Using a Depth Camera for Object Classification in Nuclear Gloveboxes

Adam Allevato*, Thomas Lu*, Mitchell Pryor*

**The University of Texas at Austin, 10100 Burnet Road MER Building Stop R8600, Austin, TX 78758*

INTRODUCTION

Ever since the dangers of radiation have been understood, humans have sought to reduce their exposure to nuclear materials. To keep levels As Low As Reasonably Achievable (ALARA), a balance must be struck between avoidable and necessary exposures. This is a difficult problem when various industries require the handling of nuclear materials on a regular basis. Some of these tasks, such as removing and inserting fuel rods in a reactor, are well-defined and repeatable, and thus easily automated. However, other procedures, including experiments and spill cleanup, require additional dexterity and versatility that until recently have been unattainable by machines. With the advent of new sensor technologies and processing techniques, robots have gained the ability to perform a broader range of nuclear material handling tasks.

A major problem still facing automated systems is the detection and classification of objects located in arbitrary locations in a robot's work area. These objects may be tools or materials that a robot can manipulate once detected. Computers cannot yet fully emulate the physics and neuroscience behind human vision, but in recent years they have made significant improvements in the realm of visual perception. Many of these advancements have been collectively captured in the software library Open Source Computer Vision (OpenCV). The first complete version of OpenCV, originally developed at Intel, was released in 2006 [1]. OpenCV contains a variety of established 2D image processing techniques, such as edge detection [2] and pattern matching [3]. This makes it suitable for detecting visual tags such as QR-style "barcodes," but it is not designed for complex processing of 3D sensor data. In 2010, the Point Cloud Library (PCL) was created by Willow Garage [4] and is currently maintained by the Open Perception Foundation [5]. PCL is designed for processing data from 3D cameras such as the Microsoft Kinect, which use infrared light to estimate distances. The library includes machine learning-based classifiers for resolving point clouds into known objects and for estimating their position and orientation in 3D space. Some of these classifiers achieve accuracy upwards of 90%, but tasks involving nuclear material demand extreme accuracy and often are performed in challenging environments (i.e. poorly lit, confined, harsh environment for sensors, etc.).

At Los Alamos National Laboratory, nuclear material is handled in lead-lined gloveboxes with reflective polished surfaces. In addition, most of the objects of interest, including tools and the nuclear materials, are metal. The high specular reflectivity of these objects make it difficult to either obtain unique patterns in individual objects for 2D recognition or generate complete point clouds using infrared scanning for 3D point cloud recognition.

To address this issue, a new 3D classifier algorithm was developed to provide faster, more robust object classification and pose estimation than other classifiers when used in a glovebox setting such as that described above [6]. Initial results were encouraging for a small set of objects using a first generation 3D camera. In this paper, results are presented for detecting objects using a second generation 3D camera on a larger dataset using the Circular Projection Histogram (CPH) classifier. Additionally, the code and demonstration environment have been updated for ease-of-use for both novice developers and end-users.

APPROACH

To evaluate the performance of the CPH classifier in a glovebox, point cloud data from 13 different objects (a 30% increase over previous sample sizes [3]) was collected and CPH was applied to these clouds. Objects selected are commonly found in a lab or glovebox environment. The list of objects tested is shown in Table I.

Most objects in the training set are metal with reflectivity similar to the glovebox surfaces. CPH is a scale-variant classifier, meaning it can discern between objects of identical shape but different size. This is in contrast to some of the classifiers provided with PCL, such as the Viewpoint Feature Histogram classifier [7]. To test this capability, some of the objects have the same shape, but different size (e.g., *large_can_lidded* and *small_can_lidded*). The experimental testing setup is shown in Figures I and VIII. The experiment was designed to be as much like actual glovebox working conditions as possible. The 3D camera, an ASUS Xtion Pro, improves on the capabilities of the first-generation Microsoft Kinect, providing highly detailed point clouds.

