LABRADOR: A Learning Autonomous Behavior-based Robot for Adaptive Detection and Object Retrieval

Brian Yamauchi*[a], Mark Moseley[a], Jonathan Brookshire[b]
[a]iRobot Corporation, 8 Crosby Drive, Bedford, MA 01730; [b]MIT Computer Science and Artificial Intelligence Laboratory, 32 Vassar Street, Cambridge, MA 02139

ABSTRACT

As part of the TARDEC-funded CANINE (Cooperative Autonomous Navigation in a Networked Environment) Program, iRobot developed LABRADOR (Learning Autonomous Behavior-based Robot for Adaptive Detection and Object Retrieval). LABRADOR was based on the rugged, man-portable, iRobot PackBot unmanned ground vehicle (UGV) equipped with an explosives ordnance disposal (EOD) manipulator arm and a custom gripper. For LABRADOR, we developed a vision-based object learning and recognition system that combined a TLD (track-learn-detect) filter based on object shape features with a color-histogram-based object detector. Our vision system was able to learn in real-time to recognize objects presented to the robot. We also implemented a waypoint navigation system based on fused GPS, IMU (inertial measurement unit), and odometry data. We used this navigation capability to implement autonomous behaviors capable of searching a specified area using a variety of robust coverage strategies – including outward spiral, random bounce, random waypoint, and perimeter following behaviors. While the full system was not integrated in time to compete in the CANINE competition event, we developed useful perception, navigation, and behavior capabilities that may be applied to future autonomous robot systems.

Keywords: CANINE, mobile robotics, computer vision, machine learning, navigation, human-robot interaction

1. INTRODUCTION

Robots such as the iRobot PackBot 510 Multi Mission Robot have saved hundreds of lives by allowing warfighters and first responders to inspect suspected improvised explosive devices (IEDs) from a safe distance. These unmanned ground vehicles (UGVs) are controlled remotely via teleoperation using video cameras and joysticks or similar controllers. As a result, multiple people are often required to operate a single robot, with one operator heads-down on the operator control unit (OCU) and one or more soldiers guarding the operator and maintaining situational awareness. Robots would be more intuitive to use and achieve greater force multiplication if operators could interact with robots in a way similar to the way that people interact with trained dogs.

For the LABRADOR (Learning Autonomous Behavior-based Robot for Adaptive Detection and Object Retrieval) Project, iRobot developed technologies inspired by canine intelligence to make human-robot interaction (HRI) more intuitive. LABRADOR was part of the CANINE (Cooperative Autonomous Navigation in a Networked Environment) Program funded by the US Army Tank-automotive and Armaments Research, Development, and Engineering Center (TARDEC). Technologies developed for LABRADOR include real-time, vision-based object learning, detection, and tracking and autonomous area coverage using behavior-based techniques. These technologies were implemented on an iRobot PackBot UGV platform equipped with a manipulator arm, custom gripper, computational payload, cameras, and LIDAR, GPS, and IMU sensors.

2. LABRADOR ROBOT PLATFORM

We used an iRobot PackBot as the platform for the LABRADOR robot. The PackBot is a rugged, man-portable UGV that can climb over rough terrain using its tracks and flippers. Over 4500 PackBots have been deployed to the battlefields of Afghanistan and Iraq and around the world, where they have saved hundreds of lives by helping warfighters to inspect and disarm suspected IEDs.

The PackBot used for the LABRADOR Project (Figure 1) is equipped with a manipulator arm and three color video cameras – one on the robot's head (the attack camera), one on the robot's arm (the gripper camera), and one on the

*yamauchi@irobot.com, www.irobot.com

robot's base (the drive camera). The manipulator arm includes a shoulder joint that rotates and pivots, two pivoting elbow joints, a gripper that rotates and closes, and a camera head that pans and tilts. The arm has a 1.87 m reach at full extension and can lift 4.54 kg at full extension and 13.61 kg close-in.

We modified the PackBot hardware by adding a custom gripper, computational payload, LIDAR, GPS, and IMU sensors, a hardware emergency-stop (E-stop) payload, and a warning light. Figure 1 (left) shows the fully-integrated LABRADOR UGV platform.

