CANINE: A Robotic Mine Dog

Brian A. Stancil[a], Jeffrey Hyams[a], Jordan Shelly[a], Kartik Babu[a], Hernán Badino[b], Aayush Bansal[b], Daniel Huber[b], Parag Batavia[a]

[a]Neya Systems LLC, 12330 Perry Highway, Suite 220, Wexford, PA, USA 15090; [b]The Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA USA 15213-3890

ABSTRACT

Neya Systems, LLC competed in the CANINE program sponsored by the U.S. Army Tank Automotive Research Development and Engineering Center (TARDEC) which culminated in a competition held at Fort Benning as part of the 2012 Robotics Rodeo. As part of this program, we developed a robot with the capability to learn and recognize the appearance of target objects, conduct an area search amid distractor objects and obstacles, and relocate the target object in the same way that Mine dogs and Sentry dogs are used within military contexts for exploration and threat detection. Neya teamed with the Robotics Institute at Carnegie Mellon University to develop vision-based solutions for probabilistic target learning and recognition. In addition, we used a Mission Planning and Management System (MPMS) to orchestrate complex search and retrieval tasks using a general set of modular autonomous services relating to robot mobility, perception and grasping.

Keywords: CANINE, TARDEC, autonomous, search, retrieve, perimeter, surveillance, mine

1. INTRODUCTION

Use of real canines in combat is a challenging proposition from both a logistical and effectiveness standpoint. In addition, success rates of these canines can be varied, due to differences in training and individual temperament. Small mobile robots that can perform the same tasks as mine and sentry dogs, including detecting intruding adversaries and learning and recognizing small objects would provide a more consistent level of performance while decreasing the logistics necessary to maintain and care for a dog.

Neya Systems, LLC has developed a robotic system (Figure 1) which can be used collaboratively with a human operator to locate, pickup and relocate objects of interest. This technology has applicability to defeating static dangerous objects in the battlefield such as Anti-Personnel Mines, Improvised Explosive Devices, and other destructive battlefield explosive devices. This basic set of capability was designed and developed within the context of the TARDEC CANINE competition which was held at Fort Benning, Georgia in April 2012. Within this contest, six teams competed to locate various objects of different shapes, sizes and colors within a set search radius of the operator and relocate them to a designated location. While the competition itself was framed as a search and retrieve challenge, these same technologies can be used to relocate hazardous objects to any desirable location including safe locations for detonation or disposal.

Figure 1. Neya's robot picking up a target object during the final TARDEC CANINE competition at Fort Benning, GA

This paper describes the platform and software developed for the CANINE competition. Section 1 describes the competition rules governing the design of the robot as well as our system level approach. In section 2, we provide an overview of the robotic platform built specifically for the competition as well as the sensors and gripper hardware used. In section 3 we present the mission-based software architecture which was used to orchestrate the autonomous behaviors in each of the competition challenges. In section 4 we present the vision-based perception algorithms used to learn and detect the location of target objects, obstacles and the operator. In section 5 we discuss the vision-based user interface by which the operator instructed the robot during the competition. Lastly in section 6 we discuss the program results and discuss lessons learned and anticipated future research.

1.1. CANINE Program Overview

The focus of the 2012 TARDEC CANINE program is to research and develop advanced robotic collaborative behaviors. The program was structured as a competition involving six teams involving both industry and academic which was held at Ft. Benning, Georgia as part of the 2012 Robotics Rodeo. The competition was structured as a series of six discrete challenges which involved finding and retrieving objects in increasingly complex scenarios. The challenges required the robot to find thrown or pre-placed objects within areas that contained distractor objects of the same color or shape, avoid mobile and statically moving human obstacles, and locate the operator. The operator was restricted to non-digital communication only, precluding the use of wireless Operator Control Units (OCU) or remote control devices.

Teams were awarded points for successful completion of the challenges. In the event that multiple teams successfully completed the same number of challenges, completion time was used to determine the winner.

