Autonomous Search-Detect-Track for Small UAVs

Robert Morris[1], **Anjan Chakrabarty**[1], **Joshua Baculi**[2], **Xavier Bouyssounouse**[1] and **Rusty Hunt**[1]
1 Intelligent Systems Division
NASA Ames Research Center, Moffett Field, CA 94035
2. Department of Mechanical Engineering, Santa Clara University

Abstract

A system is described for autonomously searching, detecting, and tracking an object of interest with a small unmanned aerial vehicle (sUAV). The vehicle is given one or more areas to search. If the object is detected, the sUAV follows the target while maintaining a fixed distance and centered on its image plane. If the object is lost, the sUAV reverts to search. This problem presents many challenges in integrating planning capabilities with sensing and control. This paper describes an architecture for autonomous search and track for sUAVs. The components of the architecture include planning, image classification and image-based control for tracking. The system has been implemented in the Robot Operating System (ROS) framework using the Parrot ARDrone platform, using ROSPlan for goal planning and re-planning.

Introduction

The role of planning in autonomous robotic systems has been extensively documented. Planning frameworks allow for representing actions and goals at multiple levels of abstraction, enabling modularity and hierarchical control. Task planning is needed to transform mission goals into sequences of tasks that will accomplish them and keep the system safe. Task planners must be combined with motion planners, manipulation planners, and sensing and control systems for actuation. In this manner deliberative systems are integrated with reactive systems for dealing with noisy sensors and dynamic environments.

One major challenge in developing autonomous robotic systems is to ensure an effective interaction between the deliberative and reactive sub-systems. Specifically, the deliberative level needs to be always 'doing the right thing', given the information it currently possesses about the state of the robot and the world it inhabits. This is a core capability of autonomy and one that offers challenges in design and development.

The main contribution of this paper is a framework that provides an interface between high level planning and low level sensing and control for solving the problem of autonomously searching, detecting, and tracking (henceforth SD&T) an object of interest by a small UAV (sUAV). To motivate, consider the problem of finding, locating and tracking poachers of rare white rhinos in Africa(SaveTheRhino 2016). sUAVs with night and day sensors are deployed to search over a large area and locate poachers in a timely manner so that rangers can be dispatched to stop poachers before they strike. Some of the technical challenges in sensing, navigation and decision-making for this application include choosing where to search, distinguishing poachers from animals based on movement, and tracking an unfriendly target trying to elude capture. Uncertainty of target location makes it necessary to plan a path that guarantees sufficient coverage of an area while using data such as previous known target location to focus the search, and does not violate resource constraints. At execution time, the vehicle must rely on robust, real time image classification to detect a potential target, as well as rely on an image-based visual controller to keep the target in view. The integrated SD&T system will also involve precise communication of location to operators, a ground control system to enable human supervision and situational awareness of the sUAV activities, and potentially more than one sUAV in a coordinated operation.

The following sections of this paper describe an architecture, component capabilities, and implementation of an integrated autonomous SD&T system for a sUAV, as well as highlighting tests in simulation and in indoor field tests.

Autonomy Architecture and Component Capabilities

An operational SD&T system for a sUAV will consist of a strategic combination of human and machine intelligence. A 'fully' manual SD&T system might consist of a human sitting at a console. The search stage would consist of the human using a joystick to navigate a path for the sUAV through a promising area for search. The vehicle sensors would allow the human to identify the target of interest, and tracking could then also be performed manually.

The focus here is an integrated system where capabilities of search, identify and track are automated. The requirements for *autonomous SD&T* encapsulate a sort of 'set it and forget it' approach: at the most abstract level, the sUAV is provided with a goal to locate an object (candidate rhino poacher) and then track it until some terminating event occurs (the poacher is captured) or the sUAV is close to consuming its power resources and is required to return to base.

There are of course many variations of an SD&T problem, but this is the nominal scenario we will use here.

No assumptions are made here about whether the machine autonomy resides on the sUAV or on a ground computer (we assume here that human autonomy resides fully on the ground). Constraints on platform size and payload weight will to a large extent determine whether the component capability can be on board. It suffices here to assume that to some degree the autonomy architecture is distributed: the capabilities for autonomy reside partly on the vehicle itself and partially on a ground computer.