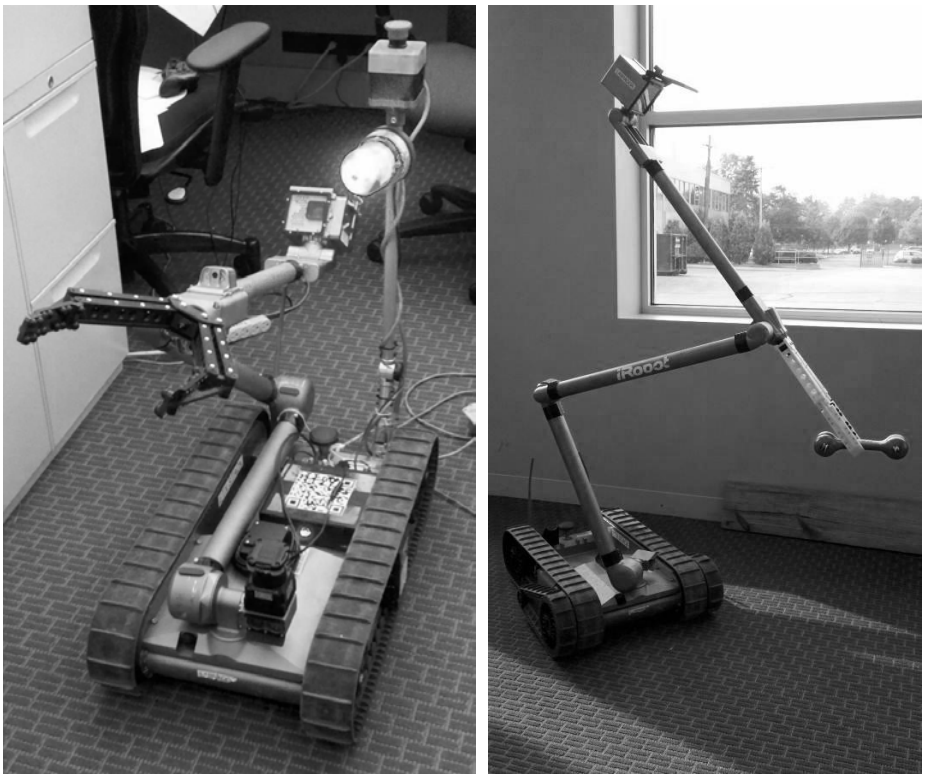

Figure 1. Fully-integrated LABRADOR robot platform (left); LABRADOR robot picking up dog toy (right)

We acquired a Payload Computer for the LABRADOR robot to run the autonomous behaviors and the perception, learning, and navigation software. The PackBot chassis CPU runs only the low-level robot control software. The Payload Computer is a Stealth PC LPC-670 with a dual-core Intel Core i7-620M processor running at 2.67GHz with four hyperthreads, 8 GB of DDR3 RAM, a 240 GB solid state drive, and an 802.11g wireless network interface. The LPC-670 fits easily within the PackBot payload bay.

We integrated a uBlox GPS receiver and a Microstrain six-axis MEMS IMU with the PackBot. Localization software running on the Payload Computer fuses sensor data from the GPS, IMU, and odometry to maintain an accurate estimate of the robot's position and orientation. We also added a Hokuyo UTM LIDAR to provide range data for obstacle avoidance.

We designed a custom gripper to allow the LABRADOR gripper to pick up objects ranging in size from a baseball to a basketball. Figure 1 (right) shows the robot picking up a rubber dog toy shaped like a bone using an early prototype of this custom gripper built using a 3D printer. The final version of the gripper was constructed from anodized aluminum and is shown mounted on the robot in Figure 1 (left).

We designed, built, and tested a Hardware E-Stop Payload for the PackBot with both wired and wireless controllers. The Hardware E-Stop Payload fits in the PackBot payload bay next to the Payload Computer. The hardware E-Stop mechanism shorts the system voltage to ground and causes the built-in circuit breaker on the robot to safely and immediately shut down all robot hardware. The wired E-Stop controller is mounted on a post on the robot where it is easily reachable.

