Autonomous Mobile Vision System For Terrorist Attack Prevention

Author: Sam Saltwick

Abstract

This research involves devising an alternative method of monitoring public areas. The goal is to demonstrate that a mobile platform controlled by a vision-based navigation system can be effective in detecting terrorist attacks in public areas. This has been done by researching common terrorist attack methods, constructing and testing a highly mobile spider-like platform, and building software for detecting objects commonly used as improvised explosive devices. Upon combining these aspects, it became apparent that this type of system is viable for detecting objects in public areas. Following this conclusion, placing a similar system in a public environment, such as a train station or airport, has the possibility to drastically reduce the impacts of terrorist attacks.

Introduction

Reports about terrorism and terrorist activity permeate the news cycle, and, since 9/11, counter-terrorism strategies have developed at a brisk pace in order to keep one step ahead of terror plots. Authorities, while performing their utmost duty to protect their citizenry, also warn us to be ever-vigilant, or, in the case of the New York Metropolitan Transit Authority (MTA), to "See Something, Say Something" (MTA New York City Transit). Currently, stationary cameras are a security measure used to supplement human vigilance, but are not monitored continuously and are consequently used reactively, only in response to events (Chan). In the modern technological era in which we live, many have realized that these methods inadequately perform the job they are meant to do and that a more proactive approach, i.e. identifying and monitoring likely targets before a threat even poses itself, would protect more lives.

Typical High Risk Areas, Behaviors, and Methodologies

Railroad Stations	Stairwells
Exits	Improvised Explosive Devices
Excessive Studying of Maps	Remaining on Platform for Many Trains

Sources: MTA New York City Transit & DHS and FBI

One high risk area for potential terrorist activity are railway stations (Larcher et al.). As evidenced by the Madrid and Mumbai railway bombings, which killed or injured over a combined 1600 people, a covertly planned attack with an Improvised Explosive Device (IED) has the ability to destroy without a terrorist actor having to be physically present to partake in the destruction (DHS and FBI 33-34), and, according to New York Police Department (NYPD) Transit Bureau Deputy Chief Vinnie DeMarino, these terrorists generally formulate their plot and scope their target weeks and months before they commit the act of terror (MTA New York City Transit). One positive that can be drawn from this, however, is that, according to the DHS and FBI, if the attack is committed remotely, the terrorist is necessarily limited in where he or she can place an IED to both ensure minimal conspicuity and maximal damage (33-34)

With knowledge of these likely areas and methods of attack, some have studied the possibility of using object recognition to determine the suspiciousness of an object and/or person. Becker et al. studied the use and usability of Interacting Multiple Model (IMM) in detecting certain human actions that many terrorists share, like, for example, remaining in one location looking at a computer or studying diagrams, as described by Chief DeMarino. Moreover, Beynon et al. studied a similar process instead in regards to abandoned packages, specifically the automatic detection of them by camera networks. Nevertheless, these studies and methods rely on static camera networks, usually the same ones described above.

We propose that a mobile camera system, mounted on automated robots, would further improve upon these counter-terrorist measures. In locations with heavy pedestrian traffic, the maneuverability of the platform can provide better coverage than a static network because it would be able to conduct real time adapting to new situations.

Many challenges come along with robotic mission planning, including programming their maneuvering patterns and object recognition algorithms. According to Alterovitz, Koenig, and Likhachev, “robot planning is often necessary for navigating through an environment, manipulating tools and objects, maintaining safety around humans, and gathering information necessary to complete a task” (2). Here, the three aforementioned researchers say that robots need to be able to adapt to its situations all while maintaining safety for the humans around it, which will be paramount for our counter-terrorism bot. Moreover, they claim that “the planning algorithms will need to enable a robot to operate in a team with other robots and/or humans, to

integrate perception with planning, to compute and execute motions in real-time, and to request help from humans when necessary" (5). Here, they explain potential challenges that our robot would have to overcome in protecting the populus, and certain solutions we can explore. One critical element we will explore is how to change directions quickly and precisely with minimal extra movements to avoid people in high volume areas.

