Steve Babcock

4/29/2017

GunVision

View this project in action at https://www.youtube.com/watch?v=0uUA03G5q60

To build yourself - verified on Mac OS X

1) Plug in a USB webcam - I used a Logitech HD C270
2) Ensure OpenCV and OpenGL libraries are installed on your machine
3) Open a terminal
4) cd to Code folder
5) enter the command found in build.txt (may need to alter depending on specifics of step 2)

This project aims to create a smart security camera capable of detecting a raised firearm using computer vision techniques such as background subtraction, template matching, and neural networks. It could be used in convenience stores or banks to send an early alert to police when a robbery is in progress. I have a sizable code base built using C++, OpenCV, and OpenGL. The project has been ran on Mac OS X, but should be very portable to other operating systems. Many assumptions about the problem have been made, and the solving philosophy I choose strives to perform well within these bounds. However, it is not concerned about performance outside the assumptions. These include a fixed location of the camera, consistent light source, few windows/doors in the background, the objects moving about the scene are humans, a buffer of 3-4 feet to the camera is generally maintained, the left/right position of a cashier relative to the scene is known, the camera is centered upon the position where a robber is likely to stand, this view is perpendicular to the line between the cashier and robber, and this view is close to parallel with the ground. Obviously many physical locations such as outdoors

and tight quarters will not be viable for deployment of our project. Though I believe many of the most important locations that our project could benefit do fit these assumptions.

The most similar of the research papers studied is Automated Detection of Firearms and Knives in a CCTV Image written by Grega, Matiolanski, Guzik, and Leszczuk. Their algorithm is significantly longer: background detection, Canny edge detection, sliding window, scaling, PCA, neural network, candidate regions, MPEG-7 classifier, spa/temp filtering, and decision. Similarities exist between our two approaches. Both begin with background subtraction, and use a sliding window technique later. However, they proceed to detect human bodies, and then search around the body for a weapon. They also have a much broader definition of the problem - attempting to detect lowered handguns and non-perpendicular views. Their results show a sensitivity of 35.98% and specificity of 100%. Our results will show a higher sensitivity and lower specificity. The higher result is likely in regard to our narrower definition of the problem. Another difference is they only noted using an umbrella and bag as random objects. While I used several objects with similar dimensions to that of a gun barrel, and held in a way to mimic a handgun. Overall, it is difficult to compare the two results due to these differences.

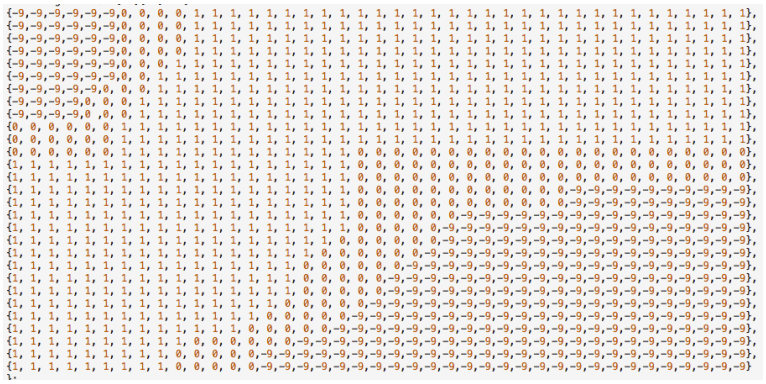
The detector is essentially a chain of sub-algorithms where positive results from one sets off the next step, and negative results break the flow. First I anchor the camera, then generate a background binary map, next search for a raised arm, and finally scan the end of the arm for a gun. The anchoring is performed by selecting eight pixels spread around the edges of the scene. If less than three pixels do not match the values of the previous frame, the camera is said to be unanchored, and this frame will not be processed further. Enough consecutive unanchored frames can result in the deletion of a background image if one has been generated.