1.2. Solution Overview

Neya's CANINE robot includes all of the necessary sensors, gripper hardware and autonomous software necessary to perform each of the search and retrieval challenges within the CANINE competition. Our software solution has multiple tiers of software including driver level hardware interfaces, autonomous services (supporting perception, navigation and grasping), mission management services and a vision-based operator interface. Although the robot can be controlled manually using a wireless remote controller, all primary control is performed through the operator interface which allows the operator to select operations on a side-mounted display using hand gestures. Through this interface, the operator selects pre-planned missions, which correspond to each of the six designated challenges in the competition.

Once a mission has been selected, the robot begins executing behaviors autonomously according to the mission plan. The mission plan details specific behaviors such as interactions with the user (as with showing the robot a target object) or driving a directed search route while looking for candidate target objects. It also designates specific contingency plans which are activated when unexpected events or problems occur. Due to the scoring methodology of the challenges, the missions were designed with alternative search patterns if the target object was not discovered within the initial search area and the termination criteria for each mission consisted of either a successful retrieval of the target object or a mission abort commanded by the operator.

2. PLATFORM OVERVIEW

The platform was designed specifically for this challenge based off of a robot chassis initially designed by the National Robotics Engineering Center (NREC) for the DARPA Learning Applied to Ground Robots (LAGR) program. We outfitted the base chassis with an electronics enclosure containing a 2.7GHz dual quad-core processor, power distribution hardware, an inertial measurement sensor (fiber-optic gyroscope), and an 802.11 wireless router. A single motor controller is used to drive the two electric wheel motors and collect wheel speed from wheel encoders mounted on the drive shaft. For the competition, we limited commanded linear speed to 3 meters per second and added a remote ESTOP circuit which overrides autonomous control.

Figure 2. Neya's CANINE robot includes a mobile platform, a sensor mast with articulated LADAR and color cameras, and a front mounted gripper for picking up and placing objects on the ground.

The sensor mast stands 62 cm above the top of the chassis (109 cm above the ground) and includes a tilt-articulated 2D LADAR and two pan/tilt articulated color cameras. We also built a custom front-mounted gripper includes two articulated degrees of freedom (up/down and open/close). For the competition, the gripper pads were modified with foam padding and metal teeth to maximize surface friction with the target object and lift objects out of thick grass.

3. BEHAVIOR SEQUENCING AND OPERATIONS

Behavior sequencing for the CANINE competition was handled through Neya System's Mission Planning and Management System (MPMS). The purpose of the MPMS is to separate the mission-specific tasking and instruction logic from autonomous software such that a library of well-defined and independent autonomous services (capabilities) can be combined in various ways to implement a wide range of missions. The MPMS includes 1) a domain-specific Tactical Scripting Language (TSL) for describing mission task templates, autonomous service interactions and process flow 2) a set of tools for developing and testing mission specifications and 3) a Mission Management Service (MMS) which resides on the robot and coordinates available autonomous services to execute the mission tasks. This software architecture is shown in Figure 3.

There are several benefits to this approach. By using the MMS, we are able to run a common set of general services between each of the six challenges. The only thing that changes between each challenge is the mission-specific tasking that the CANINE user interface (UI) passes to the MMS. This tasking, specified in ASCI-readable TSL, details the specific mission plan for each challenge and directs the robot to perform custom operations including learning object color/shape, finding objects and returning to dynamically defined operator location points. In addition, this architecture avoids the need for monolithic software processes with complex and interdependent state machines. Instead, all mission logic is represented in the TSL-encoded mission plan and the MMS maintains the mission state. To implement the mission plan, the MMS interacts with each service through a published set of discrete interfaces. This modular service-oriented approach is easier to unit test during development and provides a straightforward methodology for performing mission-level collaborative service testing and execution as well.