Being able to change directions quickly with a wheeled system can be challenging, as seen in studies with robotic wheelchairs (Prassler), but legged systems are much more adept at making quick direction changes because they avoid having to make several back and forth movements (Armada). Considering this point, a six-legged drive system will become the base for the image recognition system.

Several studies have been done on the efficiency (Pratihar) and mobility (Armada) of multi legged robots over rough terrain as compared to wheeled robots but this study is more concerned on the efficiency of legged systems on flat terrain, such as that found in public areas. Work on the theoretical force and torque distributions on an eight-legged spider-bot on horizontal, vertical, and inverted surfaces has also been studied (Gasparetto) but without a specific goal in mind or any physical implementation of the simulations.

In this paper, we explore the use of a camera equipped legged robot to actively sweep train stations and other high risk areas, all while using object recognition to detect suspicious objects and behavior. Not only will our system identify them, but it will actively look for them using the information about common locations and objects that would make up an IED or an IED wielder. To do so, we:

- Analyzed various approaches to coverage and surveillance on the map of New York Penn Station
- Evaluated different methods of mobility using kinematics.

And our contributions include:

- Various datasets for navigating a subway system/train station
- A mission algorithm that relies on previously known facts about likely zones of interest
- Notions of legged-robot movement and analysis of the robot's step.
- And utilization of machine learning object recognition on a small scale.

Methods

Mission Simulation Environment

System guidance strategies were developed through a computer simulation of varied deployment scenarios. The simulation was run on a map of New York Penn Station, with a collection of planned routes for the simulated robot to follow. Two possible routes were examined: one that focuses on visiting all points of interest for short periods of time (Figure I), and one that visits only the higher probability areas, for an extended period (Figure 2).