Figure 1 summarizes the components of the framework for autonomous SD&T described in this paper. A high level task planner alternates between two mission goals: search and track. The search goal triggers a trajectory-based control system combined with a detection system for identifying an object of interest. The track goal triggers a combined image-based visual servoing system (IBVS) and a tracker for following the target. The high level plan dispatcher ensures that the system will continuously alternate between searching as long as the object is not in view and tracking as long as it is.

The figure also shows the components of the implementation of the framework. Building upon previous work by the authors and others, the IBVS and CMT tracker are implemented as modules with ROS (Quigley et al. 2009). ROS provides the middleware for building robotic applications (primarily, in the ability to publish or subscribe to "topics" that provide state information or control lower-level behaviors). A Parrot AR.Drone quadrotor is commanded from a computer via WiFi link using the AR.Drone Autonomy ROS package (Monajjemi 2012), or in simulation using Gazebo (Koenig and Howard 2004). All simulations are run under Ubuntu 14.04 LTS 64-bit and an Intel Xeon E5-2630 @ 2.60 GHz x 17 CPU, a NVIDIA Quadro K5000 GPU, and 32 GB of RAM. Software components include ROSPlan for task planning and execution; the integrated CMT tracker and image-based controller; and the HOG-HAAR detector combined with a simple path planner. These components are described in more detail below.

Previous Work

The relevant literature on the component capabilities required for integrated SD&T is too large to adequately survey here. Rather we briefly summarize the relevant subproblems and key elements of major approaches to solving the problems.

Planning for Search

Planning for search and track (and variants such as search and rescue (SAR)) is one of the oldest problems in Operations Research. The foundations of the theory of search are Koopman's formulation (Koopman 1957) which divides the problem into two sub-problems: optimal allocation of effort (i.e. what percentage of time to spend in a given subregion); and optimal rescue track. From this core formulation models of how a target can move in an environment (e.g. a lost swimmer drifting at sea) and other environmental models, such as weather forecasts, are added, that aid in generating an optimal search path. Search trajectories are evaluated based on maximizing the probability of finding a target of interest at minimum cost (including time, manpower expended, fuel or other resources, etc.).

Planning for search is potentially challenging because it is assumed that the target is located within an area that is too large to search exhaustively; the target's location is represented as a probability distribution over subregions of the search area; and the target may or may not be moving. A typical planning cycle involves the production of a probability distribution for the object's location at the time of the next search. A trajectory uses this distribution along with a list of assigned search assets to produce operationally feasible search plans that maximize the increase in probability of detecting the object. If the search is unsuccessful, a posterior probability map for object location that accounts for the unsuccessful search and the possible motion of the object is generated, providing the basis for planning the next increment of search (Kratzke, Stone, and Frost 2010).

The work that most resembles the over all approach to search presented here is found in (Bernardini, Fox, and Long 2014). The work described in that paper is mostly complimentary to the work here. There the problem to be solved requires a search over the space of possible patterns to find one that has the best chance of finding a target. By contrast, for this paper patterns are provided as inputs to the problem being solved, and the effort here is on applying continuous re-planning to enable hierarchical control.

Vision and Control for Search and Track

SD&T is an application of object detection systems. The goal of object detection is to detect all instances of objects from a known class, such as people, cars or faces in an image. Each detection is reported with some form of pose information; here, we assume the system returns a bounding box around the object.

Object detection systems build a model from a set of training examples. Methods for inferring models are either *generative* or *discriminative*. The former create probabilistic models of pose variability and appearance; the latter create classifiers that can distinguish between images that contain an object from those that do not. Here we use a person detector based on a classification technique using grids of histograms of oriented gradients (HOG) descriptors and a linear Support Vector Machine (SVM) to classify images as person/not person; the system is described in (Dalal and Triggs 2005).

