Survey and Implementation of Computer Vision Techniques for Humanoid Robots

Aaron Baucom

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2014-83
http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-83.html

May 15, 2014

Acknowledgement

I thank Professor Don Wroblewski for aiding in the development of this project and advising the team. My gratitude to the other members of the group - Hellen Lopez, Brock Bearss, Wenbo Wang and Julian Zhang for their efforts as well.
More than anything, I thank my parents - Iris and Terry Baucom for helping me reach this point and continually supporting me throughout my life.
Finally, I'd also like to thank Jeffrey Baucom and Eva Mae Natividad for their support and encouragement.

University of California, Berkeley College of Engineering

MASTER OF ENGINEERING - SPRING 2014

Electrical Engineering and Computer Science

Visual Computing and Computer Graphics

Survey and Implementation of Computer Vision Techniques for Humanoid Robots

Aaron T. Baucom

This **Masters Project Paper** fulfills the Master of Engineering degree requirement.

Approved by:

1. Capstone Project Advisor:

Signature: _____ Date _____

Print Name/Department: Donald Wroblewski – Fung Institute for Engineering Leadership

2. Faculty Committee Member #2:

Signature: _____ Date _____

Print Name/Department: Ruzena Bajcsy – Electrical Engineering and Computer Science

Survey and Implementation of Computer Vision Techniques for Humanoid Robots

Submitted in partial fulfillment of the requirements

for the degree of

Master of Engineering

in

Electrical Engineering & Computer Science

The Graduate College

and Fung Institute of Engineering Leadership

at

University of California at Berkeley

Aaron T. Baucom

May 15, 2014

TABLE OF CONTENTS

TABLE OF DEFINITIONS AND TERMS

MNIST Modified National Institute of Standards and Technology digit-recognition database

PHoG	Pyramidal Histogram of Gradients
DARPA	Defense Advanced Research Projects Agency
PASCAL	Pattern Analysis, Statistical Modeling and Computation Learning
GPU	Graphics Processing Unit
FPS	Frames Per Second
IR	Infra-Red
LBP	Local Binary Patterns
SMA	Shape Memory Alloy

ABSTRACT

In this paper, I survey general computer vision techniques and their applicability to the space of real-time humanoid robotic systems. Specifically, I begin to construct a system which can identify and locate objects of interest at the standard 30 FPS of a constant video stream within a constrained power budget. This is done in academic code first, using a cross-validation framework to evaluate accuracy. Preliminary results show difficulty achieving a stellar accuracy rate, though some methods approach reasonable levels. Furthermore, various portions of the vision system are ported all the way down to hardware, while others remain in academic development code environments. All work was done as part of a capstone team whose goal is to pursue academic methods which can be used to reduce the cost and improve the performance of humanoid robot systems.

INTRODUCTION

Much of the work done in the computer vision research is designed to improve accuracy to the utmost degree, typically involving computationally intense, slow, or power-hungry algorithms. On the other hand, many of the ideal applications for these techniques are areas where the addition of a powerful computer system for vision would strain budgets for power, heat dissipation, or dollars. In this project, our team focuses on industrial and manufacturing robots – one example of such a space. Specifically we analyze the key technologies and innovations which will enable the rise of cheaper humanoid robot workers which could have profound effects on our society.

Recent forecasts by the International Federation of Robotics suggest that low-cost, adaptable robots will be a driver for new growth in the industrial robotics area [1]. Scholarly literature suggests that new techniques are in development for lowering the cost of human-like hands with increased adaptability and accuracy are progressing [2, 3]. Beyond mechanical prowess, these systems need increased adaptability and "smart" features in order to press into new markets. Our goal is to analyze the literature of computer vision systems to determine which could enable these new features and fit into the low-cost low-power budget of industrial robotics.

The potential social and economic changes have already been previewed in the reaction to a few emerging technologies. The first is the Baxter robot from Rethink Robotics which has already begun to call into question the future of works and the role of humans in manufacturing economies. An IEEE article on this topic revealed that it is already being seen as an aid and extension of the human capability [4]. According to Guizzo and Ackerman, it's not designed as a replacement for human labor, but to free them to do more engaging and interesting work. Contrastingly, the rise of 3D printing technology does directly question the role of workers in creating physical objects [5].

Project Level Goals and Scope

This project was developed by a large team of students across many different technical concentrations. In addition to my work in developing the computer vision and location techniques, there are other computer science students working on the smart control and actuation of the robotic joints. There are mechanical engineering students who are developing the precise and compact anthropomorphic hand using novel techniques and materials. Finally, we have students who are developing bipedal leg and foot hardware and control techniques. Each of these components addresses an area where incremental improvements can make robotics and automation viable in markets where it otherwise would not have been.

