# Robotic Doorknob Disinfector

Matthew Scott Rosoff

Department of Electrical and Computer Engineering

Cornell University

Ithaca, NY 14850

msr53@cornell.edu

## Abstract

**The robotic door-handle disinfector is a robotic platform that travels heavily trafficked hallways and disinfects door-handles to improve public health. Machine learning algorithms were created to help the Erratic ™ robotic platform navigate hallways and identify door-handles using only a commercial webcam and a laser scanner. A parts-based and a texture-based machine learning object detector were tested in a real-world environment. With the addition of a wall-aligning motion control algorithm, the parts-based object detector consistently detected >85% of the doorknobs in the environment and false-alarmed in <1% of the image frames. The robot does not require a map of the environment nor any modification to the environment. At the conclusion of the research project, the robot can autonomously navigate hallways and disinfect with UV light to help hospitals and other heavily infectious areas achieve lower infection rates.**

## 1 Introduction

Each year, 1.7 million Americans are infected just by visiting hospitals. Of those 1.7 million people, 99,000 die each year from these nosocomial infections [1]. Recently, my 86 year old grandmother was admitted to the hospital for a car accident she was in, only to be released and told it was safer for her to recuperate at home. It is estimated that nosocomial infections add $4.5 billion to $11 billion each year to health care costs in the US alone and one of the root causes is negligence from hospital workers. According to Stephen Abedon of Ohio State University, infection in hospital is partially a result of disregard for established medical protocols and transmission occurs predominately through surface to body transmission [2].

The robot described in this paper is an attempt to mitigate the threat of nosocomial infections. There are many avenues leading to infection, but research has shown that a prominent cause is surface contact [3]. In one day, a single set of hands touches a myriad of surfaces. It is much more effective to disinfect hands than disinfect myriads of surfaces. However, there is no great technology that can force people to disinfect their hands. Instead, performing automated surface disinfection on high-risk areas like doorknobs can prove beneficial to hospital care.

To solve the hospital disinfection problem, several design requirements had to be imposed. There are a myriad of design requirements for a commercial robotic doorknob disinfector – and even more if the robot is to operate in a hospital. Firstly, the design must be environmentally independent. Environmentally independent here means that the system must accommodate environment changes due to closures, construction, and moving objects. Environmental independence also means that the system *just works* whenever deployed in a hallway or room. The system does not need to be told about its environment; everything it needs to know, it will observe.

Secondly, the design needs to find >95% of the doorknobs in the hospital (high accuracy) and also not mistake other objects in the environment for a door knob (low false-alarm). The false-alarm rate should be less than 1 false alarm in 1000 frames of size 320x240. High accuracy and low false-alarms are top priorities because they almost solely determine the success of the prototype.

Prior work was done by Klingeil et al. [4] to locate and identify doorknobs using an SVM-based doorknob detector and a robotic platform. Unfortunately, the solution proposed by Klingeil et al. to identify doorknobs had an unacceptably high false-alarm rate of 15.6%. For a hospital environment, the robot cannot disinfect the wrong object 15.6% of the time.

Some solutions use RFID tags to tag doorknobs and eliminate the needs for any computer vision. These solutions have the potential to be more accurate than vision-based systems, since there are fewer chances for false-alarms with RFID tags [5]. The problem with RFID tags is that they need to be installed on every doorknob in the environment. The material and labor costs to achieve this are greater than a vision-based robot. Also, an RFID system is not easily and quickly deployable because of the installation time. A vision-based robotic solution is quickly deployable and less expensive than an RFID system, but work must be done to ensure its accuracy.

The robot here, equipped with just a laser scanner and a webcam, can autonomously locate and identify doorknobs using novel machine learning algorithms. The majority of work on the robot centered on identifying accurate real-time object detection algorithms and implementing an autonomous motion control algorithm. Some of the key challenges included inaccuracy and false positives from the object detection algorithm, slow performance, and inaccurate wall detection.

## 2 Algorithms and Approach

The task of locating and identifying doorknobs can be broken into two parts: moving around the environment in search of a doorknob and finding the doorknob when a doorknob appears in the robot's vision. The first task is taken by the motion controller and the latter by the object detector. While most work performed on the project centered on software, understanding and working around hardware constraints was essential. The Erratic™ robot platform, shown in figure 1 and loaned by Professor Saxena for use in this project, included all of the necessary hardware including motors, a laser rangefinder and a 5.0 Megapixel USB webcam. Figure 2 shows the interaction between the software and hardware components. The object detector was implemented first because the motion controller depends on output from the object detector.