Many visual servo systems, sometimes classified as *dynamic look-and-move* (Campoy et al. 2008), use a hierarchical approach, in which a vision-based controller provides set-point inputs to a lower level position controller. The result is a visual processing loop, comprised of a feature extraction component and a visual controller, and an internal loop, comprised of a state estimator and a flight controller. The feature extractor processes the tracker data and outputs a feature vector that provide feedback to the visual controller, related to the location and size of the object on the image plane. The difference between the feature vector values and

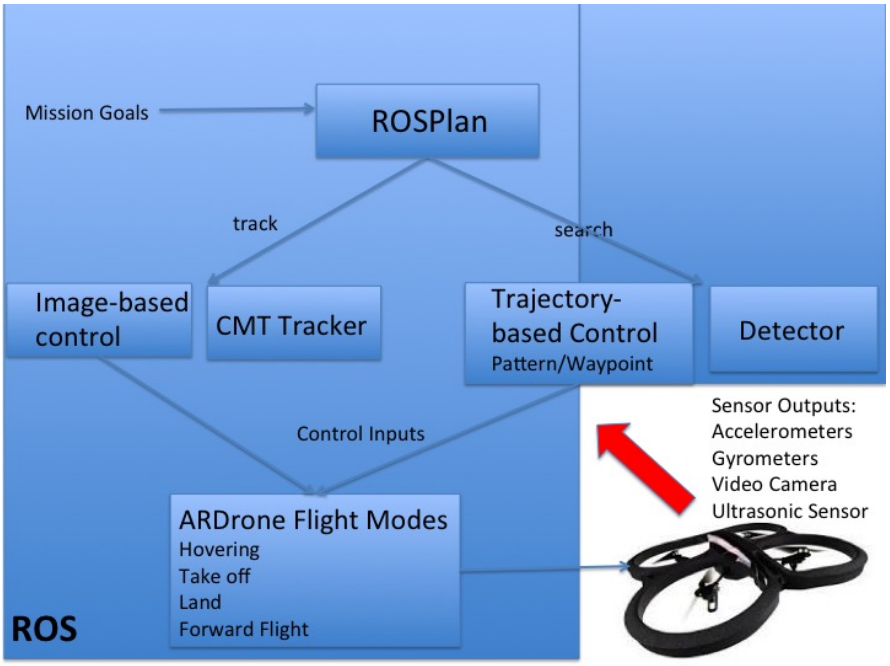

Figure 1: Hierarchical Autonomous Control using ROSPlan

the desired position and size of the features provides the input to the visual controller, which outputs velocity information to the flight controller. Meanwhile, an optical flow state estimation algorithm processes internal sensor data to maintain stability and position. In this architecture, the internal control cycle is on board the UAV, while the visual processing is performed remotely on a laptop, and communicated via a wifi connection.

Tracking is the problem of detecting an object of interest in a field of view of a camera over a period of time. More precisely, the input to the problem is a bounding box defining the set of features of an object of interest. Given a sequence of images, the problem is to identify a set of matches between features defined in the bounding box and those of the current image. These matches constitute the representation of the object of interest.

The CMT tracker (Nebehay and Pflugfelder 2015) used in the SD&T system in this paper, represents a tracked object as a set of feature correspondences of key points, relating the current position of a feature to its position in the original image. Furthermore, CMT distinguishes between static correspondences (between the current image and the original image) and adaptive correspondences (between successive images), and uses both kinds of correspondence to update its object model. Adaptive correspondence is better at handling different object appearances due to deformation, whereas the static model is better at handling the reappearance of the object after occlusion. Second, CMT introduces a tolerance parameter in order to allow the set of correspondences between frames to be robust to changes due to deformations. The CMT tracker uses heuristic estimates to generate values related to scale and rotation of the bounding box. These values, in addition to an estimate of the center of the tracked object, are the outputs of CMT.

For more on the IBVS controller and tracker used in the system described here, see (Chakrabarty et al. 2016) or (Pestana et al. 2014).

SD&T Planning and Execution Using ROSPlan

As stressed in (Nau, Ghallab, and Traverso 2015), carrying out actions in a plan in robotic applications requires continuous on-line planning and deliberation and involves a hierarchically ordered collection of modules to carry out specialized tasks. In SD&T, search, detection, tracking and navigation are specialized behaviors that need to be integrated into an autonomous system that stays safe and accomplishes high-level mission goals.

As noted earlier, planning for search is potentially a computationally challenging optimization problem involving the efficient use of resources to maximize the probability of finding a target of interest. To formulate a search planning problem, the domain model must contain a way to specify one or more subareas to be searched. In addition, the domain should contain a collection of actions that describe a comprehensive search of an areas of interest, as discussed in (Bernardini, Fox, and Long 2015).