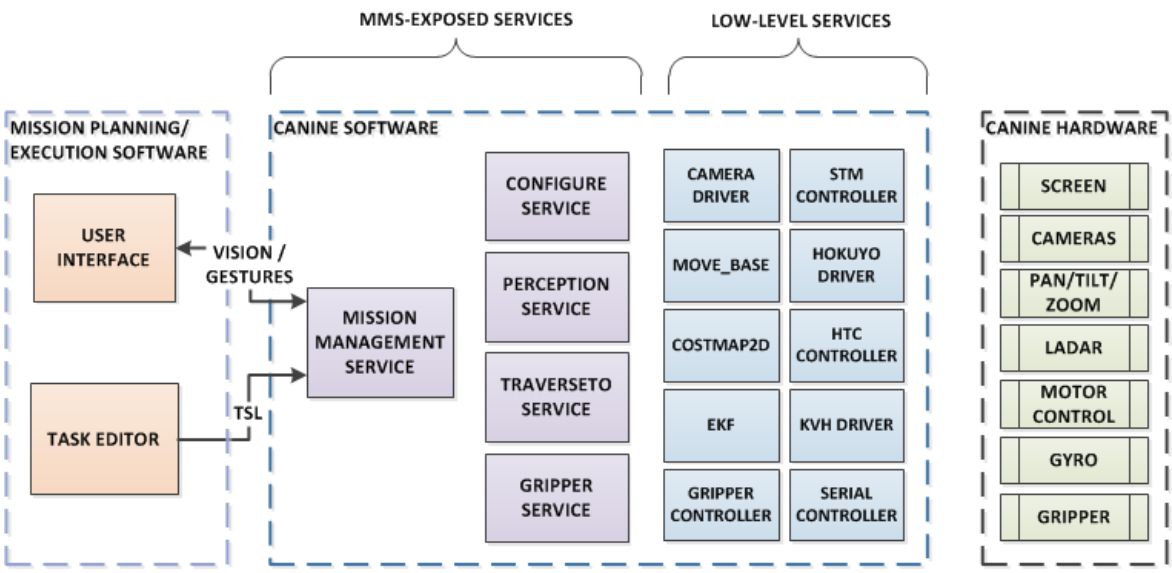

Figure 3. Neya's software architecture includes mission planning and operator interfaces, mission planning and autonomous services, and driver level software which interfaces directly with hardware components.

There are three steps to executing a mission within the MPMS architecture. Initially a set of *task templates* are generated which contain the general service requirements and interdependencies to perform the tasks in the mission. For the CANINE mission, we developed a task template for each of the six challenges. Secondly, at run time each task template is seeded with specific mission parameter values which include actual waypoint coordinates, characteristics of the target object, and the operator location. These parameters represent run-time data which will change between multiple implementations of a particular mission and cannot be specified in the mission template ahead of time. Lastly, the mission is executed according to the specified plan. Although the CANINE rules precluded any wireless digital interactions with the robot during the challenge, the mission plans included specific interaction with the human operator at the beginning of each challenge to learn the target object and also visual interaction with the operator at the end of the challenge when the robot looked for the operator to return the object.

3.1. Mission Planning using Task Templates

A mission task template contains a set of one or more sequential phases which define functional milestones within the challenge (ie. learn object, locate object, grab object). Within each phase of the mission, the task template defines the necessary services, signals and process flow required to accomplish the success criteria for each phase and ultimately, the task itself. For the initial challenge in the CANINE competition, the phases of the mission task were:

- **Search Phase**: drive a pre-defined relative waypoint path while searching for Objects of Interest (OOI).
- **Inspect Phase**: given a general location of an OOI, approach that location and determine if it definitively matches the target object
- **Pick-up Phase**: align the robot and gripper at a specific target object location and pick up the object.
- **Recover Phase**: return with the object to the operator location and drop the object within the starting circle.