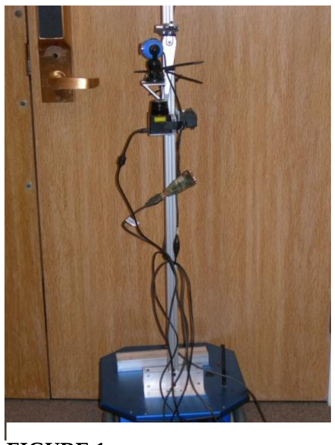

**FIGURE 1**

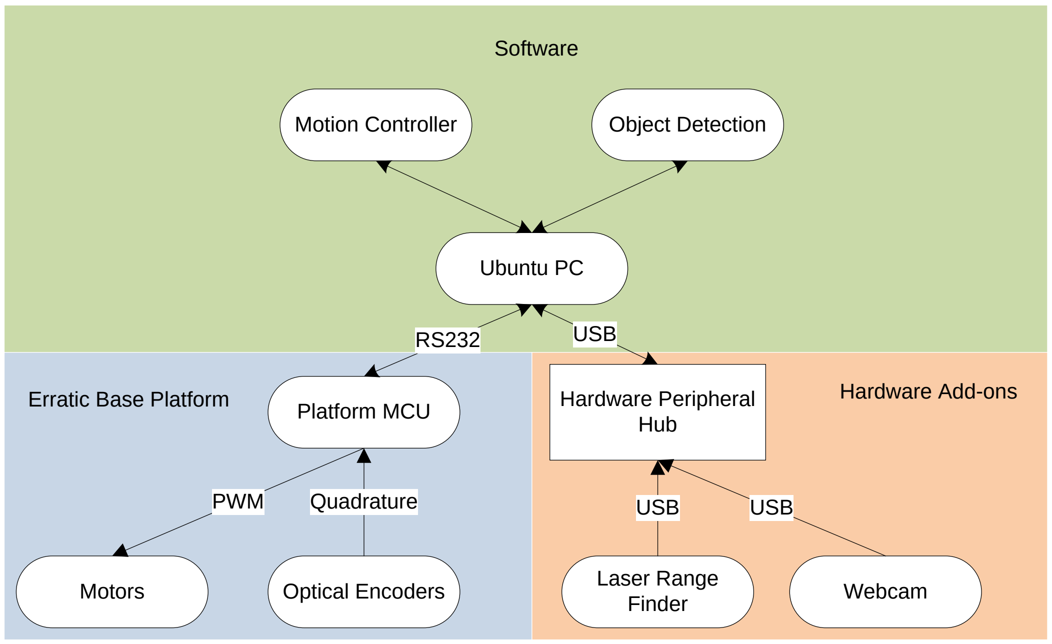

**FIGURE 2**

## 2.1 Object Detector

The goal of the object detector is to look at an image of the environment and determine, if any, the coordinates of a bounding box around the doorknob. The object detector had to reach an accuracy>95% and less than 1 false alarm in 1000 images. Based on an assessment of the best object detectors from the VOC2007 object detection competition [6] implementing such a high performance detector that could run on the Erratic™ platform would present a challenge. Furthermore, detectors differ in implementation ease, detection speed, and object specialization. In object specialization, for example, a certain algorithm can do well on grayscale images of simple, symmetrical objects, but do poorly on highly deformed objects in color images. The best classifiers perform well in a range of object types, like the UoTTCI and IRISA detectors in the VOC 2007 object detection competition [6].

The first design iteration used a haar-like feature classifier from Leinhart et al. [7]. This detector is typically used in face-detection, but its detection speed and accuracy made it a good first choice.

The Haar-like feature detector works by sliding a fixed-size window around the image. At each location, a classifier provides a yes/no decision on whether a doorknob is present in the window. An MxN input grayscale image is turned into a binary image where a 1 indicates a doorknob present in the sliding window and a 0 indicates no doorknob is present. If a cluster of 1's appear in close proximity, it is likely that there is only one doorknob located in the cluster, so a clustering algorithm tries to group clusters together to identify doorknobs. The most difficult step is implementing a good classifier to slide around the image.

The haar-like feature classifier is a cascaded and boosted classifier that detects accurately and can detect objects quickly. A cascaded classifier means that the entire classifier is broken down into a chain of N stage classifiers as shown in figure 3.