In addition to these planning requirements, there are a number of requirements for effective execution of plans. When a target of interest is found, search must stop and tracking started. From a planning perspective, this transition can be viewed as a change in plans, from one consisting of following a trajectory to one consisting of keeping a target in view (tracking). This transition from trajectory-

based to tracker-based control of the vehicle should be done in a timely manner to avoid logins the object. More generally, the system should support an "alternating behavior" between search and track, in which the target is repeatedly lost and found.

Planning and execution using ROSPlan

We use ROSPlan (Cashmore et al. 2015) to automate the planning process for SD&T. ROSPlan supports PDDL activity planning, consisting of a domain defining the actions used for planning, and a problem file that contains a description of the initial state and goal(s) of the plan. ROSPlan supports different planning solvers; for this work we use the temporal planner POPF, that is included in the ROSPlan installation. ROSPlan contains a plan executive that dispatches plans. ROSPlan uses the ROS message passing infrastructure to dispatch actions and receive feedback from the low-level controllers. The PDDL model and problem instance are also stored as a ROS nodes called the Knowledge Base (KB) and the Problem Generation nodes respectively. The KB is continuously updated from sensor data through the ROS interface. ROSPlan validates the current plan against the current model and allows replanning to occur in the case of action failure.

ROSPLan model for SD&T

The PDDL model developed for SD&T contains predicates *found* and *trackdone* which correspond to the goals of the stages of the activity. These goals are effects of actions *search* and *track*. Intuitively, the goal *found* is true if a target is found, the area of search has been exhausted, or some other condition holds (like running out of battery charge). The precondition for search is *airborne*, and the effect is *found*; conversely, the precondition for track is *found* and the effect is *trackdone*. Following previous formulations of the problem, the search phase is interleaved with trajectory planning whereby the drone can be dispatched to a a region based on predictive models of where the target might be. This phase is accomplished through a *goto-waypoint* action. (The difference between executing a search action and executing a goto-waypoint action is that the latter does not assume the detection system is turned on.)

PDDL actions map directly to ROSPlan action components that refine the actions into low-level commands. We use a procedure-based approach to action refinement, with hand-written procedures written in C++, reminiscent of systems for reactive planning such as RAP (Firby 1987). The PDDL domain actions map to the following two methods:

procedure SEARCH(path,object)
  while object not detected and battery OK **do**
    follow path
  Stop and hover
  Add found fact to KB
  if battery not OK **then**
    Add home waypoint goal to KB
  else // object is found
    Add tracked goal to KB
    Re-Plan to track

procedure TRACK(object)
  Start tracker
  Start IBVS controller
  while confidence $>$ threshold and battery OK **do**
    follow object
  Add trackdone fact to KB
  Stop IBVS controller
  if battery not OK **then**
    Add home waypoint goal to KB
  else // object is lost
    Add found goal to KB
    Re-Plan to search

We distinguish among 4 types of low-level actions that collectively implement a refinement of *search* and *track*:

- Accessing state information: the conditions on the while loop access state information (via the implementation of publishers or subscribers to ROS Topics) to determine whether the object has been found or lost and whether power resources are sufficient to continue the mission. Specifically, two state variables *PersonDetected*, set by the detector, and *Confidence*, set by the tracker, are monitored by the action components. If *PersonDetected* is true and *Confidence* is above a certain threshold, then the system is in a tracking state; otherwise, the system is searching.
- Initializing sub-systems for sensing or control: starting and stopping the tracker and IBVS controller.
- Drive the sUAV: following a path constitutes a high level trajectory planner and controller
- Re-planning: If the object is found then re-planning for tracking is started by adding and removing goals. The Track method is complementary.

In this way, manipulation of the knowledge base by adding or removing goals enables continuous alternating behavior required by SD&T.

Experiments

In this section we summarize the experiments conducted on the ROSPlan-based SD&T system described in this paper. The tests here are meant to verify the feasibility of the ROSPlan approach to continuous planning for SD&T. The performance metrics for feasibility include correctness (the planner always responds properly to the presence or absence of the target in its image plane by placing the system into a search or track behavior mode), and timeliness (ROSPlan responds to a change in the sUAV's sensor outputs quickly enough to ensure continuous, stable operations).

Experiments in simulation using Gazebo (Figure 2) and in an indoor testing facility at Ames Research Center (Figure 3) were conducted. The simulation allowed for quick debugging and adjustment of system parameters that influence performance. Some of these parameters include the expected size and distance of the target (used by the IBVS controller) and the sampling rate of the state variables within the ROSPlan action interface code.