The wireless E-Stop controller transmits a 900MHz (100mw) signal to the payload whenever the button is up (E-Stop disengaged). When the button is pressed or the E-Stop payload ceases to receive the signal, the E-Stop payload will immediately shut down the robot. We have tested this E-Stop payload to insure that the transmitter range is sufficient for the competition. In our tests, the wireless E-Stop worked at ranges up to 300 m. We have also verified that the E-Stop automatically engages if the payload loses the signal.

3. REAL-TIME OBJECT LEARNING, DETECTION, AND TRACKING

3.1. Overview

A key achievement of the LABRADOR Project was the development of a real-time computer vision system that learns to detect and track objects that are presented to the robot. We developed a hybrid vision system that combines a shape-based track-learn-detect (TLD) tracker[3] with a color-histogram-based object tracker. The color-based tracker learns to recognize the object from a set of training images. Once the color tracker finds the object, the color tracker and the TLD tracker are used together to track the object as the robot moves. Both the color tracker and the TLD tracker adapt to changes in the appearance of the object caused by changes in lighting and viewpoint.

3.2. Shape-Based Tracking, Learning, and Detection (TLD)

Many of the most successful object detection algorithms (e.g., face detection[1] and pedestrian detection[2]) require an offline learning process based on a large database of training images. For example, a face detector may need to be trained on thousands of images before it achieves satisfactory performance. For CANINE, we needed a vision system that could learn to recognize novel objects in real-time based on a limited set of training images. As one component of our system, we developed a shape-based object learning system based on the track-learn-detect[4] (TLD) algorithm. TLD-based vision systems are able to learn to track and detect objects based on a very small number of training images.

TLD is a semi-supervised learning algorithm that generalizes from a few examples to a larger data set and learns a visual model. A typical challenge with semi-supervised techniques is preventing divergence. While attempting to refine the model, the algorithm will inevitably make mistakes. If left unchecked, these misclassifications can cause the model to degrade and generate even more mistakes.

TLD techniques address this issue by providing feedback to the learning procedure by combining the output of a "tracker" with the output from a "detector." A tracker attempts to follow an object from one frame to the next. A detector attempts to find all objects within a single frame. The tracker and the detector provide two independent estimations of the location of the object. The algorithm can then apply the following logic:

1. When the tracker and detector agree, the algorithm can reasonably conclude it knows the true location of the object.
2. Trackers typically suffer when the object (or camera) moves suddenly or the object leaves the field of view. When the track is lost but a detection exists, the detector can reinitialize the tracker. No learning occurs here because the tracker and detector cannot validate each other.
3. Detectors typically suffer when there is clutter or little training data. When the tracker's confidence is high, the tracker can provide additional training data to the detector and improve its performance. When there is clutter in the scene, the detector may generate many possible locations. The location of the track, even with low confidence, can then be used to select the most likely location.
4. When both the tracker and detector have low confidence, it can be reasonably determined that the target object is not in the scene.

The shape-based component of the LABRADOR vision system is based on an implementation of the TLD algorithm developed by Kalal[3,4]. We use a Lucas-Kanade[5] optical flow tracker to track the location of the object from an initial bounding box. The detector uses a random forest of two-bit binary patches similar to Haar features[6], but with two-bits per rectangle.

The TLD algorithm provides a good starting point, but it has several limitations. First, it does not use color as a feature, so it cannot distinguish objects with the same shape, but different colors. Second, the algorithm does not perform well

with objects that have uniform coloring. Although the edges of the object can provide some features, these features are less reliable than features internal to the object.

3.3. Color Histogram Tracker

To address the limitation of the TLD tracker, we added a color detector to the LABRADOR vision system. The color detector's task is to take a single training image and model the color of the object. The detector must then identify image patches with similar color and update the model when directed by the TLD framework. Our color model is a radial basis function (RBF) support vector machine (SVM) with features from a 3D histogram in YCbCr color-space. We used YCbCr instead of RGB due to the successful results of previous research in tracking using the YCbCr color space[7]. The initial training image is used to identify a region in YCbCr color-space (Figure 2). When learning occurs, the 3D histogram is updated and the SVM is retrained. Thus, the color model can adapt as the lighting changes.