A background image holds values I assume to be the unmoving parts of a scene. In our code, it is a matrix of RGB values just like a normal image. Except one extra property is added to each pixel - a certainty value which is initialized to zero. As more video frames can in, each new pixel is compared to its corresponding background pixel. If they are similar, that background pixel's certainty is increased - decreased otherwise. This similarity has an inverse relationship to the maximum difference between the three color channels of the pixels. When the net certainty of a background rises above a threshold, it is said to be generated, and can be used to produce binary maps of a scene. Given a new video frame, any pixel similar to its corresponding background pixel is assumed to be part of the background and re-assigned to a zero. Otherwise, the pixel is assumed to be an active part of the scene and re-assigned to a one. Also if the certainty of a background pixel falls below a threshold, its value is replaced by the next incoming frame and its certainty is reset to zero. This allows for adaptation to changes in lighting, camera position, and new permanent additions to a scene.

Next, I use dynamic programming to rapidly convolve an arm template through the binary map searching for a raised arm. Below is a rough visual representation of this template where white is positive (1), gray is neutral (0), and black is negative (-1).

Each column of the template is uniform. So moving it over simply requires the next column to be added, and the oldest column to be subtracted from a running total. The assumption made here is that a robber is unlikely a hold a gun at his or her hip.

Finally, I use a series of gun templates to scan the end of a detected arm for a weapon. This process is divided into four parts: matching the whole template, score of the barrel section of the template, color channel balance, and smart template matching. Below is an example of a gun template which is a matrix of 1's where a gun or hand should be, 0's forming a buffer around the 1's, and -9's in the areas where nothing should be seen by the background subtractor.



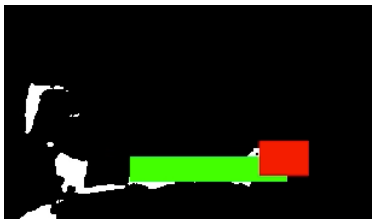
At the place where this template scores the highest (and is above a threshold), I consider the score of the right side or gun barrel. This is done because the left side will match highly against a hand holding any object. The barrel is the most identifying part of a gun, but if I were to convolve a barrel template by itself, unexpected results often happen. That is, it may match well against the top/bottom of an arm or random noise. Also since the position of the cashier is known in our assumptions, the program is started with no arguments when the gun is expected to be aimed stage left of the camera. It is started with any argument otherwise, and the input will be mirrored horizontally before passed to any detection algorithm.

Next I analyze the values of the three color channels in the original video frame at the locations where the gun barrel is thought to exist. If there is a significant difference in these values, I ignore the frame in regard to containing a gun. Our assumption here is that a robber is

unlikely to possess a blue gun or a red gun, etc. I are only interested in detecting black, gray, or silver weapons.

Finally, the detector has a training mode that will save to a file the contents of the binary map in the form of a vector at the location a gun has been detected. After generating many of these recorded at different distances (and potentially with different gun models), I have much more exact ideas of what a gun looks like to our background subtractor at its current physical location. The entire collection can then be compared future recognitions of a gun by rearranging the image segment into a vector, and calculating minimum euclidian distance. If it is below a threshold, I conclude there is a raised handgun in the scene.

We performed two rounds of testing. An initial round was in Steve's room at a distance of 8-9 feet with five rounds and five objects. About a week later, the final round occurred in the Mac lab of Lindley Hall's basement. It used five objects and ten rounds at distances of 4-5 feet, 6-7 feet, 8-10 feet, 12-15 feet, and 18+ feet. Our program draws a green rectangle when detecting an arm, and a smaller red rectangle upon detecting a gun. Reviewing the testing video, and counting the presence of these rectangles determines the statistics. For example…