Table I. Training Dataset

Label	**Description**
large_can_lidded	Brushed metal can with lid, 8in. dia.
small_can_lidded	Brushed metal can with lid, 4in. dia.
medium_can	Brushed metal can, no lid, 6in. dia.
small_bowl	8in. dia. metal bowl
large_bowl	10in. dia. metal bowl
tslot	Small piece of aluminum extrusion
ball	Foam ball, 4in. dia.
lathe_tool	Lathe tool mount
tape_roll	Roll of yellow duct tape
ring	Steel coupling ring, 6in. dia.
hub	Steel hub with axle, 4in. dia.
bottle	Plastic colored water bottle
broom	Plastic long-handled brush

The software pipeline is shown in Fig. 4. Early in the research process, it was discovered that the recognition results are dependent on the parameters used to filter the point clouds. With incorrect filtering parameters, point clouds would incorrectly cluster together, or entire clouds would be missed. A control panel was built using ROS's dynamic_reconfigure package [8] to allow for easy adjustment of filtering parameters on-the-fly. Table II shows the parameters used in the experiments.

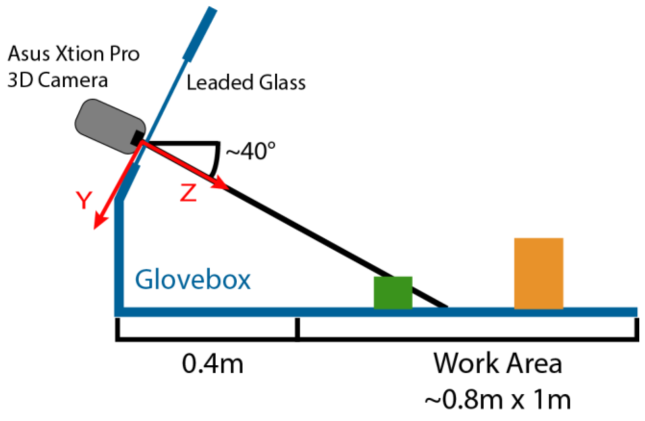


Fig. 1. Experiment Schematic

The dense point clouds from the Xtion Pro were first processed using a voxel grid filter [9], downsampling the raw point cloud data from the 3D camera using methods provided by PCL. Points outside the work area were removed by clipping the point cloud to fit within a bounding box. Next, the largest planar set of points in the scene was removed by using the random sample consensus (RANSAC) plane-finding method [10]. Planar features were removed until the number of points remaining in the cloud is less than the "Analysis percentage" parameter (see Table II). Finally, the remaining points are clustered into groups, which correspond to the different objects in the scene. Heavily filtering and processing the point clouds helped make the recognition step as fast as possible by reducing the amount of data throughput.

Table II. Point Cloud Processing Parameters

Parameter	**Value**
Bounding box x-max*	0.4m
Bounding box x-min*	-0.4m
Bounding box y-max*	0.35m
Bounding box y-min*	-0.6m
Bounding box z-max*	1.4m
Bounding box z-min*	0.2m
Maximum plane-finding iterations	50
Segmentation distance threshold	1cm
Analysis percentage	30%
Voxel grid filter leaf size	5mm
Minimum cluster size	300 points

*The origin for the bounding box's coordinate frame is attached to the camera itself and follows the right-hand rule. See Fig. 1.

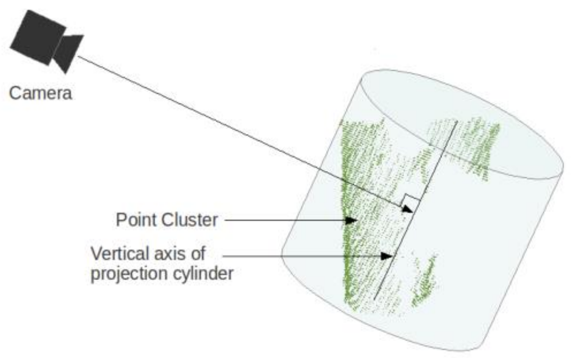


Fig. 2. The CPH point cloud classifier [6].

The processed clouds were then processed using the CPH method, which takes a point cloud and converts it to a histogram (Fig. 2 and Fig. 3). We treat the histogram as a 360-dimensional feature vector. In the training stage, this histogram was saved to a file. In the testing stage, the pipeline compared the feature vector with those saved during the testing stage to determine the classification result. To improve accuracy in the testing stage, the classification is repeated multiple times a second, and the results are updated using a naive Bayesian classifier.