While only one phase is operational at any time, the process flow from one phase to another depends on specific events, defined within the TSL, that occur during the mission. For example, the *inspection phase* does not occur until an OOI has been identified within the *search phase*. Similarly, events can be specified which handle failure cases and implement recovery plans. For instance, a failure in the *pick-up phase* results in the robot opening the gripper, backing up and then returning to the *search phase* to re-acquire the location of the object which may have occurred during the failed pick-up.

3.2. Autonomous Services used in the Competition:

The autonomous services for the CANINE competition are divided into two classes: those that interact with the MMS at the mission level, and those that represent underlying highly integrated capability. *MMS-visible* services are modularly defined and expose clear interfaces which can be referenced directly within TSL. These services include general high level capability for driving waypoint trajectories, using the gripper to pick up objects and searching for the target object. The specific MMS-visible services developed for the CANINE competition are summarized in Table 1.

Table 1: The services which are used by the mission planning layer to develop autonomous behaviors

MMS-visible Services used for CANINE Competition	
Configuration Service	Used to calibrate the cameras, pan/tilt mechanisms and gripper
Perception Service	Used to detect objects, analyze objects and find the operator
Gripper Service	Used to pick up objects and check for successful pickup
TraverseTo Service	Used to traverse search paths and go to specific waypoints

The MMS-visible services make use of a wide variety of lower level services which include mapping, localization, motion planning, sensor drivers, and drive controllers. We made significant use of Willow Garage's Robot Operating System (ROS) components including the *move_base* package for waypoint following with obstacle detection and avoidance. The ROS software includes a costmap representation, a graph-based mobility planner and a reactive obstacle avoidance trajectory controller similar to [1]. We built upon this implementation by adding specific global map representations for target occupancy and several ROS-compatible drivers for the fiber-optic gyro and the various motor controllers.

4. PERCEPTION AND OBJECT RECOGNITION

The target object recognition subsystem uses a combination of stereo and monocular image-based vision algorithms. The recognition process (Figure 4) is organized primarily as a sequence of filters that operate on the color camera images. Starting with an appearance model of the background, the foreground regions are extracted and grouped into blobs. These blobs are then filtered based on approximate size, using range estimates from stereo. Finally, depending on the recognition task, we apply filters based on either the target's appearance model, shape model, or both, to eliminate non-target blobs. Any remaining blobs are passed to the tracking sub-system as potential target objects.

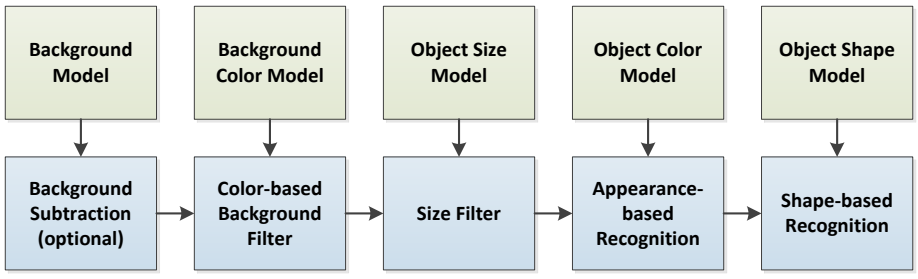

Figure 4. The perception module is implemented as a series of filters, each of which uses a model that is learned online prior to object detection and retrieval.

Ground/Non-ground Segmentation. Sparse stereo vision is used to estimate the ground surface for detecting low-lying objects and obstacles. We use a custom, correlation-based pyramidal stereo algorithm to compute sparse range estimates at strong vertical edges. Given a set of 3D stereo points, we use the RANSAC algorithm [2] to estimate the ground surface, which we assume to be locally planar. We classify points lying more than a fixed distance above the ground surface as potential obstacles. This approach can detect low-lying and overhanging obstacles that cannot be seen by the laser scanner, which only observes points at a fixed height above the ground.