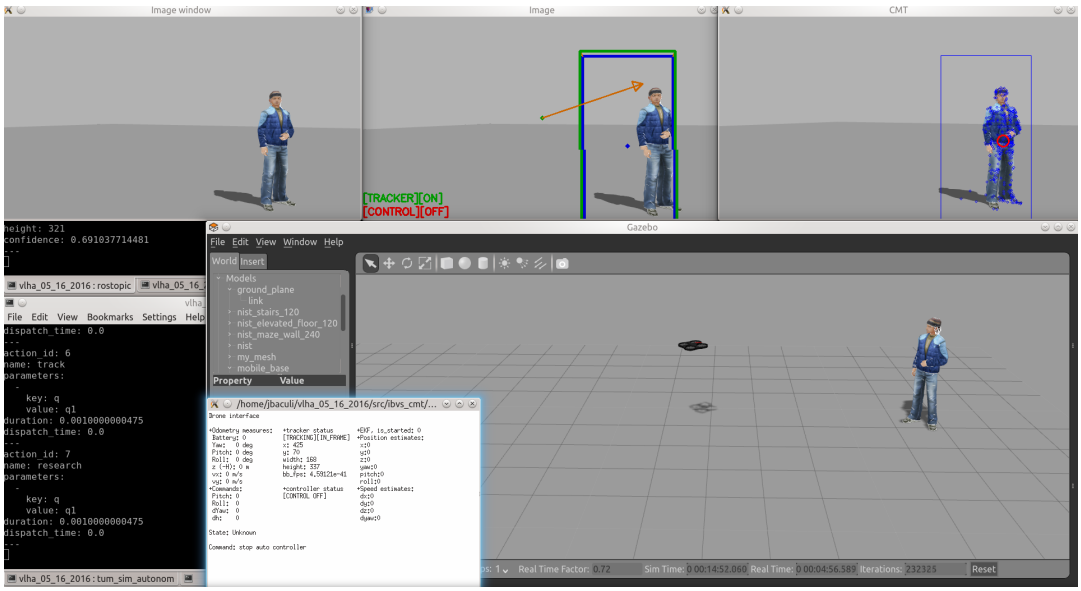

Figure 2: Simple Gazebo environment for testing SD&T in simulation. The upper middle panel shows the bounding box around the person target maintained by the CMT tracker, and control input (orange arrow) to visual controller. The right panel shows the bounding box constructed by the HOG detector. The black windows on the left stream the outputs of the relevant ROS topics. The larger panel shows the sUAV and target.

Figure 3: Testing ROSPlan for SD&T in an indoor facility at NASA Ames Research Center.

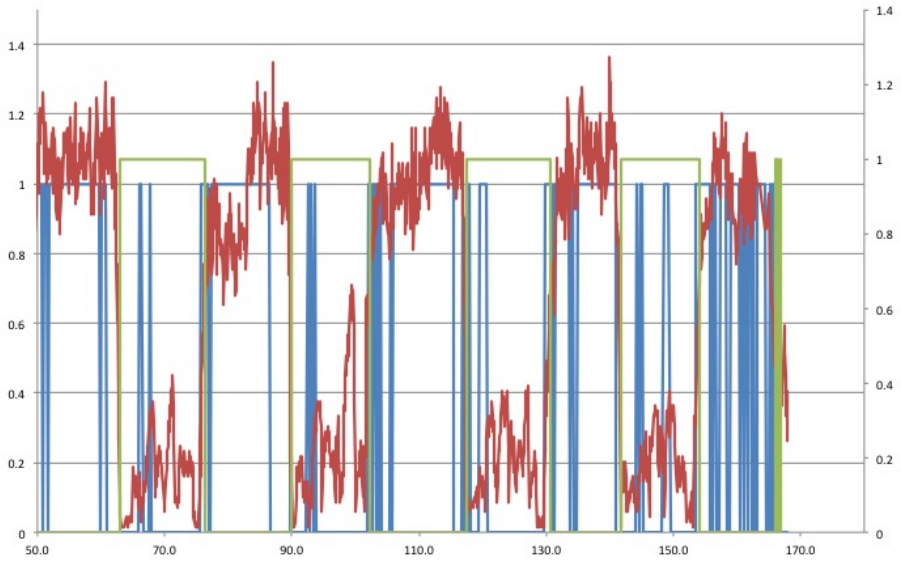

Figure 4: Interactions between ROSPlan Action Dispatching and Sensing Outputs. ROSPlan Action Dispatch Variable (Green Line) alternates between posting a search goal (1 on right y axis) or a track goal (0) on the ROSPlan KB, based on values obtained by the tracker (Confidence, red line) and the detector (PersonDetected, blue line, where 1 (left y axis) = PersonDetected true).