The output of the color detector is merged with the output of the TLD detector and the results are clustered. We found that the color detector had a fairly low false negative rate, so we use the color detections to filter the TLD tracker. When the color detector determines that the track window is not of the right color with high confidence, the track is discarded. On the other hand, the color detector has a higher false positive rate. Thus, when the track has been lost and needs to be reinitialized by the detectors, we rely primarily on the TLD detector. The color detector is used for re-initialization only after the TLD detector has failed for more than two frames.

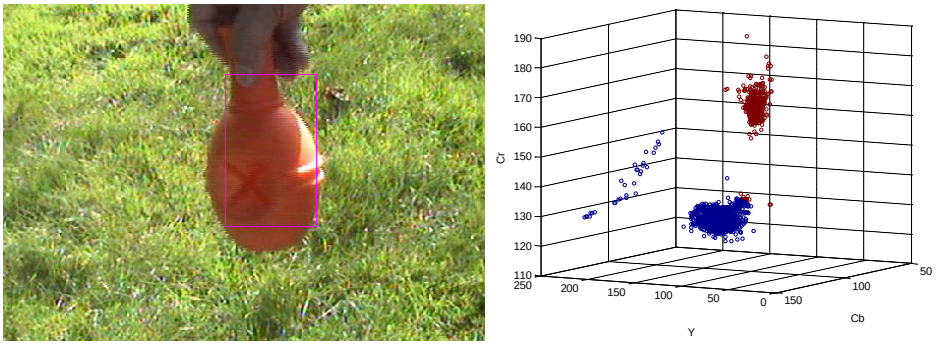

Figure 2. The initial training image (left) is used to construct the 3D histogram (right). Notice that the YCbCr values of the orange object in the 3D histogram (red) are well separated from the grass (blue). An RBF SVM is trained to distinguish the orange of the bottle from the green of the grass.

3.4. Experimental Results

The following images show several example outputs from the TLD and color algorithm. Figure 3 through Figure 5 show examples of successes; Figure 6 and Figure 7 show two failure modes. On the left of each figure, a magenta box shows the final bounding box on original color image. On the right of each figure is the color score (red means more like the target object, blue means less like the target object). The yellow box on the right shows the output of the TLD detector; the black boxes shown the output of the color detector.

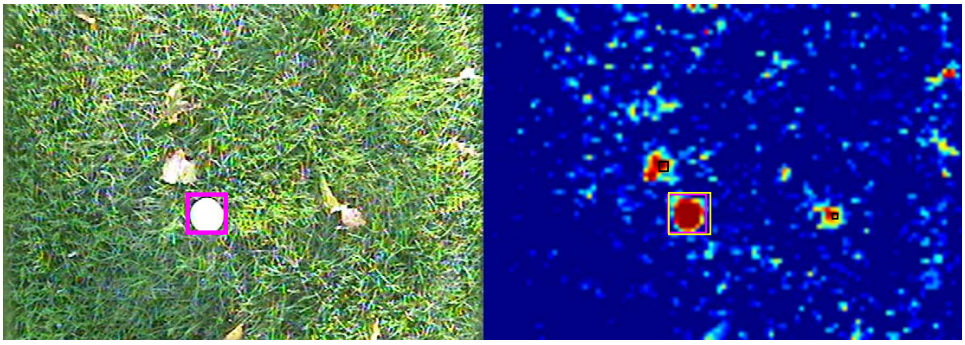

Figure 3. Color detector has not detected the baseball, but has detected the reflections off two leaves. The system relies on the TLD detector because its confidence is higher.

