increased performance and reduced user interventions. These multi-domain findings revealed that APEX consistently outperformed tele-operated setups, and further resulted in competitive task efficiency among non-expert users when compared to expert-tuned non-adaptive robot autonomies.

We are actively expanding our investigations on the generalizability of APEX-enabled human-robot teams within diverse applications. We also seek to improve this method's scalability for large numbers of discrete parameters. Finally, we plan to characterize in further depth the performance differences between the deployment of APEX on black-box autonomies and on white-box autonomies.

IX. ACKNOWLEDGMENTS

We would like to acknowledge the NSERC Canadian Field Robotics Network (NCFRN) for its funding support. We would also like to sincerely thank all of the participants in our controlled user study and our field study.

REFERENCES

[1] A. Xu and G. Dudek, "Trust-driven interactive visual navigation for autonomous robots," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA'12)*, 2012, pp. 3922–3929.

[2] 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, pp. 469–483, 2009.

[3] A. Y. Ng and S. J. Russell, "Algorithms for inverse reinforcement learning," in *Proc. of the Int. Conf. on Machine Learning (ICML'00)*, 2000, pp. 663–670.

[4] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proc. of the Int. Conf. on Machine Learning (ICML'04)*, 2004.

[5] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Survey: Robot programming by demonstration," Handbook of Robotics, ch. 59, 2008.

[6] N. D. Ratliff, "Learning to search: Structured prediction techniques for imitation learning," Ph.D. dissertation, School of Computer Science, Carnegie Mellon University, 2009.

[7] J. D. Brookshire, "Enhancing multi-robot coordinated teams with sliding autonomy," Master's thesis, School of Computer Science, Carnegie Mellon University, 2004.

[8] M. B. Dias, B. Kannan, B. Browning, E. G. Jones, B. Argall, M. M. Veloso, and A. Stentz, "Sliding autonomy for peer-to-peer human-robot teams," in *Proc. of the Int. Conf. on Intelligent Autonomous Systems*, 2008.

[9] D. H. Grollman and O. C. Jenkins, "Dogged learning for robots," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA'07)*, 2007, pp. 2483–2488.

[10] A. Dragan and S. Srinivasa, "Formalizing assistive teleoperation," in *Robotics: Science and Systems*, 2012.

[11] M. C. Vanier and J. M. Bower, "A comparative survey of automated parameter-search methods for compartmental neural models," *J. Computational Neuroscience*, vol. 7, no. 2, pp. 149–171, 1999.

[12] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach Learning Research*, vol. 13, pp. 281–305, 2012.

[13] S. Zilberstein, "Using anytime algorithms in intelligent systems," *AI magazine*, vol. 17, no. 3, pp. 73–83, 1996.

[14] A. Bachrach, R. He, and N. Roy, "Autonomous flight in unknown indoor environments," *Int. J. of Micro Air Vehicles*, pp. 217–228, 2009.

[15] M. Desai, M. Medvedev, M. Vázquez, S. McSheehy, S. Gadea-Omelchenko, C. Bruggeman, A. Steinfeld, and H. Yanco, "Effects of changing reliability on trust of robot systems," in *Proc. of the ACM/IEEE Int. Conf. on Human-Robot Interaction (HRI'12)*, 2012, pp. 73–80.

[16] P. Young, "Optimal voting rules," *J. Economic Perspectives*, vol. 9, no. 1, pp. 51–64, 1995.