We deliberately chose a fairly controlled indoor environment in order to isolate the interactions between the continuous planning framework and the underlying sensing and control frameworks, thereby demonstrating the feasibility of a hierarchical autonomy architecture based on ROSPlan. Consequently, however, there is little in the way of assurance that the system is robust to realistic dynamic environments or noisy sensors. Future outdoor field tests are planned, using more powerful platforms than the AR Drone, in order to improve robustness.

In a typical run of the system, a human target would begin out of sight of the sUAV. The sUAV would take off and begin a pattern maneuver (such as a square pattern) in search mode. Once the human target is found, we studied the response time of the system to transition into track mode. Once in track mode, the target would first move in a slow linear walk to demonstrate simple tracking behavior. At some point, the target would take evasive maneuvers until he was out of sight of the sUAV. We then could observe the transition back into search mode, which consisted of following another pattern search (typically a simple rotation). This alternating behavior of search and track was typically repeated many times in a single run.

Figure 4 illustrates the run time performance in the indoor facility, illustrating the alternating search-track behavior induced by ROSPlan. In the figure, changes in the state of three ROS topic variables over time (in seconds) are displayed: *confidence*, published by the tracker, *person detected*, published by the detector, and *action dispatch*, used within ROSPlan. The action dispatch variable is represented as a step function with two values: 0 (for track) and 1 (for search); depending on the value of this variable, one or the other of the procedures described above are executing.

In the execution snippet in the figure, the sUAV starts in track mode (ActionDispatch=0, green line). Recall that ROSPlan triggers a new plan for tracking by adding a goal to the KB, while invoking the CMT tracker and IBVS controller. The CMT tracker inherits the bounding box generated by the the detector, and initializes the *confidence* state variable, which it updates during tracking. If the confidence variable goes below a certain threshold (0.6 in the experiments here) ROSPlan concludes that the object has been lost, and a new search goal is placed on the KB. For example, at roughly the 62nd second of this run, the ActionDispatch variable is set to 1 as a response to the *confidence* variable reaching its threshold.

To aid in robustness and stable behavior, we found through experiments that it is best to re-trigger tracking when *both* the detector finds the human figure and the confidence variable exceeds the desired threshold. For example, roughly around the 95th second, the tracker has a confidence value greater than the threshold, but the detector has not found a target. The tracker has in fact falsely identified a target to track in the image plane at this point. The plan dispatcher, however remains in search mode, waiting for the person detector to register the target. Conversely, there are

times (such as around second 135) in which the detector shows uncertainty between target and no target, and yet the tracker shows confidence in its target. In this case we allow the tracker to direct the plan dispatcher.

To summarize, the experiments conducted validated the feasibility of using ROSPlan as the basis for a hierarchical architecture for autonomous control. We were able through successive refinements of operational parameters to achieve stable and correct behavior in a controlled environment using an AR Drone. SD&T is a difficult challenge for autonomy because of the need for high-level management and control of subsystems that are themselves comprised of complex sense-control loops. We showed that a high-level decision-maker based on continuous planning fulfills the requirements for an effective system for SD&T.

Future Work

We are exploring improvements to the autonomous SD&T framework in parallel on a number of fronts. On the platform side, we are in the process with replacing the ARDrone with higher-cost platforms with better flight stability and more advanced sensing units. This would allow us to incorporate different sensors that would allow searching in a wider range of realistic outdoor environments. In turn, this would allow us to pursue a more phased approach to search, whereby, say, a motion detector is used to find a moving target, which triggers the vehicle to move towards the object in order to identify it.