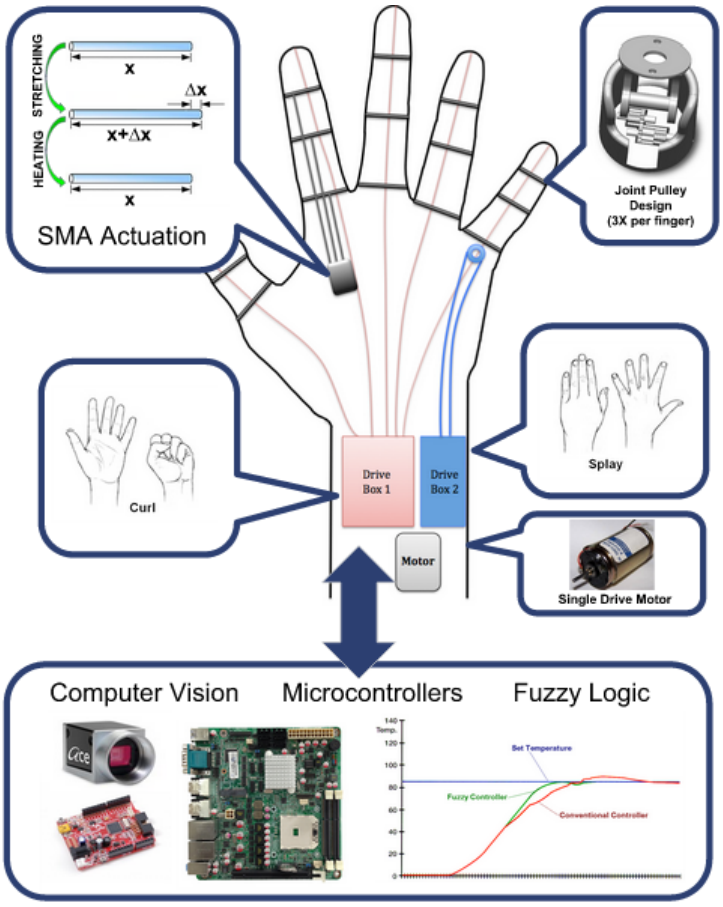

Figure 1: Overall system architecture of this project.

Figure 1 details the overall interactions of the project's subsystems. The mechanical design of the robotic hand is developed in tandem with an SMA actuation scheme. Similarly, there is an algorithm developed using fuzzy logic that drives the movement and the arm and hand components. Finally, the computer vision aspect which is detailed in this paper will monitor the visual field around the robot and feed instructions to the other components of the system

LITERATURE REVIEW

There are two aspects to this project which are based in a rich background of scholarly literature. We break them out as Object Detection and 3D Reconstruction. Both are necessary for a coherent computer vision application that interfaces with real-world data, such as that for a humanoid robot. Object detection is required to identify what materials of interest are in the field of view. A reconstructed 3D model of the surrounding world is necessary to understand where the objects are actually located. Together they allow a robotic system to identify and manipulate their environment.

Object Detection – Learning Frameworks

One of the earliest approaches to object classification and machine learning in general was the nearest-neighbor classifier as discussed in Cover and Hart [6]. These systems rely on mapping training examples as points in some D dimensional space where distances (typically Euclidean) can be computed. When a new point is classified, the algorithm calculates the distance to all other points and assigns the class of the point found to be nearest. The default implementation of such an algorithm defers all computation until classification time, which is unacceptable for real-time applications. Furthermore, the naïve search for the nearest neighbor typically involves n distance calculations. While optimizations like those discussed in Arya et al. can be used to reduce the search time by pre-processing the dataset, they introduce inaccuracies and more complex machineries [7]. These factors are aggravated by increasing dimensionality which render them less attractive for computer vision applications.

Alternative approaches such as support vector machines and decision trees (or forests) have been fallen into better favor with the computer vision community due to their lower classification-time computational costs. Support vector machines are popular for their simple linear form as well as their adaptability to more complex decision spaces by using different (and sometimes exotic) kernels [8]. The landmark Viola-Jones methodology uses AdaBoost, a decision forest derivative which enables rapid classification with low steady-state computational costs [9]. Furthermore, it effectively searches a large feature space which can lead to better classification. Both of these cases are applicable to our space of real-time, power-constrained computer vision systems.

Finally, the more exotic space of neural networks can be applied to object detection and computer vision techniques. These classifiers are seeing a renaissance in modern literature under the moniker of deep learning. One of the most important papers for this field was Yann LeCun's handwriting recognition system LeNet and the unprecedented MNIST accuracy levels it generated [10]. On the other hand, these systems are often criticized for the complex hyper-parameter space which must be tuned by the system architect and the occasional tendency to become stuck in low accuracy states [11].