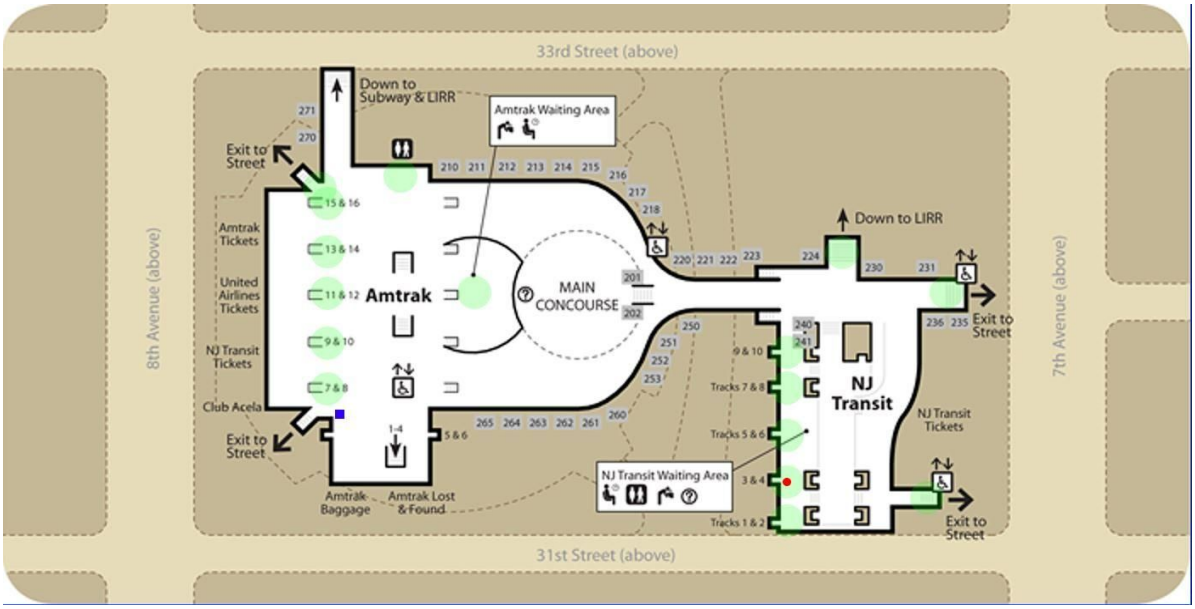

Figure I. Simulation Modeling Total-Coverage Algorithm

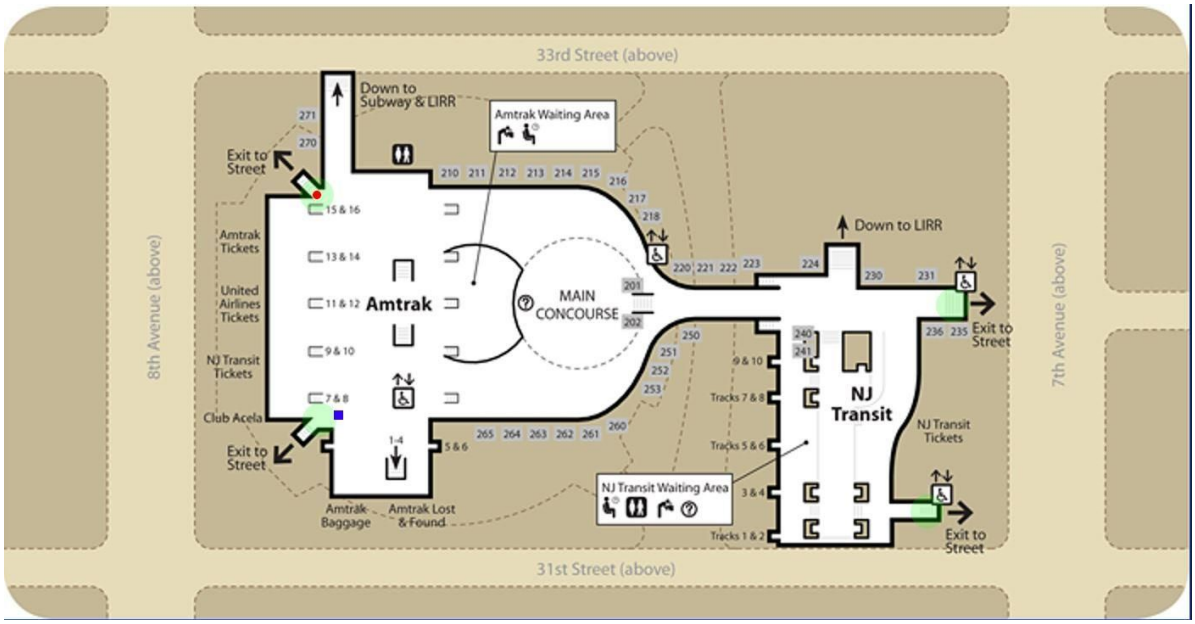

Figure II. – Algorithm Modeling High-Priority Only Waypoints

The waypoints in the simulation are depicted as the green regions on the map, the simulated suspicious object is shown as a small red circle, and the robot is shown as a blue square. The time that the robot spends at each waypoint is determined by a multiplicative factor determined from the researched threat-chance of each area and a random length of time, simulating the robot scanning an area. The placement of the simulated explosive device is also determined by our research. The simulation chooses a pseudo-random waypoint for the target to be placed at, influenced by the risk-factor of each waypoint. This causes higher priority waypoints, such as the exits, to have a higher chance of containing the target. The route focused

on high-threat-chance positions focuses on the exits of the ground floor of the train station, which have been determined to be the most likely areas for a bomb threat.

The path that covers many waypoints has runs past exits, bathrooms, entrances to other terminals, and the main waiting areas. These areas all have large amounts of traffic flow in tight spaces, and spaces for the conspicuous placement of devices. The path that covers many waypoints better models a system that could act more effectively than static CCTV cameras. With the robot traveling to multiple locations, in corridors, and in crowded areas, IEDs and other suspicious objects could be discovered more autonomously than even a camera running the same detection algorithm (DHS IED Attack).