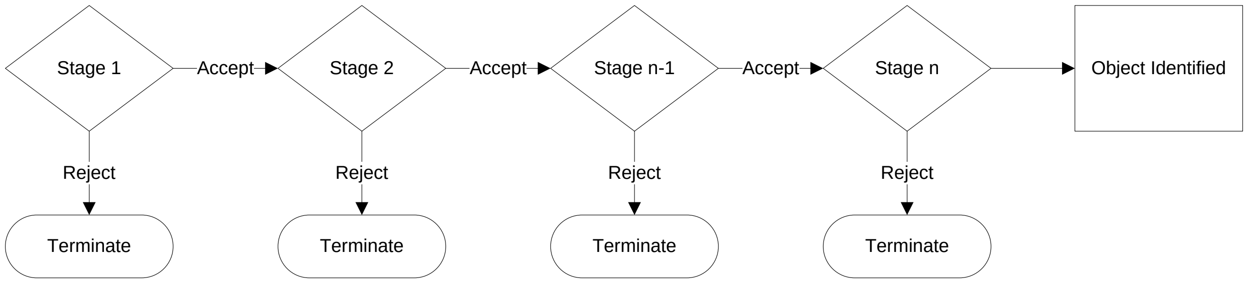

**FIGURE 3**

At each stage, a yes/no decision is made to either accept the image and pass it on to the next stage or to reject the image and move on. To be detected, an image must pass through all N stages of the classifier. To be rejected, the image just needs to be rejected by a single stage. The cascading helps to quickly and effectively narrow down the feature space. Boosting is used at each stage as a form of voting [8]. A set of weak classifiers are used at each stage and collectively decide whether an image can proceed to the next stage. Kearns [8] argues that a set of classifiers individually only slightly more accurate than random guessing can be combined to form a classifier of arbitrary accuracy. In the implementation here, based on the work of Lienhart et al., the weak classifiers are decision trees based on haar-like features.

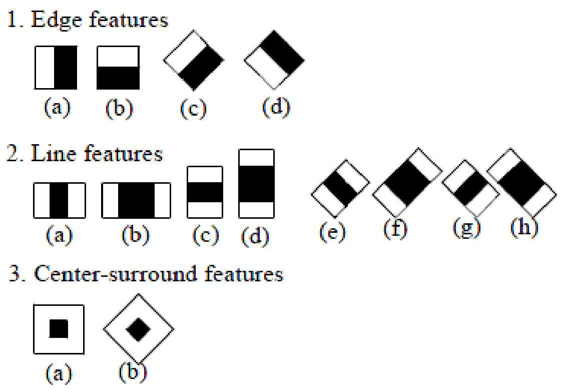

**FIGURE 4 [7]**

Haar-like features are the outputs of convolving the grayscale image with various filters. The filters shown in figure 4 from Lienhart et al. are the filters used in convolution.

Once adapted for use on the Erratic robot platform and with input from the onboard webcam, the detector was trained on a dataset from Upson Hall and tested in Philips and Rhodes hall. Unfortunately, after extensive testing, the detector could not perform to the requirement that accuracy be greater than 95% and less than 1 false alarm in 1000 frames. Typically, this detector performed with accuracy near 60% and about 484 false alarms per every 1000 image frames. More discussion of performance can be found in the test results section. This detector needed to be replaced by a more accurate detector.

After observing the misses and the misclassifications from the haar-feature classifier, it became clear that a new classifier had to be more robust to variations in size, shape and configuration of the doorknobs. This was the main failing of the haar-like feature classifier.

A parts-based classifer looks at an object as a collection of N parts, where N in specified by the user. The classifier is complex and is considered a departure from the paradigm that simpler classifiers yield better results than more complex classifiers. The classifier trains by identifying the various parts of the object, the labeling considered latent, and then constructing a grammar on how these parts join together. A grammar is used instead of a more strict and simple model because objects, like a bicycle, can take on many forms and configurations. However, it is always known that a bicycle, whether a mountain bike, road bike, or 19th century bicycle, has one wheel in the front and one wheel in the back, and handlebars attached to one of the wheels. The details of training and detection including how the parts are identified and then pieced together can be found in Felzenszwalb et al [9].

The parts-based classifier was written by Felzenszwalb for competition in VOC2007 in MATLAB and adopted for use in this project. To send images to the classifier, the classifier's MATLAB code was modified and called via Python scripts and MLabWrap. MLabWrap is an open source MATLAB wrapper for Python so that python scripts can access MATLAB code. Python is used to call the MATLAB classifier because Python has OpenCV libraries to access the onboard webcam and laser rangefinder.