Computer Vision – Feature Selection

There are a number of influential papers on computer vision and selection of image features. The first I will discuss the popular PHoG descriptor. The PHoG descriptor computes the gradient at each pixel and bins the angles into a normalized histogram [12]. It has been used to great success in smaller-sized image databases such as the original human body recognition corpus explored by Dalal and Triggs [12].

The next feature set is used in the Viola-Jones detector and is a derivative of the Haar feature descriptor [9]. These are block-based image descriptors which are easy to compute on a given image but can be extremely powerful. One of the most interesting results of the Viola and Jones paper is that some complex shapes can often be detected with only a few box-shaped objects.

A third option we will explore is called local binary patterns of LBP. Descriptors of this form are targeted at identifying image texture. LBP features are circular collections of pixels which establish a relationship between the grayscale intensity of pixels along the perimeter of the circle as well as that in the center [13]. LBP features were first used to characterize uniform images of textures such as canvas, cloth, and paper but were gradually expanded for full-fledged use in characterizing other forms of images.

Finally, a new innovation in the feature detector space is the deformable parts model. In this system, Felzenszwalb et al. develop a system that uses a root feature as well as a few adjacent descriptors which can be located in different spots in relation to the root [14]. The big success here is that their example runs on the PASCAL dataset which is much more relevant to our application. The authors show an example where extreme foreshortening of the object (a bike) does not prevent them from training a good model.

3D Reconstruction

Epipolar geometry is the fundamental approach that is used to arrive at depth information based on geometrical cues. Traditionally, these cues are from a binocular stereopsis system, like the one that we humans use with our own two eyes. In these frameworks, two cameras are separated by a reasonable margin and an algorithm is used to reduce the information from the two images into a solvable geometry problem. This typically involves finding corresponding points of interest and solving an over constrained linear system [15]. Unfortunately, these techniques do not translate well to the low-cost humanoid robot space. Two cameras can be very expensive and the additional compute power to process the data is non-trivial.

A traditional approach to identifying corresponding points is an algorithm called RANSAC. RANSAC is a method which uses multiple rounds of voting between suspected matching feature points. These features can be computed using a corner or blob detection algorithm. The algorithm then compares the

suggested results for various pairs of suspected feature matches [16]. These feature matches can be fed into the linear system and generate depth data for each new pair of images that comes from the sensors.

A different, more cost-effective approach has been taken by the popular Kinect sensor developed by Microsoft. Instead of using multiple cameras, the system uses an infrared projection system whereby it can eliminate the additional computation and camera necessary. This sensor actually emits patterns that correspond to locations into the world which are then read back by an IR camera. The result is a matrix of coordinate pairs without the need for RANSAC or any similar algorithm. The infrared sensor based approach shows acceptable accuracy and is largely limited by the error term which scales with depth [17]. This type of system is ideal for the low-cost, low-complexity space of humanoid robotics and I explore a new take on this technology.

METHODOLOGY

Data Corpus

Since our target is humanoid robotic system, the team decided to model task selection off of existing robotics challenges in this space. A premier example of this is in the 2013 DARPA robotics challenge [18]. This challenge tests a range of human-like activities such as driving, navigating irregular terrain, and tool use. I selected the task of hand-held electric drill use as the activity of choice. In order to address the space, I built up a corpus of images that focused on both positive examples (those with electric drills pictured) and even more negative examples which feature objects similar to drills or picture backgrounds where you would often find electric drills. These images were scraped from the Google image searches and manually classified, cropped and cleaned. In the end, I amassed over 4,000 positive and negative images which could be used to train the classifier.

Comparing our corpus to MNIST and facial recognition databases illustrates why generalized object detection is considered difficult. Whereas digits and faces are rigid and mostly planar (as well as rarely foreshortened or distorted), objects such as drills are often deformable and have an intricate geometry. This complex 3D geometry causes the same object to look drastically different from various perspectives. This is due to what artists call foreshortening, a product of the projection of a 3D world onto our 2D retina. These distortions are not seen in the planar case can limit the efficacy of a strict single model classifier. Finally, we notice that many of our images of drills in use are photographs of workers holding the object in their hands. This introduces occlusion complexities since the object appears to be broken, but is actually just covered by the user's hand.

Cross-Validation Framework

In the field of machine learning the cardinal sin is to train your learning algorithm using a priori knowledge of the test data set. In real use cases, each new data point will be a test set and you do not have any ex-post knowledge of an event that has not yet occurred. As a result, to evaluate algorithms in ways that closely mimic real use accuracy rates, the research field generally uses data subsampling methods. The most common of these is cross-validation, a process in which the data set is divided into *n* folds. We iterate through each fold, leaving it out of consideration and training our classifier on the remaining data. Then accuracy can be estimated on the held out fold of data. This is repeated *n* times and the resulting accuracy is averaged [19].