Determining the risk of probable areas allows for the robot to take an optimized path around the designated area. Given locations to investigate, the system can prioritize the most dangerous locations and follow a route based on this. The simulation provides data on the time taken to locate the explosive device based on which guidance algorithm was chosen. Choosing between these two algorithms narrows the focus of the system, and allows for a higher level of efficiency. Being able to only traverse through the highest priority areas allows the robot to make more cycles of the station, creating a safer environment that would be harder for a terrorist to penetrate. This data could then be used to build a navigation algorithm for the autonomous system to traverse an area consisting of pre-determined probable-threat areas.

IED Detection Demonstration Software

For the robot to be able to detect suspicious objects, it needs a method of detecting and classifying what is in front of it. Initially, our system utilized only the OpenCV library, a collection of image processing functions. Two methods were testing with this library: template matching and feature matching. Both methods used a supplied image to recognize objects, limiting the system's abilities to being able to detect one object at a time. Template matching looks for the closest exact copy of the supplied image in the camera feed, which works in near-perfect situations. Rotations, distance, and lighting drastically decrease the ability of a template-matcher to recognize their trained objects.

The second method used feature matching, which populates a data structure with prominent features of a supplied image, primarily edges. A likelihood-threshold is used to limit the number of excess features that are detected, and a specified number of matches is required to decide that the object is in sight. As seen in Figure III, this method worked better for the detecting in less perfect conditions, but did not work when the object is viewed from an angle not specified in the given image.

Figure III – Feature Matching Prototype System

These two methods led to searching for a method that would be unaffected by changes in conditions like spatial transformations. The final method for our system's ability to 'see' is powered by Google's Tensorflow Object Detection API. This API contains a pre-trained model of over 100 common objects, including people, suitcases, and backpacks. Using this, the software can detect and navigate to suspicious objects while traversing the designated waypoints. If needed, the API includes the ability to train more objects into the model, so the system can detect any object. Using Google's pre-trained model is significantly faster and more reliable than a personally trained model, due to time and computational limits. Compared to the previous two methods, the Tensorflow model can recognize several objects at once, in less time, and in many conditions. Using the pre-trained model, the algorithm can calculate the probability of the model being correct, giving a quantified measure of reliability for the object detection.

The object detector can be used to locate designated objects, such as suitcases or backpacks, relative to the camera on the robot. Using the object's coordinates relative to the camera's field of view, the robot can maneuver to maintain line-of-sight on the object, adjusting to keep the target centered. Doing so will allow the system to navigate to located objects for further inspection and determination of their suspicion.

Multi-Legged Platform Kinematic Analysis

The preliminary design for the spider-bot has been created and will be made up of six legs that move in groups of three. This mechanism will ensure that three legs are always on the ground to support the robot while the other three move forward. The two sets have different designs so that they can move in opposing vertical paths (as one set moves up, the other set must move down) while using the same driving mechanism. This means two simulations for the legs needed to be created and integrated in order to have a full picture of the movement of the robot. Both simulations have been created. One contains six variables that can be altered in order to determine the optimal configuration. The other is more complicated and includes nine variables. The simulations plot common points between the paths that different parts of the leg move in, then draws the actual leg movement based on these and plots where the bottom of the leg will be at several points in its movement. These simulations will need to be combined to represent the actual movement of the robot. If only one leg simulation was used or the simulations were run separately, the transition from one set of legs to the other would not be accurately portrayed. The final simulation will need to take into account how the transition affects the path of the robot in order to determine the configuration most appropriate to the task of supporting the object recognition system.

A physical model of the rover has also been created but not tested. This design can be used in later tests to determine the differences between the simulation and physical implementation.