The parts-based classifier showed huge performance gains over the haar-like feature classifier, giving an accuracy of 97% and only 4 false alarms out of 1000 images. These results meet the accuracy requirement for the solution, but the classifier proved slower than expected. A further discussion of speed and accuracy for the parts-based classifier can be found in the results section. An incredibly useful observation found that the classifiers performed much better (roughly 40% improvement in accuracy) on datasets where the robot was aligned in parallel to the wall and doorknobs. Further discussion of this observation can be found in the results section. This observation motivated the requirement for a more sophisticated motion control algorithm where the robot could only traverse hallways such that the webcam looked directly at the doorknobs and the walls.

## 2.2 Motion Controller

After evaluating the performance of the classifier while the camera looked directly at the wall (no object skew) and while looking at side-views of doorknobs, it became clear that better accuracy could be obtained by aligning the robot's camera with the wall. The goal of the motion controller is to drive the robot down hallways, while avoiding people and staying parallel to the longest wall in the robot's field of vision. A velocity trajectory is calculated to be parallel to the wall. This requires an observation of the environment by the robot to determine the location and orientation of the longest wall in the scene. To find the wall, a laser scanner is used to collect a point cloud of the environment along a plane parallel and 2 feet above the floor. A point-cloud hough transform was written to pick out the dominant lines in the scene. The hough transform moves lines from the (x,y) to a (θ,r) line parameter space [10]. Θ is the orientation of the line and r is the distance of the line to origin. The transformation is calculated as:

$$y = \frac{-\cos(\theta)}{\sin(\theta)} * x + \frac{r}{\sin(\theta)}$$

A major line is represented by a peak in the hough space and the peaks are detected by using MATLAB's built-in houghpeaks function. A wall in the scene is typically the longest line in the point cloud, so finding walls in the scene is equivalent to finding the most dominant line in the point-cloud. The dominant wall or line is the peak with the most votes in the hough space.

The motion controller is fed laser scan data from a Python script and the MLabWrap wrapper. Once the walls are identified, a velocity vector is calculated by scaling up the wall orientation vector $v = \alpha * \langle cos\theta | sin\theta \rangle$ if the robot is in the range of 3-6 feet from the wall. Fig 7 shows a point cloud with a wall detected in blue.

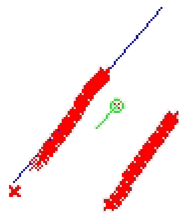
FIGURE 5

If the robot is closer than 3 feet from the wall, the webcam may miss doorknobs because the field of vision is reduced. If the robot is farther than 6 feet from the wall, then doorknobs may appear too small in the image frames and too difficult to detect. To solve this problem, the distance to the wall is calculated by looking at the radial component of the line in the houghspace and determining if the radius is less than 3 feet or greater than 6 feet. If the robot is outside this range, normal driving is suspended and the robot is turned $90^0$ and driven toward the wall to move within range. If the robot is in the 3-6 foot range of the wall, normal driving resumes by following the longest wall in the scene.

## 3 Experiments and Results

Testing and validation for the robot was performed offline by individually testing the motion controller and the object detector. In testing the best parts-based detector using laser correction exceeded its target accuracy, finding doorknobs 97.3% of the time and false-alarming only 0.10% of the time. The motion controller never missed finding the longest wall in the scene while running for 10 minutes.

### 3.1 Object Detector

There are two supervised object detectors that were considered for use in the robot to find doorknobs, a haar-like feature detector and a parts-based detector. The haar-like feature detector was designed for face detection and detects quickly, around 28 (320x240) frames per second, but ended up performing inaccurately when trained on doorknobs. The parts-based classifier was designed by Felzenszwalb et al. to compete in the PASCAL VOC challenge. Detection time was not a main concern for their research group and the detector performed around 1.45 frames per second on the doorknob dataset.

| Detector Type | Name | Num. Training images | Training Time* | Detection Speed* |
|---|---|---|---|---|
| **Haar** | Knob14.cls | 2000 | 2.8 hours | 28 fps |
| **Parts-Based** | Knob_final.mat | 579 | 7 hours | 1.45 fps |

*The algorithms were trained on an Intel i7 930@2.8GHz 3GB RAM running Ubuntu 9.10

The parts-based detector is far more accurate than the Haar-like feature detector. The haar-like feature detector took the greatest amount of time to optimize because of the large number of parameters.