In our evaluation, we will perform two levels of data exclusion. First we separate approximately one-fifth of our data into a test set – never to be viewed or used until we decide on the best classifier via cross-validation. Then we divide the remainder up into four folds with which to perform cross-validation. The ideal number of folds is said to be higher, typically 5 to 10, but as fold size decreases, accuracy can

fall and bias creeps into the measurements [19]. As a result, due to our limited corpus, we choose a simple four-fold cross-validation scheme.

Algorithm Evaluation

In this paper, we will survey a number of different machine learning and computer vision frameworks as well as different configurations thereof. As a result, we will strive to evaluate each one using the cross-validation method and discover the most accurate single algorithm. This will form the basis of future development on the humanoid robot and we will evaluate it again using the clean test data set.

The decision forest derived algorithm we use for training classifiers has two hyper-parameters that affect its performance. They are the number of training iterations and the target false-positive rate. For each feature type we attempt, we will search a reasonable space of these parameters and present the best error rate found. This is a traditional machine learning approach and this type of tuning is what makes the application of a cross-validation methodology so important.

Targeted Algorithms

In this project we focus on the AdaBoost decision forest methodology that is detailed in the Viola-Jones paper [9]. Unlike Viola-Jones, we broaden our search to the three different feature types: HoG, LBP, and Haar. We will evaluate each approach on our data corpus to discover the best system architecture for robotics applications.

DISCUSSION

This project saw the creation of many subsystems which were required to address various aspects of a difficult problem. Below, I enumerate the areas individually, but they all fundamentally attempt to solve some aspect or sub-problem of enabling the robot to identify and locate objects. It can largely be

divided up into recognition and reconstruction tasks or data driven and algorithm driven tasks. Each have their merits and are discussed below.

Feature Space and Associated Challenges

I spent a quite a large amount of time collecting and labeling a large dataset to train our classifier. The corpus was largely negative images – of the 4,100 images, just over 1,000 were positive examples of drills. This imbalance is due to what we call a false positive bias. The false negative error rate is something that is much more concerning than the false positive error rate. In our application, a robot that sees a false drill will move to approach and inspect the object. This will likely cause false positives to disappear after the robot gains new data. Conversely, if a robot never detects a true drill, it will make no effort to further investigate or gather new data.

To that end, we aimed to populate the corpus with both random negative examples as well as specifically chosen corner cases that are likely to generate false positives. Things like nail guns, hand guns, drill presses, and toy drills are very similar to the positive examples and we wanted to make sure these were represented in our training data to discover the best possible classifiers. In all likelihood, these objects will be misclassified and cause poorer performance than a more easily separable dataset, but we believe that this methodology is best for the future applications of our project. As you can see in Figure 2, these cases manifested themselves, but some new irregularities were discovered as well. The first is that we saw even more extreme occlusions or orientations of the drill relative to the camera than we expected, and we expect this to lead to poorer performance. Finally, we noted a variability that was higher than we expected in the actual boundary of the drill class. We see drills with radically different geometry and in this case, a drill that is be broken down into two different shapes.

Figure 2: From top left, clockwise: a foreshortened drill in scene, a drill with a normal shape but interrupted form, a similarly shaped non-drill object, an ideal drill form, and a drill with a vastly different shape.
Images scraped from google image API [20] [21] [22] [23] [24].

RGB-based Canny Edge Detector

One experimental area we explored was the use of color-based edge detection and making the most of the full RGB spectrum that we had from our high quality camera. Traditionally, images are converted to grayscale before applying any edge detection methods. In this method, we execute a Canny edge detection algorithm on each color channel before combining the three. Our sample input image will be the colored photo of the bus in Figure 3. As you can see in Figure 4, there is additional information which appears in this image. Nuanced edges arise from some regions like the bumper of the bus due to both real edges and lighting constraints. In other cases, such as the rear windows of the bus, edges that

had been suppressed before reemerge (albeit faintly). These edges had been suppressed because the dark red and the greenish-blue of the window glass mapped to similar grayscale values.

Figure 3: a colorful example image which we use to explore the space of RGB edge detection. Scraped from google images [25].