Results

Mission Simulation Findings

Our simulation provides results in terms of the system's efficiency in locating the target. The simulation records which path was chosen and measures the time taken to locate the randomly placed target. In our tests, Path 1 is the route where the robot visits more areas of interest for a shorter period of time, while Path 2 visits only the highest priority areas (exits), but for a longer period of time. Both of the graphs below modelling the algorithms' time are ordered from the least to greatest time taken by each algorithm. Below, in Figure I, is the results from the simulation being run 50 times. The simulation chose Path 1 twenty-two times and Path 2 twenty-eight times. The average time that Path 1 took was 21.58873 seconds, while Path 2 took on average 17.09286 seconds. While these results do correlate with Path 2 having less waypoints, on average each waypoint took longer to visit while navigating Path 2.

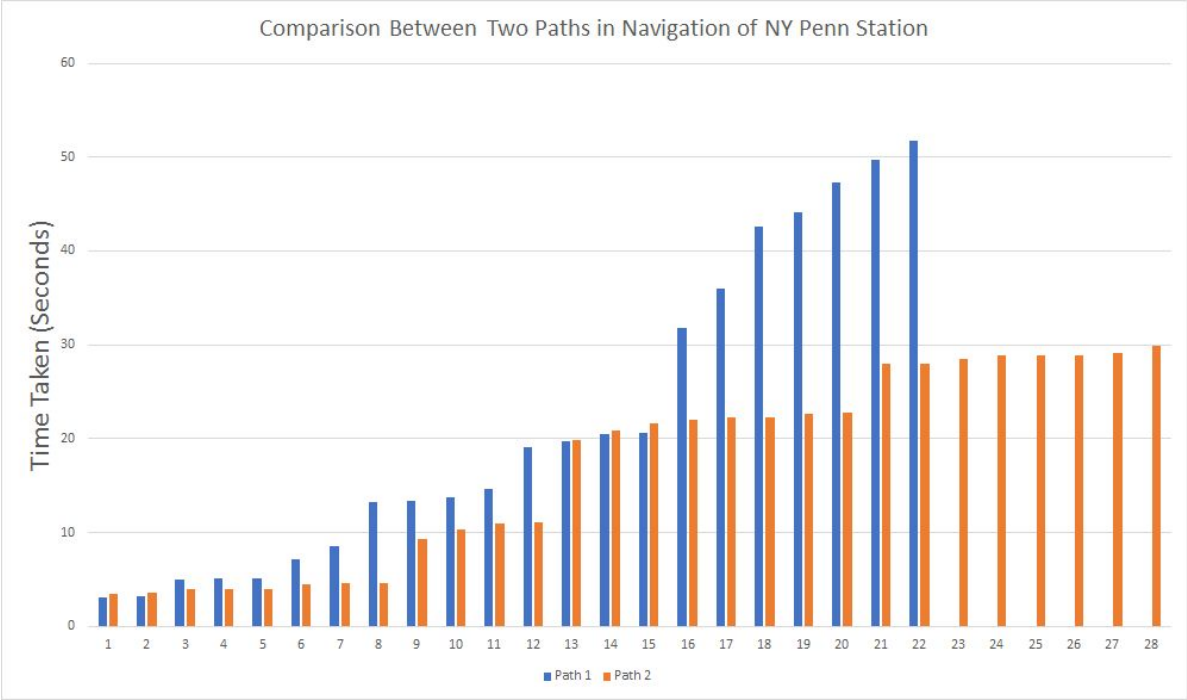

Figure IV. - Data from 50 Mission Simulations

As the number of runs for the simulation increases, the distance between the average times appears to increase. Figure II below depicts the results from 100 iterations of the simulation. The average time taken when Path 1 was chosen for 100 runs was 24.13163 seconds and the average time taken for Path 2 was 16.88863 seconds. For 100 simulation runs, the average difference in time taken between the algorithms is 6.767217 seconds. As seen in both figures, as the total time taken by each algorithm increases, the time difference between them significantly increases. The minimum times taken for both algorithms are very similar, since these are when the bomb is located close to the robot's starting point. The maximum times taken for both algorithms are much further apart. This trend leads us to believe that the guidance algorithm focused on only the highest risk areas will, on average, find the target in the shortest amount of time.