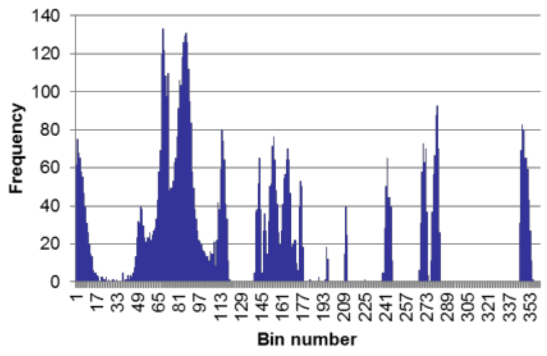


Fig. 3. An example CPH result [6].

Because the training step treats each angle view of the object as a separate data point, classification automatically provides an estimate of the object's pose (rotation about the object's vertical axis). In each step, the detected position is taken as the centroid of the 3D point cloud. Both the pose and position estimates are prone to errors from incorrect classifications and noisy data, so to reduce the effects of noise on the overall results, we filter the pose and position using a Kalman filter with the prediction step omitted [11] (this is the same approach taken in previous research efforts [6]).

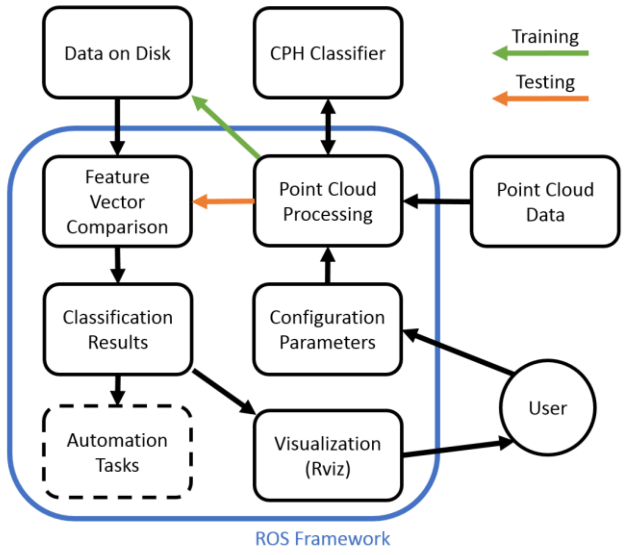


Fig. 4. Recognition Software Pipeline

Robotic Operating System (ROS) was used to control both the training and testing stages. ROS is a modular code framework that contains many computer vision-related packages, allowing for easy extension and modularity [12] (for example, using a different 3D camera or pan table). In the training stage, ROS interfaced with the Arduino-controlled pan table which rotated the object 360°. Training objects at different angles allows the system to identify the object's rotation about the vertical axis during the testing stage, in addition to classifying the object type.

In the testing stage, the pan table is removed and objects are placed in the work area. A feature vector is created for each point cluster in the scene and passed along the pipeline for recognition based on training data collected for the 13 objects listed above. The pipeline uses the FLANN k-Nearest Neighbor (k-NN) algorithm [13] to match testing feature vectors to training data and return object classifications.

Since the vision pipeline treats each cluster in the work area individually, it is capable of recognizing an object from a collection as well as recognizing individual objects, provided they are far enough apart to be segmented as distinct clusters. In addition, the system classifies objects iteratively, making continuous passes over each cluster in the work area and updating its determination using a predictive statistical model. Until it has reasonable confidence to return one label consistently for each object, the system may return conflicting object labels on each pass of the classification algorithm. Thus, a useful metric of the CPH method's effectiveness is the accuracy of the reported labels in the first N passes.

RESULTS

It is demonstrated that the vision pipeline using the CPH method is suitable for near real-time applications. The system was tested with objects in the dataset placed individually at various locations in the glovebox. The accuracy of object classifications within the first 40 passes was recorded at each location, and the predictive statistical model for each object was reset each time the object was repositioned. The average system accuracy for a few objects is shown in Fig. 5.