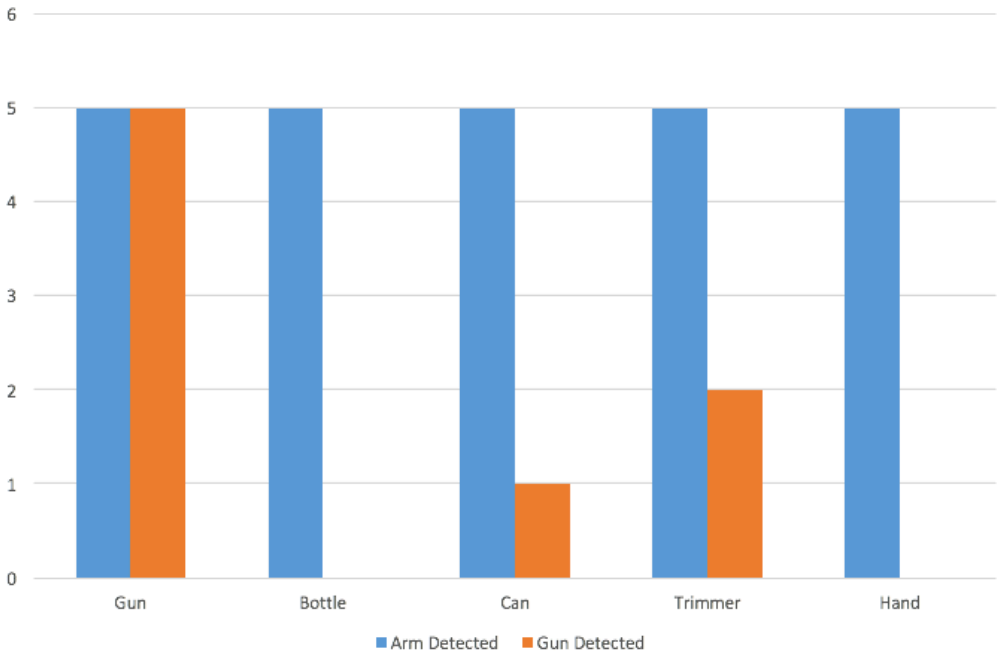
Initial Testing

Objects Used

Left to Right: Hand, Can, Gun, Trimmer, Bottle

Location: Steve's Room

Results - Out of Five Rounds



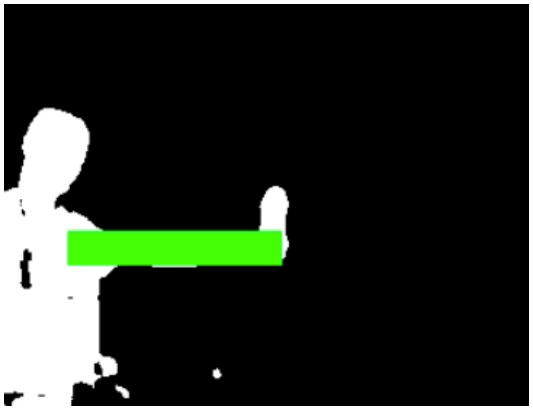
Final Testing

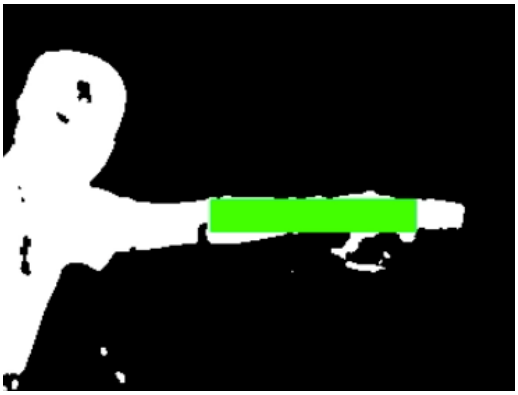
Objects Used

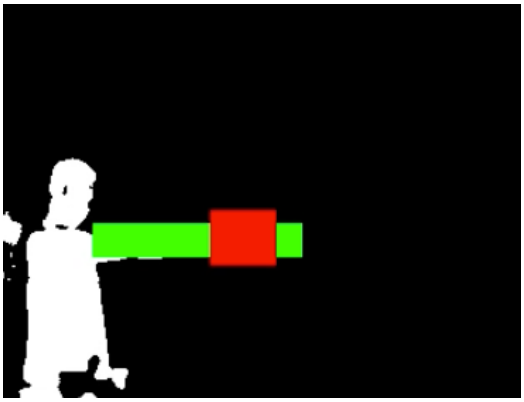
Left to Right: Shampoo, Toothpaste, Gun, Razor, Hand