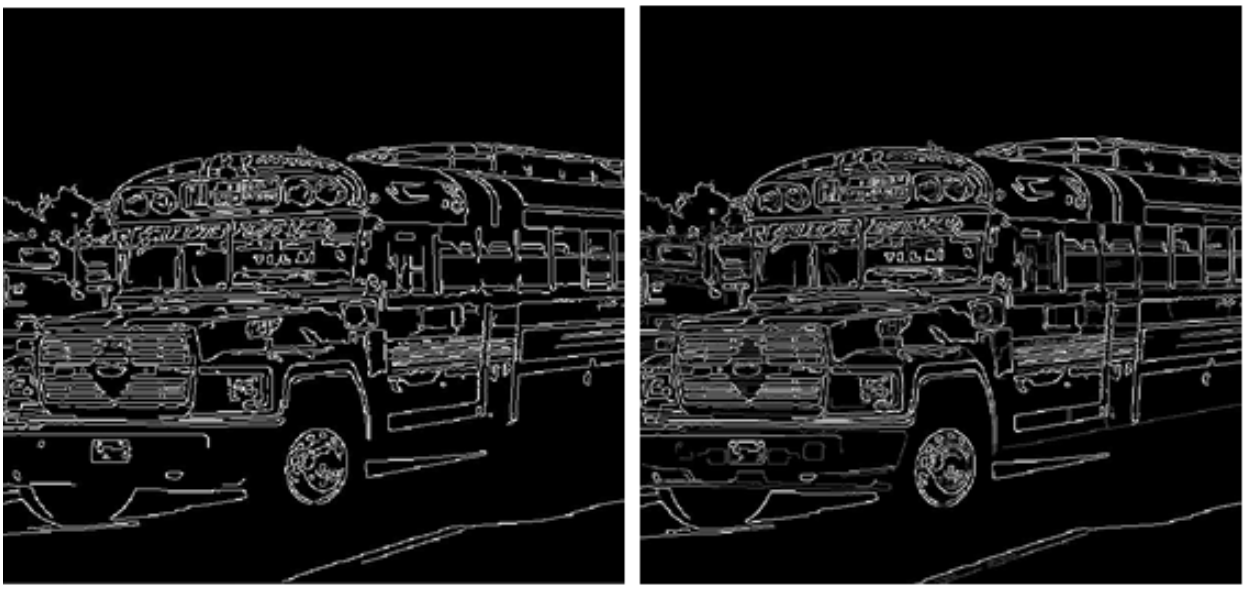

Figure 4: A grayscale Canny implementation on the left versus an RGB canny on the right. Notice the additional detail in the windows, bumper and sides of this bus.

In the end, the application for this modification is relatively limited. While some real edges can be recovered, it introduces additional edge noise into the image and sometimes becomes overly sensitive. Furthermore, the Canny edge detector is a more intense workload than RGB merging. For this reason, swapping their order results in a slightly higher workload. Since our goal is to reduce the cost and increase the efficiency of this pipeline, we have decided not to include this step in our system.

PHoG Feature Implementation

Here we explore the existing work of histogram-based feature descriptors. As was mentioned earlier, this work was really explored well by Dalal and Triggs [12]. We notice an interesting problem of image scale when implementing our PHoG feature detector in Figure 5 and Figure 6. Each image in our heterogeneous corpus has its own sweet spot for generating smooth edges to go into the HoG constructor. If we use too large of a scale with no smoothing, the edges become scattered and meaningless. On the other hand, if we down sample our image too far, we lose information about the contents. The HoG becomes dominated by the less informative vertical edges and the smooth round sections become crude diamond and square shapes.