Background Subtraction. The scene is observed while the robot is waiting for the object to be thrown, and the change detected after the object lands indicates potential target locations even at distances beyond the capabilities of the recognition algorithms. The algorithm handles moving objects in the scene by forming a composite image over a fixed time period, which is constructed from the median pixel value in the HSV color space, treating each channel independently. A composite image is created before and after the object is thrown, and any pixels with an average difference across the H and S channels greater than a fixed threshold are labeled as changed. The V channel is ignored due to its sensitivity to changes in illumination. The detected changes are clustered into connected blobs.

Color-based Background Filter. At runtime, the robot learns an appearance model for the background, which is used to segment the background while moving. Our approach is based on the method described in [3]. The background is modeled using a normalized two dimensional (2D) color histogram in the HSV color space. Only the hue and saturation components are used, which makes the model more robust to shadows. We use a generative model in which pixels are classified as background if the corresponding histogram bin is above a given threshold. We also evaluated RGB and Lab color space models, but those were less discriminative.

Figure 5. Color-based background filter. The model learns the appearance of grass based on its color (left), allowing uniquely colored objects to be segmented (center) and overlaid on the original image (right).

Size Filter. The blobs are filtered based on their size relative to the known size of the target object. The distance to the blob is estimated using stereo. We average the estimated range over the nine pixels at the center of each blob using the stereo algorithm described above. We use a large window size because the target objects were specified to be textureless. If the targeted stereo estimation fails, we fall back to the ground plane estimate to determine the range based on the blob's image position.

Appearance-based Recognition Filter. The appearance-based recognition algorithm uses the same color histogram algorithm used for color-based background classification. We use only color because the target objects are textureless. Since the target is unknown in advance, we present the object to the robot at runtime by holding it in front of the robot and rotating it to present all sides. The operator wears a black glove to facilitate automatic segmentation of the hand and forearm. Target objects are distinguished from non-target objects of the same color using the same threshold-based classifier described above.

Shape-based Recognition Filter. Surface shape information has been used to effectively recognize large databases of objects [4] and object classes [5] in real data under conditions more challenging than the proposed scenario. Our shape-based recognition algorithm uses features extracted from the object's silhouette. As with the appearance-based recognition, the robot learns the target object model at runtime immediately before the object is thrown. We use the color-based background filter to create a binary image of the target object and, implicitly, its silhouette. The shape feature for a silhouette is computed by traversing the silhouette contour and recording the angle between each triplet of boundary points at a fixed distance apart. For scale invariance, the distance between subsequent points is a fixed

proportion of the total boundary length. A descriptor for the silhouette of the shape is built by computing at each point, the angle of three equally spaced boundary points. The distance is proportional to the total length of the boundary and kept constant for a given silhouette.

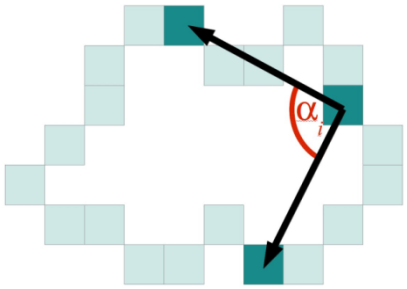

Figure 6. The silhouette descriptor uses scale and image-plane rotation invariant boundary angle histograms to describe the visible cross-section shape of objects

As the object is presented to the robot from different viewpoints, new descriptors are computed and added to a database of descriptors provided they are sufficiently different (in terms of the L2 norm) from the descriptors already in the database for the specified model. After all unique views of the object are obtained, additional views of the object are used to learn a model of the distribution of shape similarities that would be encountered during recognition. We compute the minimum L2-norm distance between the silhouette descriptor from each new view to all the descriptors in the database. These distances are then used to define the measurement PDF (P(descriptor histogram | object). The PDF is then transformed into its corresponding CDF. At runtime, a blob is recognized as a match if its difference measure cumulative probability is below a given threshold. This process is shown in Figure 7. For robustness, the algorithm reports a detection only after several matches are found in a short time interval, which allows the robot to observe the object from different viewpoints.