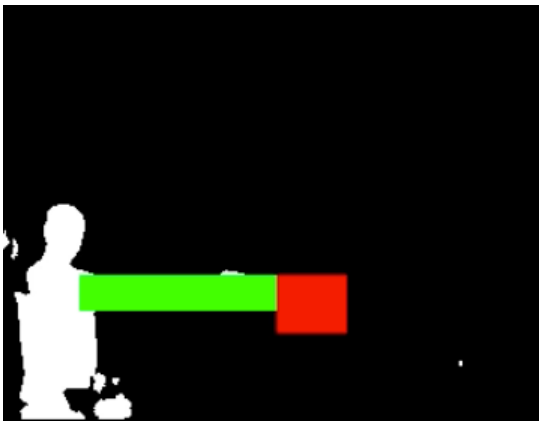
Location - Lindley Hall Mac Lab

Testing Images



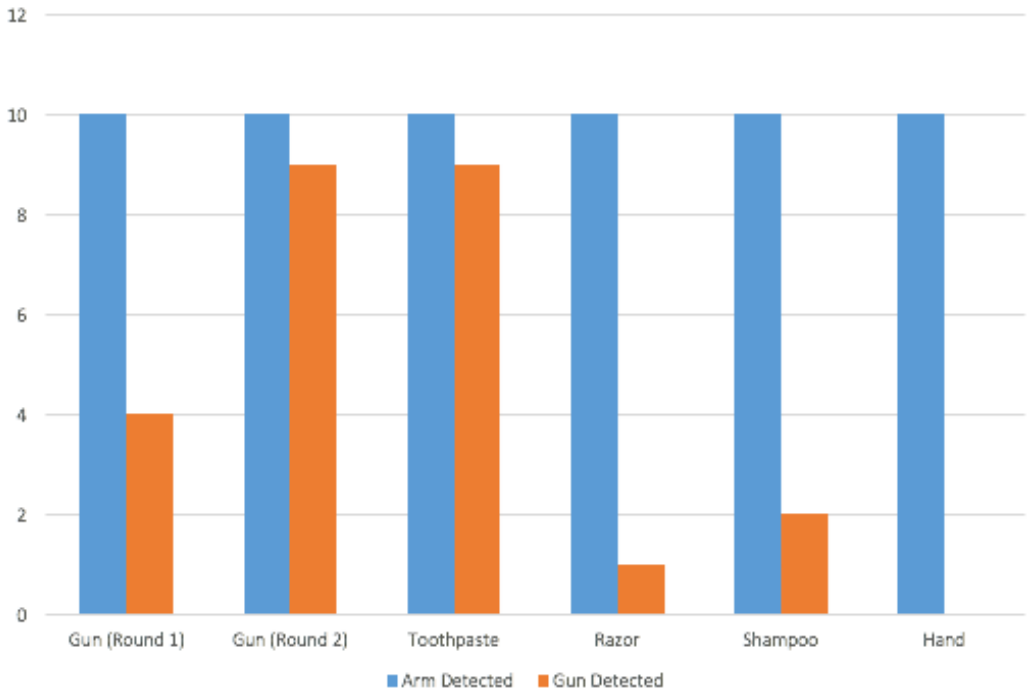




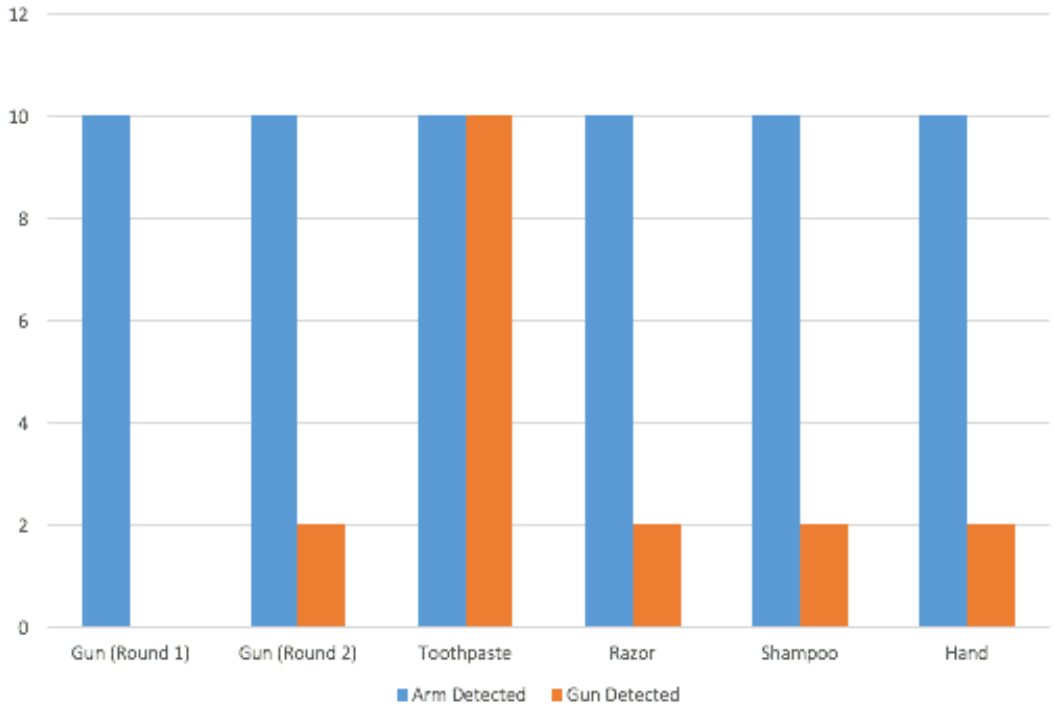


All Final Results Are Out of Ten Attempts

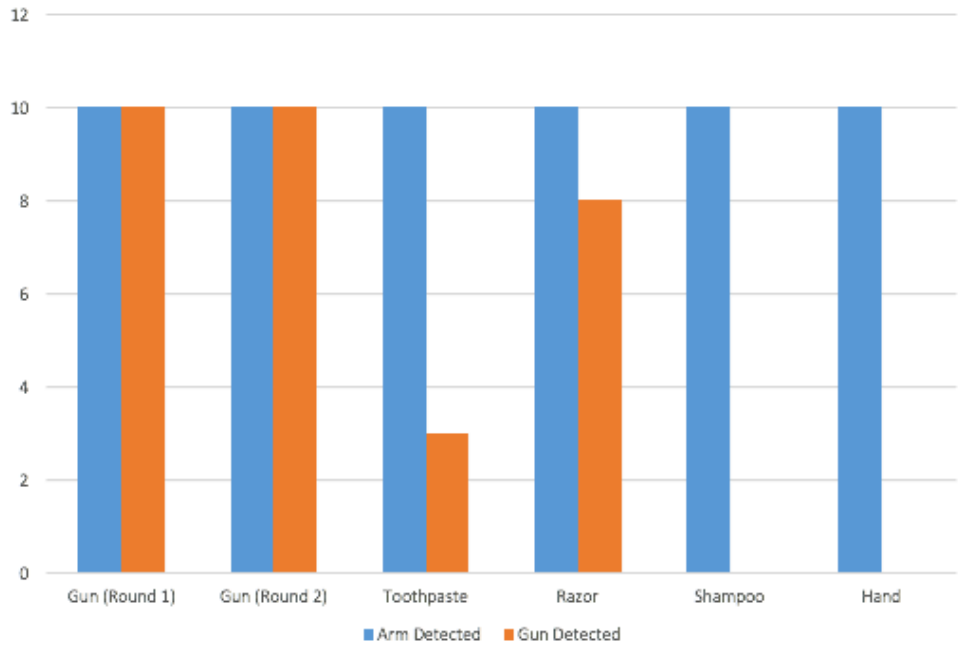
Results: 4-5 feet



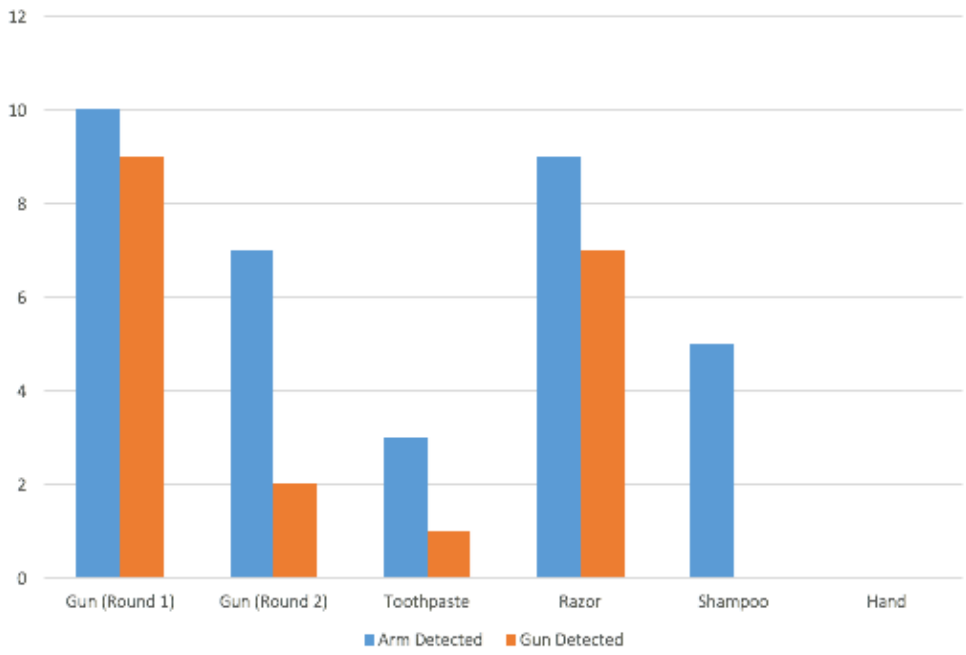
Results: 6-7 feet



Results: 8-10 feet



Results: 12-15 feet



Final Sensitivity: 4-15 feet

Raised Arm: 214/240
89%

Handgun: 46/80
57.5%

Final Specificity: 4-15 feet

Other Objects: 113/160

70%

Download Testing Videos

Initial Testing https://drive.google.com/open?id=0B69-xfrx3aQmSFpNcER0QWRoMUk

Final - Gun Round 2 https://drive.google.com/open?id=0B69-xfrx3aQmZWxOSm5KYmN2OXM

Final - Gun Round 1 https://drive.google.com/open?id=0B69-xfrx3aQmVjZuT0FMWjZZTVk

Final - Hand https://drive.google.com/open?id=0B69-xfrx3aQmR2ZtWHJndGVRanc

Final - Toothpaste https://drive.google.com/open?id=0B69-xfrx3aQmNFI2YXVucTIxS1k