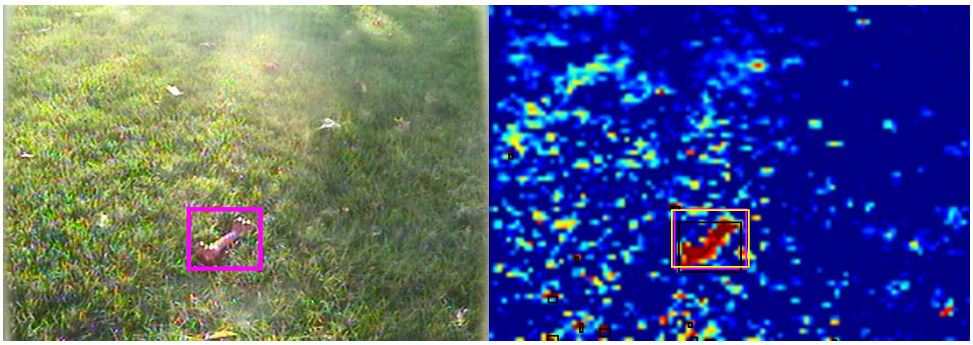

Figure 4. Several false positives from the color detector are visible in lower left, but the most confident color detection and TLD detector correctly detect the bone.

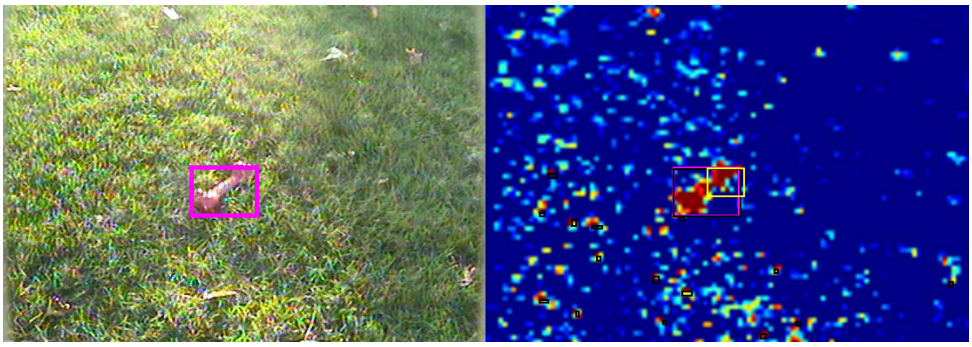

Figure 5. TLD detector incorrectly locates the yellow bounding box on the upper right corner of the red bone. However, color detector has detected the bone well and is used to correctly produce a better final bounding box.

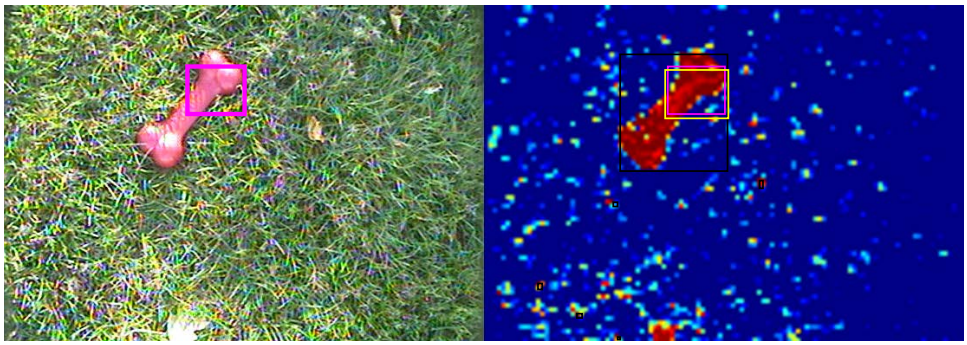

Figure 6. Color detector has correctly identified the bone outline, but TLD detector is overconfident and track is kept on only a portion of the bone.

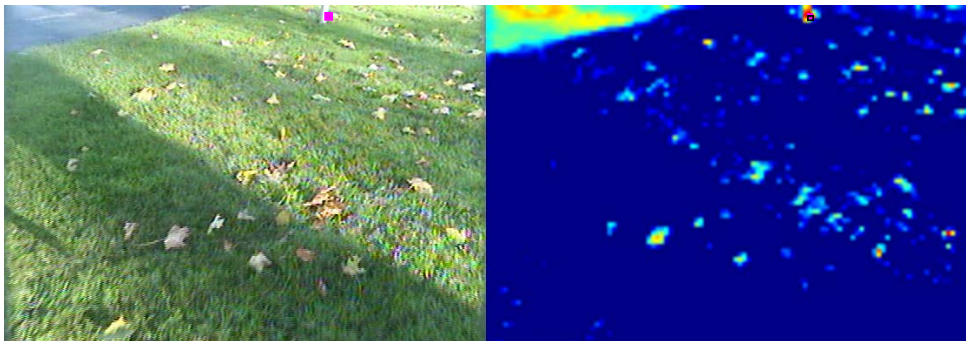

Figure 7. Color detector, trained on the white baseball, has detected a distant tree in bright sunlight. Since no other detections exist, the track has been incorrectly assigned to the tree.