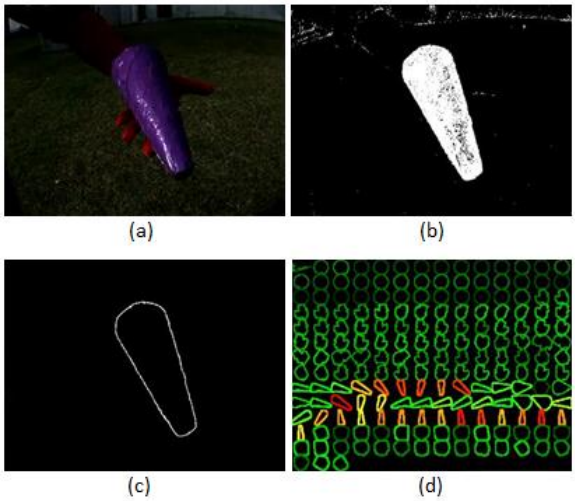

Figure 7. Shape-based recognition. The target object is presented to the robot (a). The background is removed (b), and the silhouette is estimated (c). The shape is matched to a database of views of the object (d). The color of the outline indicates the strength of the match – green = low, yellow = medium, red = high. Similarity strengths for non-target objects are also shown for comparison.

Any blobs that pass through all of the active stages of the filtering process are passed to the tracking module. The specific filters that are active are task-dependent. In the CANINE competition, some tasks required only appearance-based recognition, some required only shape-based recognition, and some required both. In practice, appearance-based recognition was always used. The background subtraction filter was ultimately not used in the competition because it was not necessary.

Operator Recognition Filter. Our perception system used color and shape to determine the location of the human operator. During points in the competition when the robot was required to find the location of the operator, each frame captured by the camera was segmented using the same color classification algorithm used to detect the targets. This color classification step resulted in binary images with a number of discrete contiguous blobs representing colors that fall outside of the learned background color histogram. We filtered the blobs based on several geometric criteria including the total number of pixels, the absolute size of the blob (in meters) and the general shape of the blob.

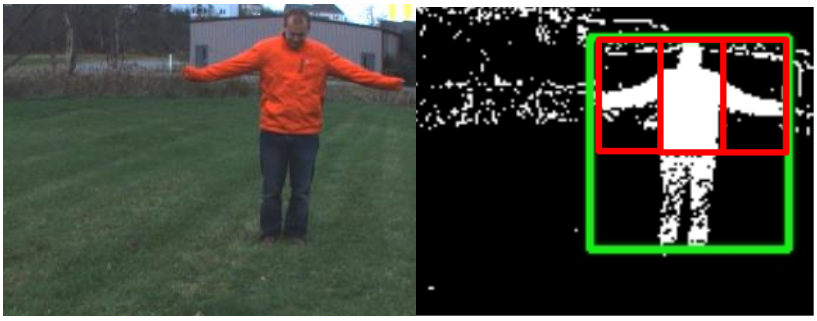

Figure 8. The operator detection filter uses background segmentation in combination with shape heuristics to identify static gestures. In the right figure, this the operator blob is outlined in green and the three vertical torso components are outlined in red.

The shape of the blob is defined heuristically to represent the profile of the operator with their arms outstretched. To do this, we divide each blob of approximate human size into three equal sized vertical partitions. If the aspect ratio of the entire blob is near 1:1, the blob is assumed to contain the legs of the operator and only the top half of the blob is used for gesture recognition. The classifier passes the blob as an operator if the top half of each of the side partitions and the full center partition contained over a threshold percent of non-background pixels (Figure 8). This pixel allocation of non-background pixels corresponds to the operator's arm and center torso when wearing a jacket when they are within a specific range of the camera. To further reduce false positive observations, the classifier requires multiple frames of the observed operator before registering that the operator has been found.

5. HUMAN ROBOT INTERFACE

The CANINE competition rules disallowed digital or wireless control as well as typical mouse, keyboard and touchscreen interactions for operator control and feedback during each of the challenges. Although most of the behaviors required for each challenge were pre-planned as part of the mission plan, our system does require operator interaction for a small set of basic operations including configuring/calibrating the robot initially, starting missions and presenting the robot with the target object.