Final - Razor https://drive.google.com/open?id=0B69-xfrx3aQmeUVkOGY1cVNDaVE

Final - Shampoo https://drive.google.com/open?id=0B69-xfrx3aQmbnVSeU5fS0MzSVk

The initial test results were very promising, and I believe this occurred because the location coincided with the place most of the code was written. Steve's room also has a rather confined space, so varying distances were hard to simulate. After the initial tests, the templates were further refined, and thresholds for gun detection increased. Expectations for final testing were high. However, moving to the lab in Lindley Hall served as a rude awakening in some regards. Initially, virtually no guns were being detected at all. After dropping the thresholds to identify a gun, false positives became numerous and even overtook real detections at some distances. In fairness, the test subject did hold the razor, toothpaste, trimmer, and hand in a way to mimic a gun as best as possible.

None of the smart training filters generated in Steve's room seemed very effective, and were regenerated. I believe this is because the camera was mounted generally lower than the levels the gun was raised to for initial tests, and mounted equal or higher for final tests. The background also varied in color greatly between the two locations. Steve's room provided a mostly brown background, but the Mac lab had stark white walls and computer screens that

were very dark. The gun would often blend in to the background when held in front of a screen. Also, there were several windows and even a few people studying in the background during final testing. One benefit of the final location was it had dozens of light sources. This allowed for much clearer binary maps at the 6-7 and 4-5 foot ranges. In Steve's room, there is only one light source, and standing that close to the camera often creates noisy images by interrupting it.