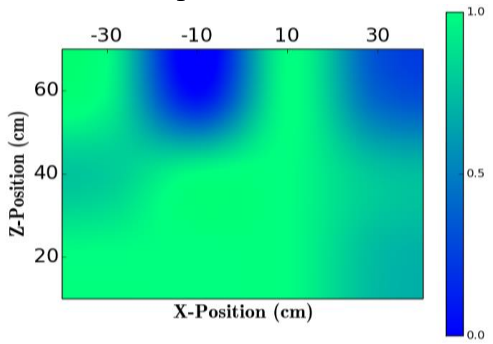
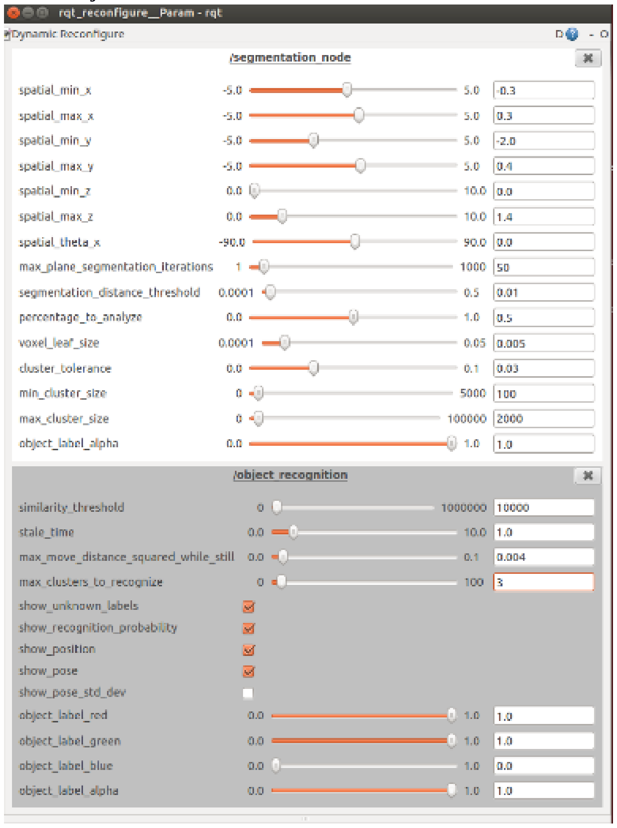


Fig. 5. Average Classification Accuracy in 40 Passes by Location

The system accuracy shown in Fig. 5 is the worst-case scenario, and naïve Bayesian classification improves the overall accuracy significantly. Each pass was completed in approximately 210 ms. Given that the Bayesian classifier reliably classifies objects in a small number of passes of the classification algorithm, the system operator can be reliably shown object recognition results with negligible latency.

Fig. 6 shows the user interface. The interface is designed to provide the operator with maximum control over the vision pipeline, while providing important feedback from the classifier. The full point cloud is overlaid on the color image from the depth camera, with the point clouds for detected objects highlighted on a black background. The yellow labels provide the classification result with a certainty value as determined by the Bayes classifier, and also display the estimated position and pose of the object.



```
rqt_reconfigure__Param - rqt
Dynamic Reconfigure

/segmentation_node
spatial_min_x                          -5.0     5.0      -0.3
spatial_max_x                          -5.0     5.0      0.3
spatial_min_y                          -5.0     5.0      -2.0
spatial_max_y                          -5.0     5.0      0.4
spatial_min_z                           0.0     10.0     0.0
spatial_max_z                           0.0     10.0     1.4
spatial_theta_x                        -90.0    90.0     0.0
max_plane_segmentation_iterations       1       1000     50
segmentation_distance_threshold         0.0001  0.5      0.01
percentage_to_analyze                   0.0     1.0      0.5
voxel_leaf_size                         0.0001  0.05     0.005
cluster_tolerance                       0.0     0.1      0.03
min_cluster_size                        0       5000     100
max_cluster_size                        0       100000   2000
object_label_alpha                      0.0     1.0      1.0

/object_recognition
similarity_threshold                    0       1000000  10000
stale_time                              0.0     10.0     1.0
max_move_distance_squared_while_still   0.0     0.1      0.004
max_clusters_to_recognize               0       100      3
show_unknown_labels                     ✓
show_recognition_probability            ✓
show_position                           ✓
show_pose                               ✓
show_pose_std_dev
object_label_red                        0.0     1.0      1.0
object_label_green                      0.0     1.0      1.0
object_label_blue                       0.0     1.0      0.0
object_label_alpha                      0.0     1.0      1.0
```

Fig. 6. Recognition output and configuration panel.