Figure 9. The user interface uses a menu overlay displayed on the left camera image (right image) to allow the user to select operations with their hand on an externally mounted screen (left image)

For these operations, we developed a vision-based user interface (UI) which displays a menu overlay on the left camera image (Figure 9). The UI uses the same underlying background filter to differentiate the operator's hand from the camera background, allowing them to use their finger to emulate a mouse on the menu. To minimize the chance of noise affecting the menu system, our operators use a glove with a color not present in the background image. However, in practice, the operator's skin color is often easily segmented from the background. The menu buttons operate on a timer basis, requiring the operator to hover over the desired button for a few seconds to activate the button.

6. RESULTS AND LESSONS LEARNED

The system and algorithms developed under this program culminated in a competition hosted by TARDEC at Fort Benning, GA in April 2012. Neya's robot competed on April 5th and received 2nd place (out of six teams) in the competition. To prepare for the competition, Neya and CMU conducted system-level testing in a variety of environments and ran dozens of complete practice competitions (Figure 10). The environments included indoor areas with lots of obstacles and background color complexity, parking lots with sparse obstacles with uniform structured background complexity (lane markings, parking spots), and grass fields with patchy grass and dirt areas. In the Table 2, we summarize the test locations (including the competition site) and highlight the system level difficulties experienced within each.

Table 2: System level testing within various environments helped to uncover challenges prior to the competition

	Lighting	Background Color Complexity	Gripper Difficulty	Obstacle Clutter	Locating Operator
Indoor	Controlled lighting	High; colored carpeting, furnature, etc	Low; Flat consistent surface	High; Many obstacles, walls, chairs, etc	Short range, moderate background noise
Parking lot	Inconsistent (clouds/time of day)	Low; lane markings, scattered leaves	Low; Flat consistent surface	Low; Few large obstacles around perimeter	Long range, medium background noise (trees, buildings)
Grass field	Inconsistent (clouds/time of day)	Medium; patchy dirt and grass	High; Tall/thick grass	Medium; Multiple large obstacles around perimeter	Long range, medium background noise (trees, buildings)
Competition	Mostly sunny with patchy clouds	Medium; patchy dirt and grass	Medium; Trimmed/coarse grass	Low; Few large obstacles around perimeter	Long range, high background noise (people, cars, buildings)

There were several key challenges uncovered within each of these testing environments. As expected, the largest challenge we noted after transitioning to an outdoor test site was lighting consistency. Depending on the clouds and time of day, the appearance of objects and background surfaces can actually change dramatically, reducing the intensity contrast and even changing the measured color. In addition, specular effects on shiny objects or wet grass could cause loss of color information entirely. To deal with these issues, we used a fixed white balance on each camera but used an automatically adjusting iris to accommodate for dramatic increases/reduction in lighting associated with cloud cover and time of day. In addition, software color operations were performed in the hue-saturation color space, specifically reasoning about the intensity (value) only to threshold dark or saturated pixels. Lastly, procedurally, we calibrated the system at the beginning of each set of test runs by learning the local background and target object appearance. This served to reduce the effect of low frequency systematic color/intensity consistency issues induced by the location of the sun or indoor vs. outdoor testing.

The second largest challenge during testing was developing gripper hardware and software which was robust to picking up objects in grass. Tall thick grass served to not only limit the visibility of objects (in some cases changing the visible shape cross-section), but would also put significant strain on the gripper motor as it closed around the object given our initial gripper design with flat paddles on each arm. Since closing force (motor current required by the closing motor) was used to determine if the object was in the gripper or not, the grass would often trigger a *grasp success* event within the software, resulting in a false positive classification of whether the object was picked up successfully.