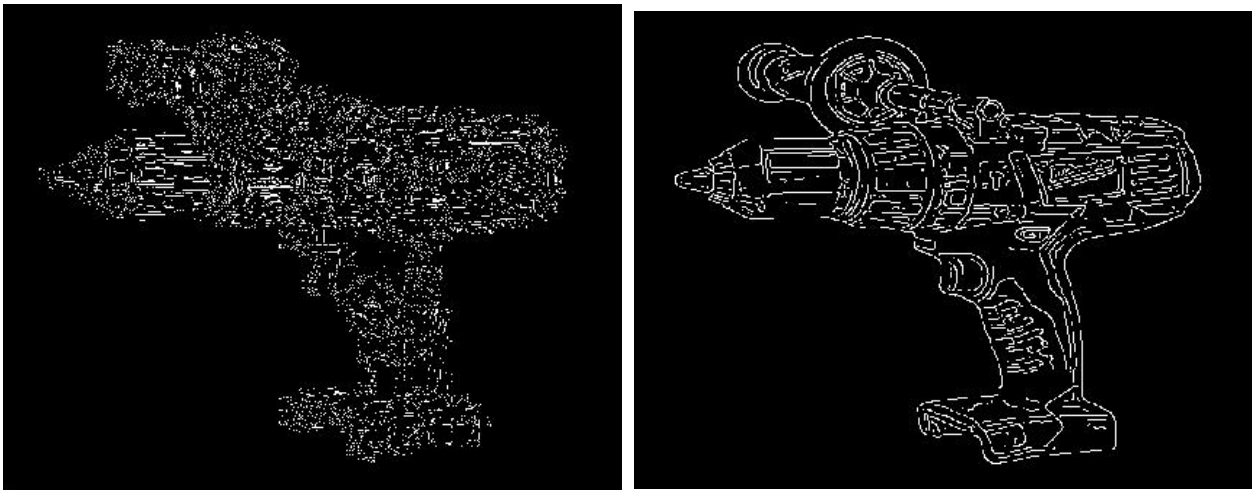

Figure 5: The input to the PHoG computation we used was a Canny edge detector. Notice how scale sensitive this detector is. The left image is at native resolution while the right image has been down sampled.

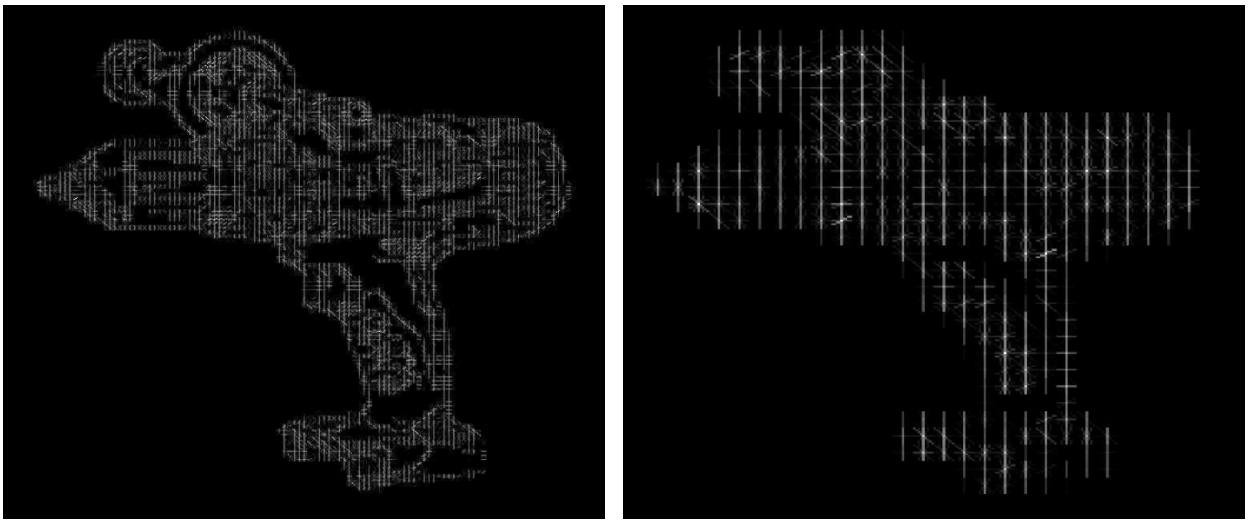

Figure 6: This is a representation of the HoG features at each window. The left image shows which angles dominate various sections of the drill. The right image shows the downside of over down sampling – the image has become less representative of the drill as a whole.

Haar Features

Next, we train using the original Viola-Jones framework using Haar features. Figure 7 shows approximately how Haar features work. Given a patch in the image, it sums the intensity values found in those patches and then adds or subtracts the sums in specific pattern.

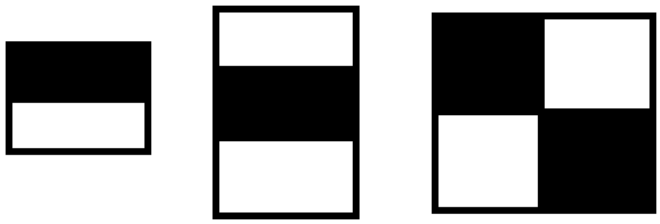

Figure 7: Visualization of Haar Features

3D Reconstruction

We start with a general two-image depth reconstruction algorithm. This algorithm computes the gradient and intensity of a local window around each pixel and finds the closest match in the other image. This algorithm assumes a perfect height match between the two images since it does not do full blown

feature finding and RANSAC. Our addition to this approach is to utilize feature projection – creating a feature in the world which is able to be tracked in real-time.

Hardware Porting and Helper Functions

Finally, we discuss the process of porting to hardware. We run our tests on an AMD APU consisting of a Trinity quad-core CPU and a Raedon HD7660G GPU. While much of our exploratory code was created in MATLAB, the final implementation details are constrained to the domains of Python, C, and C++. This is due both to hardware limitation and the pursuit of more optimized code. The process of porting to real hardware was not complete at the time of this writing, but certain helper functions such as GPU optimized partial gradient calculation and convolution calculations were completed. Similarly, portions of the 3D reconstruction code were developed fully on real hardware from the ground up.