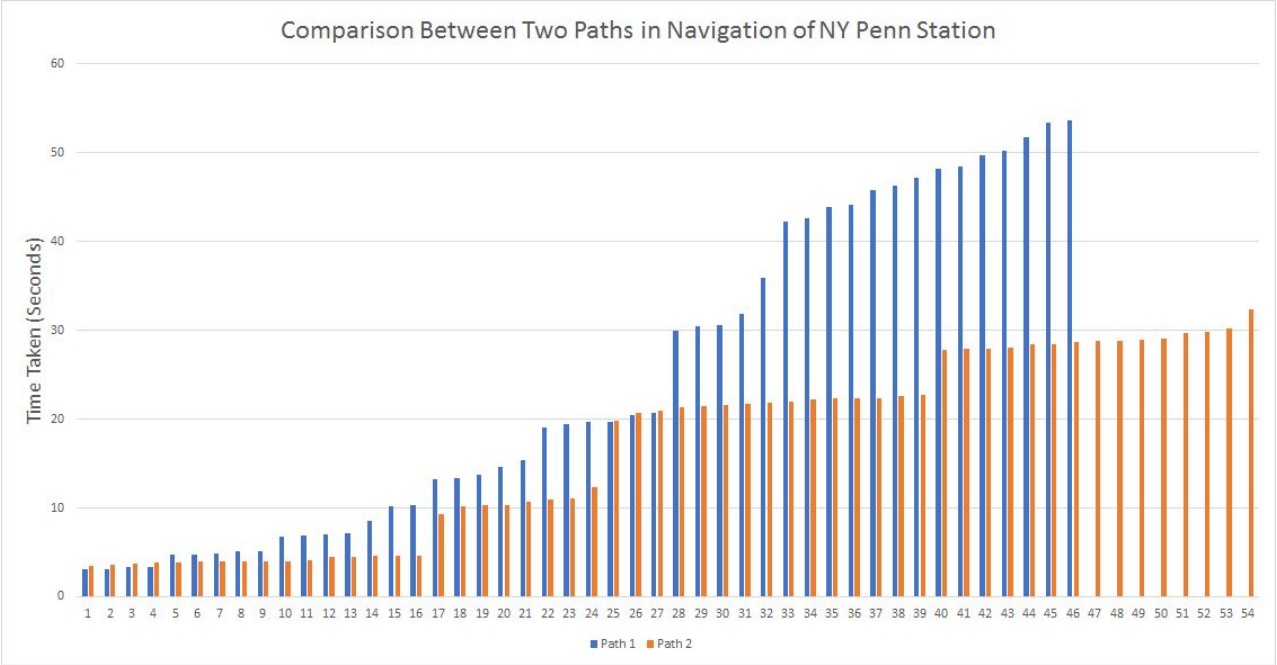

Figure V. - Data from 100 Mission Simulations

Suspicious Object Detection Software Testing

Using Google's Tensorflow Object Detection API, we tested the effective range of the software using a standard integrated web camera. From multiple tests varying the distance and lighting conditions, the software was able to recognize a standard backpack from a range of seven feet. The limiting factor of the software's ability to detect the backpack was the detail that the camera could pick up at the further distances. This could be increased with either superior lighting conditions or a better camera. Figure III below shows the software correctly detecting a backpack at the integrated camera's maximum range.

Figure VI. - Object Detection at maximum range

The software is able to detect larger entities at a further range, such as people and suitcases. These have larger, more distinct details that even the integrated web camera can pick up at a larger range.