The final results showed some flaws in the gun template matching strategy. Three templates were used: far, near, and regular. The regular template was optimized for 8-10 feet. The near template turned out to work best at the 4/5 foot range. In between these distances, the gun detection hit a dead spot. Only 2/20 guns were detected at 6-7 feet. Templates performed inconsistently at the 12-15 foot distance for both arm and gun. This proved to be the boundary at which arm detection functioned. An arm must be fully outstretched and holding an object in a way that extends its silhouette for the detector to register at this distance. Any further back, arm detection doesn't work at all with the current setup. These distances also provide much less detailed silhouettes of the gun. Nevertheless, gun detection can be accurate around 12-15 feet as demonstrated in the first round of final testing. I assume the test subject was likely standing closer to 12 feet this round. Towards 15 feet, the accuracy of both arm and gun detection plummeted. Side note - six other templates were tried where the gun was held slightly up and down at the fear, near, and regular distances. However, these seemed to only increase false positive rates, and were not included in the official testing.

Upon conclusion, background subtraction and template matching can provide a good idea of where a gun is most likely to be in a scene which fits our assumptions. However, the final testing showed these techniques alone cannot accurately determine whether there truly

exists a gun at this specific location in an image. It may seem possible to obtain good results if enough time is spent configuring parameters to an exact camera height, distance, and background. Though these tailored parameters become of little use when placed in a new environment. I believe this work could be useful in providing input to a more accurate gun detection algorithm. Another possibility may be to replace the smart template generated from background subtraction with templates generated from a Sobel filter/edge detection. Perhaps SIFT could be involved in some regard, and still function in real time. I believe the idea of going from background subtraction to detecting a raised arm is novel, and could benefit others working on this problem. This strategy is very computationally efficient, makes sense, and helps reduce false positives. Also, a narrow problem definition and high level of assumptions are different from most others' approach. Some may scoff at this philosophy, but I see it as still being very applicable to real world situations.

In general, I believe the idea of a smart security camera could very valuable to our modern society. Currently, a verbal description of an attacker is typically given to emergency responders/police. This can lead to false arrests and unneeded confrontations between law enforcement and civilians who happen to fit the given description. Our society currently has all the technology to make verbal descriptions of someone robbing a bank or convenience store obsolete. However, I simply are not using it in that regard. Hopefully, this project can inspire more computer scientists, community leaders, and law enforcement to believe emergency response can be guided by computer vision.

References

Grega, Michal, Matiolanski, Andrzej, Guzik, Piotr and Leszczuk Mikolaj. Automated Detection of Firearms and Knives in a CCTV Image. Multidisciplinary Digital Publishing Institute - Sensors. Published 1 January 2016.

You Only Look Once: Unified, Real-Time Object Detection Joseph Redmon , Santosh Divvala, Ross Girshick , Ali Farhadi University of Washington , Allen Institute for AI , Facebook AI Research (CVPR Paper)

Dalal, Navneet and Triggs, Bill. Histograms of Oriented Gradients for Human Detection. Computer Vision and Pattern Recognition. Published 20 June 2005.

Grega, M., Lach, S., Sieradzki, R. Automated recognition of firearms in surveillance video. In Proceedings of the 2013 IEEE International Multi- Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), San Diego, CA, USA, 25–28 February 2013; pp. 45–50.

Chen, C.; Zhuang, Y.; Xiao, J. Silhouette representation and matching for 3D pose discrimination—A comparative study. Image Vis. Comput. 2010, 28, 654–667.

Acknowledgements

Thanks to Dr. David Crandall who taught CSCI B657: Computer Vision at IU Spring 2017.

Thanks to Gurleen Dhody who was my partner on this project. He trained a neural network to detect handguns in still images, and we may combine our efforts at some point in the future.

All contents of this report and C++ code was written by myself - Steve Babcock