The vision pipeline used does have minor drawbacks. The location-based accuracy of the system reveals that the angle at which the system sees an object in the work area affects its ability to classify the object. When the angle between the camera's viewpoint and the object's vertical axis differs significantly from the angle at which the system was trained, the feature vector generated for the testing object will appear skewed with respect to the training data and result in a greater probability of misidentification. In general, this effect becomes more pronounced as objects in the work area are brought closer to the camera. This results in increased accuracy for objects towards the side of the glovebox far from the camera. This deficiency could be addressed by lowering the camera to minimize the change in the viewing angle or using distance data to ascertain which of multiple training sets are appropriate, or – most simply – using cameras located at multiple windows and using results from those with the most appropriate viewing angle.

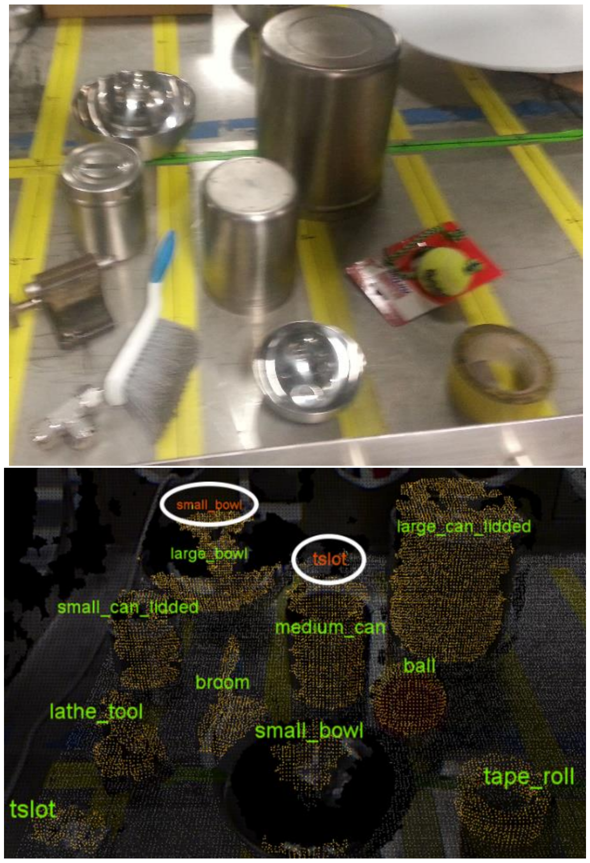


Fig. 7. Cluttered scene with recognition results and a reproduction of the scene in the glovebox

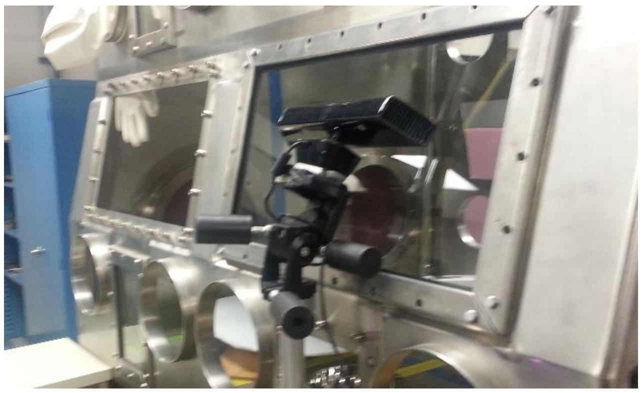


Fig. 8. Glovebox with camera outside glovebox window

Fig. 7 shows a cluttered scene in a glovebox containing 10 objects. The detected point clouds are shown in orange with the classification labels overlayed. The recognition pipeline provides a correct classification for all objects (green labels in the figure). However, in some cases (circled red labels), a single object produces more than one classification result. This is due to objects' material properties and shapes, which result in incomplete point clouds. The incomplete clouds are then split into two different clusters and classified separately.