RESULTS

Object Recognition System

As mentioned in the methodology section, our approach uses four-fold cross validation to evaluate the algorithms we developed over the course of the project. In addition to this approach, we also investigate the effects of scaling this approach up to include more data. Figure 8 illustrates this pattern for the three major feature frameworks we used. HoG responds the best to data scaling, showing a smooth decline and an impressive overall error rate. The LBP features respond well initially, but experience a steep regression when scaling to the full data set. Similarly, Haar features are even worse in this regard and scale poorly across the board. This is quite a surprise given the traditional view of machine learning and big data. Often it is assumed that more data can be panacea for high error rates, but in this case we don't see that at all. It is possible that our corpus is still far too small and that what we are seeing is noise due to the image partitioning or slight algorithmic details.

In each case, we have tuned each classifier on a number of different parameters.

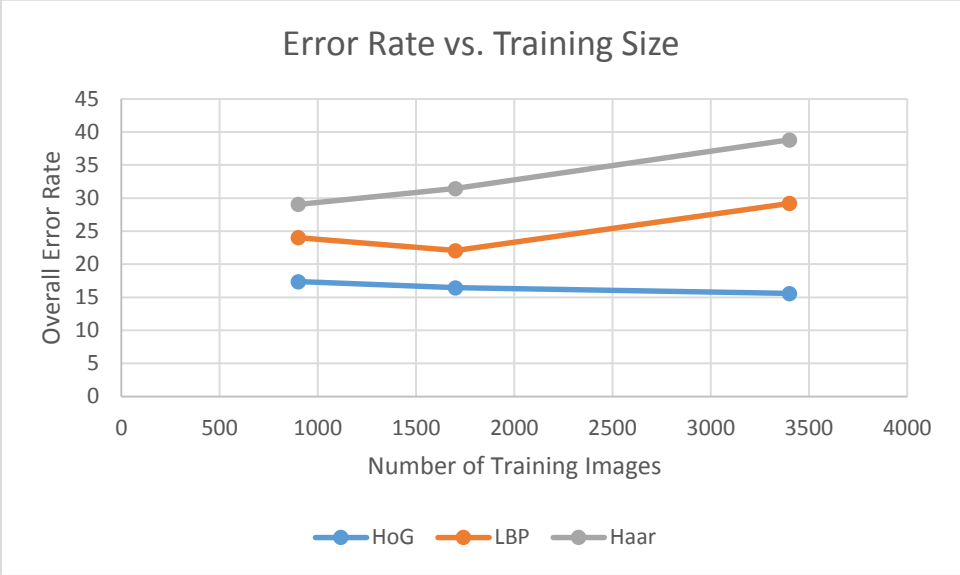

Figure 8: Different feature systems respond differently to more training samples.

Next, we breakdown the full-scale error set in further detail. Figure 9 details how each of the false positive and false negative error rates contribute to the overall error. This paints a very different picture than the previous analysis. Now it appears that the HoG classifier is being selective in the direction that we established would be less favorable for robotic applications. On the other hand, the Haar detector achieves a great false negative error rate despite its decidedly lackluster overall performance rate. LBP splits the difference, showing a reasonable overall and false negative error rate.

Full-Sized Training Details	HoG	LBP	Haar
Overall Error Rate (%)	15.6	29.2	38.8
Errors Due to False Positive (%)	4.3	23.7	35.8
Errors Due to False Negative (%)	11.3	5.5	0.30

Figure 9: Error breakdown for full-sized training results

In addition to assessing accuracy performance, we also want to evaluate our methodology in terms of speed and latency. While much of object detection code is still in an academic, unoptimized state, the performance is still quite reasonable. We find that all types of classifiers can generally process a single new image in less than 1 second. Averages hover around 0.6 seconds, which is far from real-time, but still

quite good for academic code. We believe that algorithmic improvements and optimizing this code for our hardware could easily bring this within the realm of real-time image processing.

3D Reconstruction Attempts

Our 3D reconstruction code was functionally incomplete at the time of this writing, but multiple portions of the system have been fully implemented on hardware. A crucial component of this is the feature locator code which finds the projected feature in the camera. This was demonstrated to run in approximately 0.03 milliseconds per frame, enabling a frame rate much higher than 30 FPS.

CONCLUSION

Future Work

Given our satisfactory but not extraordinary accuracy results from Viola-Jones and PHoG, it is likely that we have some retooling to do in our database. For comparison the famed MNIST database is comprised of 60,000 training images and a 10,000 image test set. Furthermore, the images from this database are of much lower dimensionality than the images we attempt to train. In this project, we experimented in software and then introduced hardware. In some regards though, it may be better to fix hardware parameters such as resolution and image processing pipelines prior to applying computer vision techniques. If we could capture training and test data using actual hardware in a repeatable way, we might see a big boost in performance.