### 3.2 Datasets

The dataset used to train the different classifiers must be identical for an accurate comparison. The dataset used to test the different classifiers must be identical, as well, for an accurate comparison. Since one of the criteria for the robotic disinfector is to detect and clean many different sizes and shapes of doorknobs, the training and testing datasets have doorknob images from different buildings. Each dataset is a set of images taken by driving the robot around hallways and recording the webcam frames. Each dataset was manually split into images with doorknobs (positives) and images without doorknobs (negatives). The positive images were tagged manually with bounding boxes using software originally written by Florian Adolf [11] and modified by the author. Almost every positive image contained exactly one doorknob; rarely (<0.1%) did a positive image have two doorknobs in it. All datasets were taken from buildings at Cornell University. Some positive images are shown below in figure 6.

| Dataset Name | Location | Positive Images | Negative Images | Comments |
|---|---|---|---|---|
| **10.26.2010** | Upson/ Rhodes | 134 | 1851 | |
| **10.29.2010** | Upson/Phillips | 205 | 1647 | Has side-views of doorknobs |
| **11.2.2010.2** | Rhodes | 309 | 1417 | |
| **11.3.2010** | Upson | 335 | 2805 | |

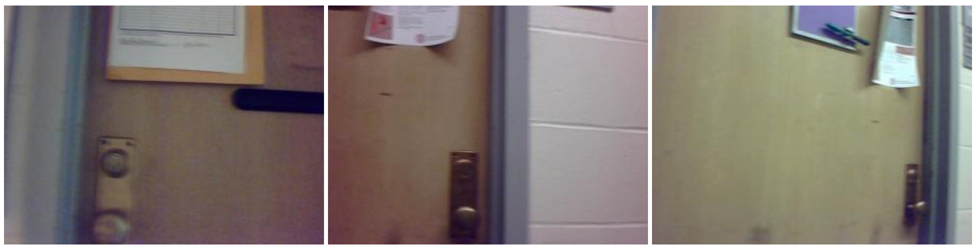
FIGURE 6

The haar classifier requires the following parameters to be specified: number of positive images in the dataset, number of negative images in the dataset, number of stages, and number of splits.

After 15 haar-like classifiers were trained using different dataset sizes, number of stages, and stage-propagation criteria, classifier 14 proved the most accurate classifier with false accept rates<55%. Classifier 14 used 10 stages, a single stump weak classifier, 1000 positive training images, and 1000 negative training images. Classifier 14 was taken for further performance review, but was deemed too inaccurate, hitting only 53.47% of its target in the test data set and false accepting 1454 times in 3000 images.

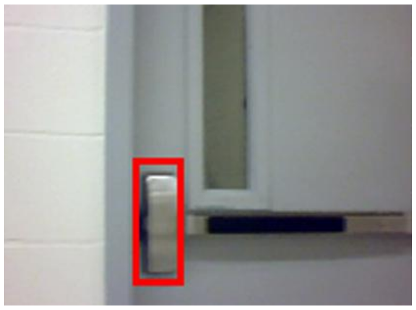
FIGURE 7

The parts-based classifier was used as an alternative to the haar-like feature classifier. It performed extraordinarily well, achieving 97.28% accuracy and recording a 0.10% false alarm rate.  The false-alarms were strikingly similar to a real doorknob. Figure 7 shows a false-alarm where it is likely the classifier got confused because the object in the bounding box looks like the mounting plate many doorknobs in the training set had. Ironically, the fire door shown here should be disinfected as well.

| | | | | 11.3.2010 Test Dataset | | |
|---|---|---|---|---|---|---|
| **Type** | Train Dataset | Number of Positives | Number of Negatives | Hits | False Alarms | FA Rate |
| **Haar** | 10.26.2010 | 1000 | 1000 | 45.92% | 1456 | 46.00% |
| **Haar** | 11.2.2010 | 1000 | 1000 | 53.47% | 1454 | 46.40% |
| **Parts** | 11.2.2010 | 279 | 300 | 97.28% | 2 | 0.10% |

## 3.3 Laser Correction

Dataset 10.29.2010, containing both front and side viewed doorknobs. allowed for a key observation that detecting doorknobs head-on is more accurate than side-viewed doorknobs. On dataset 10.29.2010, the parts-based classifier hit around 90% of the doorknobs that were viewed head-on, but hit only 50% of doorknobs viewed from the side. This observation led to the addition of the parallel-path requirement on the motion controller.