This challenge required several system adaptations to properly allow for robust object pickup. First, we implemented a specific inspection behavior which approached candidate objects and tilted the cameras down to inspect the object from above. This resulted in a clear visible cross-section of the shape outline even when the object was laying in tall grass. In this way, the robot would mark likely candidate objects based on color alone, but reserve the final shape classification once it was close enough to accurately inspect the outline without noise induced by the surrounding grass. In these cases, if the shape did not match the target object even at the closer inspection location, the location would be marked as *inspected* in the object probability map and the search would continue.

We also made several modifications to the gripper hardware and software to handle pickup in thick grass. This included the addition of metal teeth on each of the gripper pads which acted like two rakes, allowing the grass to pass in-between the teeth inducing a smaller resistive force. Even with this modification, the grass would often wrap around the teeth, preventing the pads from closing and requiring sufficient current draw by the motor to trigger a *grasp success* event. To mitigate this, we implemented a tiered pickup action similar to that used by a person raking grass wherein the rake is raised off of the ground to a certain height at which the grass is thin enough to not impede the closing motion. To determine this height, the robot initially drops the gripper pads to the ground and then incrementally raises the gripper height until either it is able to close the gripper all the way or a threshold gripper height is reached – at which the grass is assumed to fall below the gripper and an accurate force-based judgment can be made as to whether the pickup was successful or not. After the gripper has been fully raised, the robot attempts one additional close as a final check to ensure that the object was lifted properly before returning to the operator.

During the final competition at Fort Benning, the largest challenge encountered by our system was determining the location of the operator. Although the target object was properly recovered each time, in several of the challenges when the robot was required to search for the operator, the robot was either unable to find the operator or falsely determined the operator's location. The cause of this problem was likely the background colors associated with spectators, vehicles and buildings around the perimeter of the competition field which did not match the trained background colors of the test field. This induced noise in the estimation of the operator location and in some cases, provided a sufficient match to the operator model in our filters to falsely identify a location for the operator. There are several ways that this problem can be mitigated moving forward. A more formal shape classification similar to that used by our target shape estimation algorithm can be used to identify static gestures (such as raising the arms). Additionally, moving gestures could be recognized by aggregating multiple observation frames to allow the robot to recognize actions (such as waving the arms). Lastly, sanity checks using voice or interactive gestures could be used to verify the location of an anticipated operator location. For example, if the robot suspects an operator location, it could travel closer to that location and wait for an affirmation gesture before continuing to that location.

Figure 10. Testing environments used to prepare for the TARDEC CANINE competition included indoor spaces (upper left), outdoor parking lots (upper middle), outdoor grassy areas (upper right) and Fort Benning (bottom) the days leading up to the competition.

This work represents an initial step towards fielding a robotic mine dog capable of working collaboratively with a soldier in a military context. However, there is a significant effort still required to bridge the gap between technology demonstration and fielded military equipment. Future work may involve the relaxation of constraints imposed by the CANINE competition to include a broader set of objects, more complex search areas or more intuitive robot/operator interaction. Additionally, although not the primary focus of this program, the development of a highly mobile platform with dexterous manipulation capability is necessary to be able to handle and transport a general object of interest.

REFERENCES

[1] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. IEEE Robotics and Automation Magazine, 4(1):23- 33, March 1997

[2] Martin A. Fischler and Robert C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". Comm. of the ACM 24 (6): 381–395. June 1981

[3] P. Batavia and S. Singh, "Obstacle Detection Using Adaptive Color Segmentation and Color Stereo Homography," in Proceedings of the Conference on Robotics and Automation (ICRA), May, 2001.

[4] B. Matei, Y. Shan, H. Sawhney, Y. Tan, R. Kumar, D. Huber, and M. Hebert, "Rapid object indexing using locality sensitive hashing and joint 3D-signature space estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI),* vol. 28, pp. 1111-1126, July 2006.

[5] D. Huber, A. Kapuria, R. Donamukkala, and M. Hebert, "Parts-based 3D object classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* June, 2004, pp. 82-89.