The LABRADOR vision system was able to successfully learn to detect and track objects presented to the robot in less than a minute. The robot automatically positioned the manipulator arm at a number of preset poses to gain multiple viewpoints on an object placed in front of the robot. The vision system was able to track objects at a 2 Hz update rate. We could reliably detect test objects, such as the red rubber bone designated as the first CANINE test object, at ranges up to 4 m in outdoor environments, on both asphalt and grass.

4. AUTONOMOUS NAVIGATION

4.1. Waypoint Navigation

In addition to the vision system, we also developed an autonomous navigation system for the LABRADOR robot. This system provided the robot with a robust waypoint navigation capability, which fused odometry, GPS, and IMU to maintain an accurate estimate of the robot's current position.

To test the waypoint navigation behavior, we generated random waypoint paths and measured how quickly and accurately the robot can follow the path. We conducted experiments outdoors in an asphalt courtyard adjacent to the iRobot Headquarters building.

Random paths were generated by defining a bounding box for the test area and generating random waypoints within that bounding box. The order in which the waypoints must be traversed is random.

We used the following quantitative performance metrics to evaluate system performance:

1. Time
2. Average position error (accuracy)
3. Average position standard deviation (repeatability)

Time was the total time required to visit all of the waypoints on the path and arrive within the final waypoint capture radius.

Accuracy was measured by computing the average position error for each path. Position error was determined by identifying the closest approach of the robot to each waypoint, and taking the average of those values.

$$E_a = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\sqrt{\left(waypoint\_x_i - closest\_x_{i,j}\right)^2 + \left(waypoint\_y_i - closest\_y_{i,j}\right)^2} \quad (1)$$

Where n is the number of waypoints, m is the number of trials, $(waypoint\_x_i, waypoint\_y_i)$ is the ith waypoint on the path, and $\left(closest\_x_{i,j}, closest\_y_{i,j}\right)$ is the robot's point of closest approach to waypoint i in trial j.

Repeatability was measured by computing the average standard deviation of the points of closest approach over all trials for a given path. First, we compute the mean position of closest approach to each waypoint:

$$\overline{closest\_x_i} = \frac{1}{m}\sum_{j=1}^{m} closest\_x_{i,j} \quad (2)$$

$$\overline{closest\_y_i} = \frac{1}{m}\sum_{j=1}^{m} closest\_y_{i,j} \quad (3)$$

Then we compute the average standard deviation for each axis:

$$\sigma_x = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\frac{1}{m}\sum_{j=1}^{m}\left(\overline{closest\_x_i} - closest\_x_{i,j}\right)^2} \quad (4)$$

$$\sigma_y = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\frac{1}{m}\sum_{j=1}^{m}\left(\overline{closest\_y_i} - closest\_y_{i,j}\right)^2} \quad (5)$$

The repeatability metric was then defined as the Euclidean distance corresponding to the average standard deviation:

$$E_r = \sqrt{\sigma_x^2 + \sigma_y^2} \quad (6)$$

Figure 8. iRobot Courtyard Test Area (left) and random waypoint navigation trial (right) (red line: waypoint path, green line: simulation path, blue line: actual robot path).

We tested the waypoint navigation behavior in the iRobot Courtyard Test Area (Figure 8, left) adjacent to iRobot HQ. We used a Monte Carlo technique to generate random waypoint paths, and ran 10 trials per path. We then evaluated the performance of the UGV in terms of time to complete the mission, average position error (accuracy), and average position standard deviation (repeatability).

Figure 8 (right) shows the results from a single waypoint navigation trial. Six primary waypoints (red) were randomly generated within the iRobot Courtyard Test Area, and secondary waypoints were added between each pair of waypoints. The green path is the path taken by the robot in simulation. The blue path shows the actual path taken by the robot.