## 3.4 Motion Controller

The motion controller performed well at identifying the longest wall in the laser-scanned environment and

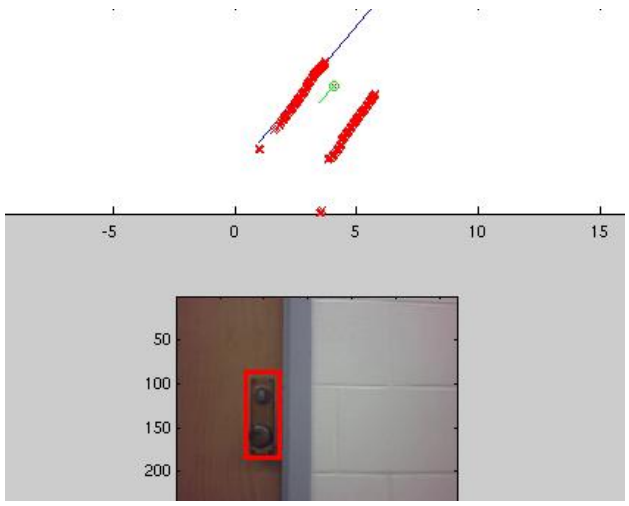

FIGURE 8

calculating this wall's orientation and offset parameters. In the 13.2 minute test run with 3074 laser scans, the motion controller did not miss calculating the longest wall once. Figure 8 shows the motion controller identifying the wall to follow and the orientation of the robot. The blue line in the top pane is a linear approximation of the wall. The lower pane shows a positive image and the bounding box surrounding the doorknob detected by the parts-based classifier.

## 4 Conclusion

At the end of the project, there are still many areas left to complete and issues in need of resolution. One of the major concerns with the results is the issue of overtraining. The performance of the object detector is stellar – reaching 97.3% accuracy and only a 1/1000 false alarm rate; however, the results are so good, particularly the false alarm rate, that further research needs to be done to verify that the object detector is not overtrained on images from a small set of doorknob types. Overtraining was, in fact, explored here by testing the robot in a different building on campus with different knobs from the training set. Further exploration should be done given the very good performance recorded here. To verify the performance of the classifier, an image dataset can be taken from a completely different location with even more variation in doorknob type. For example, some doorknobs have mounting plates, some have locks directly above the knob, or some have locks on the knob. Ensuring that the detector performs acceptably on these varieties of doorknobs is essential to its universality.

The research shown here demonstrates the feasibility of a highly-accurate and mobile object detector. The software written for this project met the project's accuracy, environmental independence and mobility requirements and exceeded the performance of solutions in the field. The work outlined in this paper is part of a larger project for the author's Master of Electrical and Computer Engineering project.

## Works Cited

[1] Andrew Pollack. (2010, February) New York Times. [Online]. http://www.nytimes.com/2010/02/27/business/27germ.html?_r=1&em=&adxnnl=1&adxnnlx=1267412412-yP2bfl/3pu4+g34XVmluJA

[2] Stephen T. Abedon. (1998, May) Nosocomial Infections. [Online]. http://www.mansfield.ohio-state.edu/~sabedon/biol2053.htm#black_1996

[3] John G. Bartlett, Neil R. Blacklow Sherwood L. Gorbach, *Infectious Diseases*.: Lippincott Williams & Wilkins, 2004.

[4] Ashutosh Saxena, Andrew Y. Ng Ellen Klingbeil, "Learning to Open New Doors," *Int'l conf on Intelligent Robots and Systems (IROS)*, 2010.

[5] Lin Zhong, Matti Hiltunen, Rittwik Jana Ahmad Rahmati, "Reliability Techniques for RFID-Based Object Tracking Applications," *ACM*, 2007.

[6] Mark Everingham, "The PASCAL Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision*, 2009.

[7] Rainer Lienhart and Jochen Maydt, "An Extended Set of Haar-like Features for Rapid Object Detection," *Proc. IEEE Int"l Conf. Image Processing*, vol. 1, pp. 900-903, 2002.

[8] Michael Kearns, "Thoughts on hypothesis boosting," *Unpublished manuscript*, 1988.

[9] Pedro F. Felzenszwalb et al., "Object Detection with Discriminatively Trained Part Based Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, 2010.

[10] Richard O. Duda and Peter E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, 1972.

[11] Florian Adolf. Willow Garage. [Online]. http://opencv.willowgarage.com/wiki/ObjectDetection