Secondly, we are in general interested in more rigorous testing in more realistic environments, whether actual or in simulation. On the simulation side, we are exploring the use of the Modular OpenRobots Simulation Engine (MORSE) combined with the Blender Game Engine for more realistic environments. An outdoor testing facility at Ames will also be used. In general, more scenarios for demonstrating autonomy in more realistic conditions are needed.

Finally, on the planning side, we are interested in expanding the current model and planning system in a number of ways. First, we are interested in incorporating various optimization criteria into planning, following traditional methods for solving Search-and-rescue problems. Secondly, we would like to devise ways to handle some of the uncertainty at planning time, for example, through contingent planning, which is supported within ROSPlan.

Conclusion

An approach was presented for combining simple task planning with sensing, motion planning and reactive control to achieve autonomous SD&T. The developed system is based on ROS and the ROSPlan planning framework, integrating state-of-the-art algorithms for object recognition and object tracking. The ability to coordinate complex components to enable continuous search and tracking using a sUAV illustrates the challenges and rewards of autonomy. In both simulation and field experiments using the AR Drone the ability of deliberative and reactive systems to work together to achieve high-level goals was demonstrated.

Analogous to the way toy problems like Blocks World provided testbeds for the development of search algorithms which could solve real world problems, ROS and the AR Drone together provide a toy testbed for designing architectures for robotic autonomy that have the potential for solving real world problems like SD&T. In addition, as shown here, ROSplan adds further infrastructure for enabling complex behaviors through hierarchical control.

Acknowledgements

The authors thank Michael Cashmore, Sara Bernardini, Jindrich Vodrazk and Jesus Pestana for helpful discussions. Also thanks to the reviewers to helping identify gaps in the discussion in the original draft.

References

Bernardini, S.; Fox, M.; and Long, D. 2014. Planning the Behaviour of Low-Cost Quadcopters for Surveillance Missions. In *Proceedings of the 24th International Conference on Automated Planning and Scheduling (ICAPS-14)*.

Bernardini, S.; Fox, M.; and Long, D. 2015. Combining temporal planning with probabilistic reasoning for autonomous surveillance missions. *Autonomous Robots* 1–29.

Campoy, P.; Correa, J. F.; Mondragón, I.; Martínez, C.; Olivares, M.; Mejías, L.; and Artieda, J. 2008. Computer vision onboard uavs for civilian tasks. In *Unmanned Aircraft Systems*. Springer. 105–135.

Cashmore, M.; Fox, M.; Long, D.; Magazzeni, D.; Ridder, B.; Carrera, A.; Palomeras, N.; Huros, N.; and Carreras, M. 2015. Rosplan: Planning in the robot operating system. In *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling*, 333–341. ICAPS.

Chakrabarty, A.; Morris, R.; Bouyssounouse, X.; and Hunt, R. 2016. Autonomous indoor object tracking with the parrot ar. drone. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, 25–30. IEEE.

Dalal, N., and Triggs, B. 2005. Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1.

Firby, R. 1987. An investigation into reactive planning in complex domains. *Proceedings of AAAI*.

Koenig, N., and Howard, A. 2004. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2149–2154.

Koopman, B. 1957. The theory of search, part iii: the optimum distribution of searching effort. *Operations Research* 5:613626.

Kratzke, T. M.; Stone, L.; and Frost, J. R. 2010. Search and rescue optimal planning system. *Proceedings of the 13th international conference on information fusion*.

Monajjemi, M. 2012. Ardrone autonomy : A ros driver for ardrone 1.0 & 2.0.

Nau, D.; Ghallab, M.; and Traverso, P. 2015. Blended planning and acting: Preliminary approach, research challenges. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Nebehay, G., and Pflugfelder, R. 2015. Clustering of static-adaptive correspondences for deformable object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2784–2791.

Pestana, J.; Sanchez-Lopez, J. L.; Saripalli, S.; and Campoy, P. 2014. Computer vision based general object following for gps-denied multirotor unmanned vehicles. In *American Control Conference (ACC)*.

Quigley, M.; Gerkey, B.; Conley, K.; Faust, J.; Foote, T.; Leibs, J.; Berger, E.; Wheeler, R.; and Ng, A. 2009. Ros : an open-source robot operating system. In *EEE International Conference on Robotics and Automation (ICRA 2009)*.

SaveTheRhino. 2016. The use of drones in rhino conservation. *https://www.savetherhino.org*.