Similarly, we were not able to finish porting our classifiers to hardware platforms. This severely limited our ability to evaluate meaningful runtime data and take power measurements. Furthermore, implementation details of image processing pipelines and GPU acceleration could lead to disproportionate speedups in certain algorithms. The block-based nature of the Viola-Jones' Haar descriptors especially lends themselves to accelerated computing. Finally, though we were able to

evaluate quite a few frameworks and descriptors, we had to avoid the more exotic neural network systems and deformable parts models. This is disappointing because these techniques are some of the most cutting edge and can give rise to very accurate classifiers. Future work might explore some of these more exotic systems in earnest and potentially discover a more optimal framework, but for now we are limited by the data we could gather.

Reflection

In this paper we have presented a survey of historical and modern computer vision systems. We experiment and evaluate some of the more promising systems from modern research literature. In general, it appears that general object detection is still a difficult problem to perform accurately, especially when we limit ourselves to a certain compute budget. That said, there are many promising techniques within computer vision which can be applied to enable the high-performance, affordable robotic systems on the horizon. Broadly, we have explored the space and design considerations that should be taken into account when building a full computer vision software stack.

The first lesson is to develop a high-quality, consistent, and large data corpus in order to best characterize the problem. Despite the size of our corpus and the labor we went through to construct it, we found regression in accuracy as we trained with more data. Since we were limited in how far we could push these tests, it's not clear whether this is a limitation with our problem or an inconsistency that would be resolved given larger folds of data. The second lesson is to explore the space of possible algorithms and select the one which works the best for a given application. There are many performance, affordability, and task-specificity tradeoffs to be made for each system. With a combination of a more rigorous and thorough data corpus and a well selected learning framework, we believe this and other problems are tractable.

References

[1] IFR Statistical Department, "Industrial Robotics Statistics," International Federation of Robotics, 2013. [Online]. Available: http://www.ifr.org/industrial-robots/statistics/. [Accessed 14 April 2014].

[2] S. Nasser, D. Rincon and M. Rodriguez, "Design of an anthropomorphic underactuated hand prosthesis with passive-adaptive grasping capabilities," *Florida Conf. on Recent Advances in Robotics and Robot Showcase,* pp. 25-26, 2006.

[3] M. Luo, T. Mei, X. Wang and Y. Yu, "Grasp characteristics of an underactuated robot hand," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, 2004.

[4] E. Guizzo and E. Ackerman, "The rise of the robot worker," *Spectrum, IEEE,* vol. 49, no. 10, pp. 34-41, 2012.

[5] M. Ratto and R. Ree, "Materializing information: 3D printing and social change," *First Monday,* vol. 17, no. 7, 2012.

[6] T. Cover and P. Hart, "Nearest neighbor pattern classification," *Information Theory, IEEE Transactions on,* vol. 13, no. 1, pp. 21-27, 1967.

[7] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *Journal of the ACM,* vol. 45, no. 6, pp. 891-923, 1998.

[8] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters,* vol. 9, no. 3, pp. 293-300, 1999.

[9] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, 2001.

[10] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, no. 11, pp. 2278-2324, 1998.

[11] Y. Bengio, "Learning deep architectures for AI," *Foundations and trends in Machine Learning,* vol. 2, no. 1, pp. 1-127, 2009.

[12] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005.

[13] T. Ojala, M. Pietikainen and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 24, no. 7, pp. 971-987, 2002.

[14] P. F. Felzenszwalb, R. B. Girshick, D. McAllester and D. Ramanan, "Object detection with discriminatively trained part-based models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 32, no. 9, pp. 1627-1645, 2010.

[15] Z. Zhang, R. Deriche, O. Faugeras and Q.-T. Luong, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," *Artificial intelligence,* vol. 78, no. 1, pp. 87-119, 1995.

[16] O. Chum, J. Matas and J. Kittler, Pattern Recognition, Berlin: Springer Berlin Heidelberg, 2003, pp. 236-243.

[17] K. Khoshelham and S. O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors,* vol. 12, no. 2, pp. 1437-1454, 2012.

[18] DARPA Defense Advanced Research Projects Agency, "What is the DARPA robotics challenge," DARPA, June 2013. [Online]. Available: http://archive.darpa.mil/roboticschallengetrialsarchive/about/index.html. [Accessed 14 April 2014].

[19] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," *IJCAI,* vol. 14, no. 2, pp. 1137-1145, 1995.