Table 1. Results from waypoint navigation experiments

Path	Capture Radius (m)	Number of Trials	Mean Time (s)	Time Std Dev (s)	Mean Position Error (m)	Position Error Std Dev (m)	Repeatability (x, y)
1	0.25	10	79.5	9.8	0.40	0.88	(0.03, 0.03)
1	0.50	10	89.9	45.4	0.41	0.89	(0.15, 0.20)
2	0.25	10	75.0	9.0	0.30	0.38	(0.09, 0.02)
2	0.50	10	70.2	5.5	0.37	0.35	(0.06, 0.02)
Average			78.7		0.37		(0.08, 0.07)

We ran a total of 40 trials and measured the effects of different capture radii (0.25 m vs. 0.5 m). The results of these experiments are summarized in Table 1. In these experiments, the LABRADOR robot was able to navigate to these waypoints with an average position accuracy of 0.37 m and an average repeatability of 0.11 m.

5. SEARCH AND COVERAGE BEHAVIORS

To allow the LABRADOR robot to search an outdoor area for a specified object, we developed a number of search and coverage behaviors that make use of the robot's waypoint navigation capability. These behaviors operate within a specified bounding box that represents the entire area being searched. The robot's localization system determines its current location within the area using fused odometry, GPS, and IMU data.

We implemented the following search and coverage behaviors:

1. **Spiral Outward:** Spiral outward from the center of the test area until the boundary is reached.
2. **Random Bounce:** Select random points on the perimeter of the test area and navigate from one point to the next
3. **Random Waypoints:** Navigate to random points within the perimeter of test area.
4. **Perimeter Following:** Follow the perimeter of the test area.

We tested these behaviors outdoors in the iRobot Courtyard Test Area. As the robot executes a search behavior, it constantly scans the ground ahead of the robot using the attack camera mounted on the robot arm. The color-based object detector continuously searches for the target object. When the object is detected (typically at a range of around 4 m), the investigate behavior is invoked. This behavior causes the robot to drive up to the object until it approaches close enough to pick up the object.

6. CONCLUSIONS

We completed the implementation of the object learning, detection, and tracking system; the waypoint navigation system; and the search and coverage behaviors for LABRADOR. Due to budget limitations, we were unable to implement the visually-guided manipulation behaviors that would have been required to retrieve objects and deliver them to a specified location. As a result, we were unable to participate in the CANINE competition event. However, the computer vision and autonomous navigation technologies developed for LABRADOR will be useful in future development of autonomous robots that can perform search and retrieval tasks.

7. NOTES

OPSEC #: 23498 DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited. Disclaimer: Reference herein to any specific commercial company product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the Department of the Army (DoA). The opinions of the authors expressed herein do not necessarily state or reflect those of the United States Government or the DoA, and shall not be used for advertising or product endorsement purposes.

REFERENCES

[1] Viola, J. and Jones, M. “Robust real-time object detection,” Second International Workshop on Statistical and Computational Theories of Vision – Modeling Learning, Computing, and Sampling, 905-910 (2001).

[2] Felzenszwalb, P., Girshick, R., and McAllester, D. “Cascade object detection with deformable part models,” Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2010).

[3] Kalal, Zdenek et al. “Online learning of robust object detectors during unstable tracking,” Proc. of the IEEE Third On-line Learning for Computer Vision Workshop, Kyoto, Japan, IEEE Computer Society, (2009).

[4] Kalal, Zdenek et al. “P-N learning: Bootstrapping binary classifiers by structural constraints,” Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2010).

[5] Lucas, B. and Kanade, T. “An iterative image registration technique with an application to stereo vision,” Proc. Of the Image Understanding Workshop, 121-130 (1981).

[6] Viola, J. and Jones, M. “Rapid object detection using a boosted cascade of simple features,” Proc. of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), (2001).

[7] Wu, K. “Face detection based on YCbCr Gaussian model and KL transform,” Proc. of the 2011 International Symposium on Computer Science and Society (ISCCS), 100-103 (2011).