CONCLUSIONS AND FUTURE WORK

Detecting objects quickly and reliably in a laboratory environment is an important step towards automating complex tasks or – at a minimum – automating additional verification capabilities that are required when handling nuclear materials. The pipeline that was built around the CPH classifier is a fast technique that quickly categorizes objects and determines their pose (i.e. location and orientation), even in specularly reflective glovebox environments that contain reflective objects. Tests have shown that the classifier pipeline can discern between numerous objects within a reasonable work area.

The classifier and vision pipeline have been integrated into a pick-and-place system, where a robotic manipulator discerns between objects using CPH. The robot can then pick up the desired object and move it to a target position (Fig. 9).

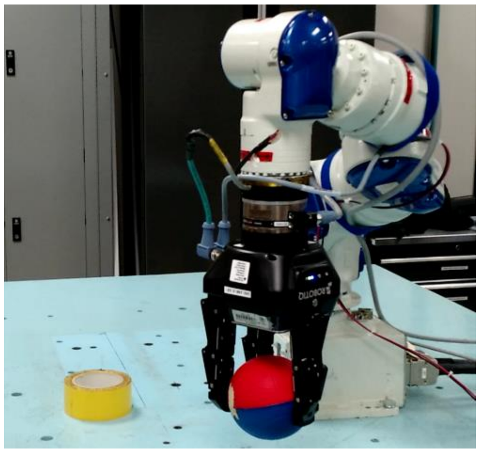


Fig. 9. Vision-controlled pick-and-place task

ACKNOWLEDGEMENTS

This material is based upon work supported by Los Alamos National Laboratory and the University of Texas at Austin.

REFERENCES

1. "ABOUT | OpenCV," Itseez; opencv.org/about.html (current as of Feb. 26, 2015).
2. "Canny Edge Detection," OpenCV; docs.opencv.org/trunk/doc/py_tutorials/py_imgproc/py_canny/py_canny.html (current as of Feb. 27, 2015).
3. "Template Matching," OpenCV; docs.opencv.org/doc/tutorials/imgproc/histograms/template_matching/template_matching.html (current as of Feb. 27, 2015).
4. R. RUSU and STEVE COUSINS, "3D is here: Point Cloud Library (PCL)," *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13, 2011.
5. "Open Perception," Open Perception, Inc.; openperception.org (current as of Feb. 28, 2015).
6. B. O'NEIL, "Object Recognition and Pose Estimation for Manipulation in Nuclear Materials Handling Applications," Dissertation, University of Texas at Austin, Mechanical Engineering (May 2013).
7. R. RUSU, G. BRADSKI, R. THIBAUX, J. HSU, "Fast 3D Recognition and Pose Using the Viewpoint Feature Histogram," *Proc. of the 23rd IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS),* Taipei, Taiwan, October, 2010.
8. "dynamic_reconfigure," ROS.org, wiki.ros.org/dynamic_reconfigure (current as of Feb. 28, 2015).
9. "Downsampling a PointCloud using a VoxelGrid filter," Point Cloud Library, pointclouds.org/documentation/tutorials/voxel_grid.php (current as of Feb. 28, 2015).
10. M. FISCHLER and R. BOLLES, "Random sample consensus: a paradigm for model fitting with Applications to Image Analysis and Automated Cartography," *Comm. of the ACM*, 24, 6, 381 (1981).
11. R. KALMAN, "A New Approach to Linear Filtering and Prediction Problems," *Trans. of the ASME – Journal of Basic Engineering*, 82, Series D, 35 (1960).
12. "About ROS," Open Source Robotics Foundation; ros.org/about-ros/ (current as of Feb. 26, 2015).
13. M. MUJA and D. LOWE, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration," *Proc. of Int. Conf. on Computer Vision Theory and Application (VISSAPP '09)*, Lisboa, Portugal, Feb. 5-8, 2009, pp. 331-340, INSTICC Press (2009).