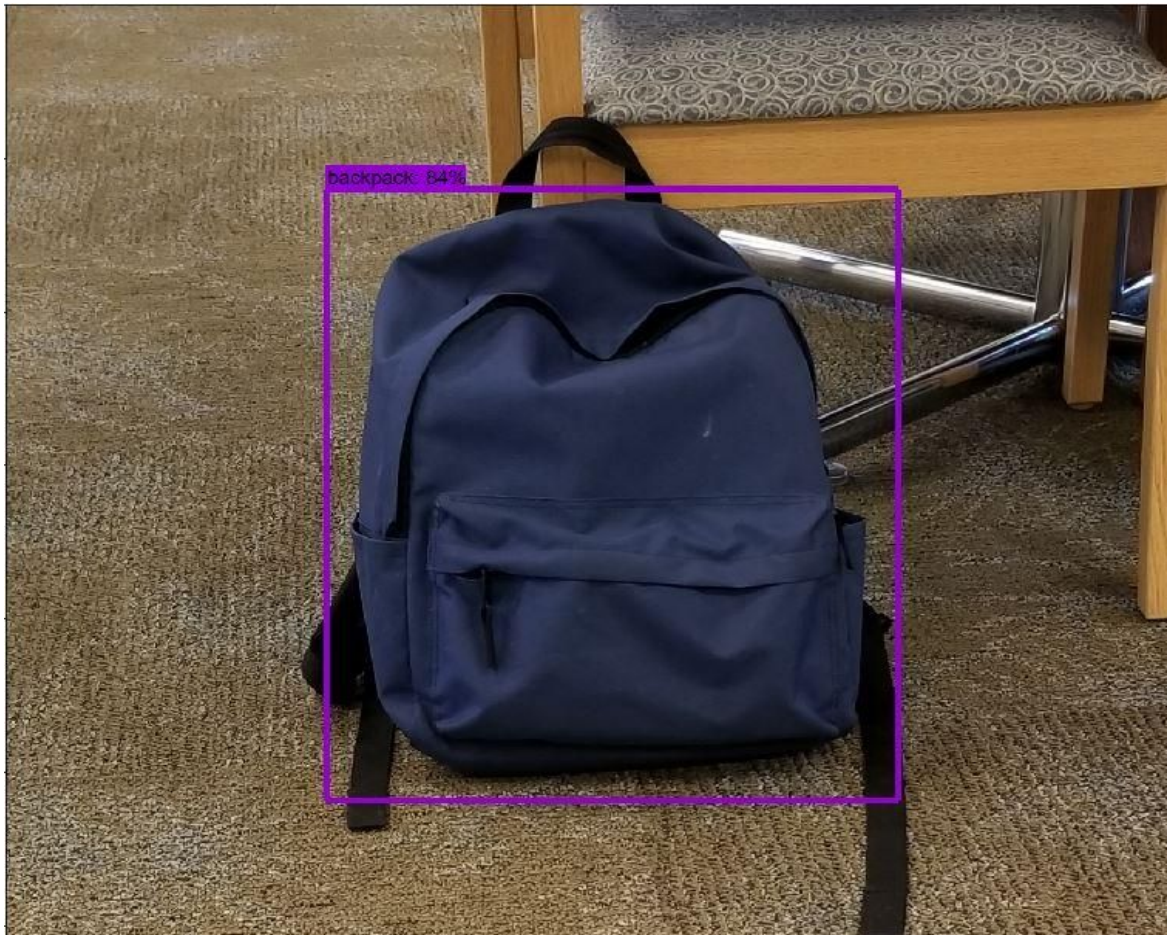

Figure VII. - Object Detection at close range

At closer ranges, the software is able to more reliably recognize objects. Figure IV shows the program recognizing the same backpack as Figure III, but at approximately half of the range. The program is able to recognize the backpack as such quicker, and with a higher degree of confidence.

The largest shortcoming of the software is its performance on standard computers. Running the program on an Intel i5 processor in a laptop yields under 5 frames per second using the built-in webcam. Using Tensorflow for the GPU and an NVIDIA GTX 940m, the fps increases significantly, to around 20 frames per second. An even more powerful GPU would result in a smoother live-video experience, but is not necessary for this application. To increase performance, a higher quality camera could be used with a significantly lower framerate,

Multi-Legged Platform Optimization

Presently, simulations for both leg shapes have been created and several of the possible variables have been tested to find patterns in the movement. Before discussing how the variables affected the movement of the rover, it is helpful to see what part of the legs each variable refers to, this is presented in Figure VIII.

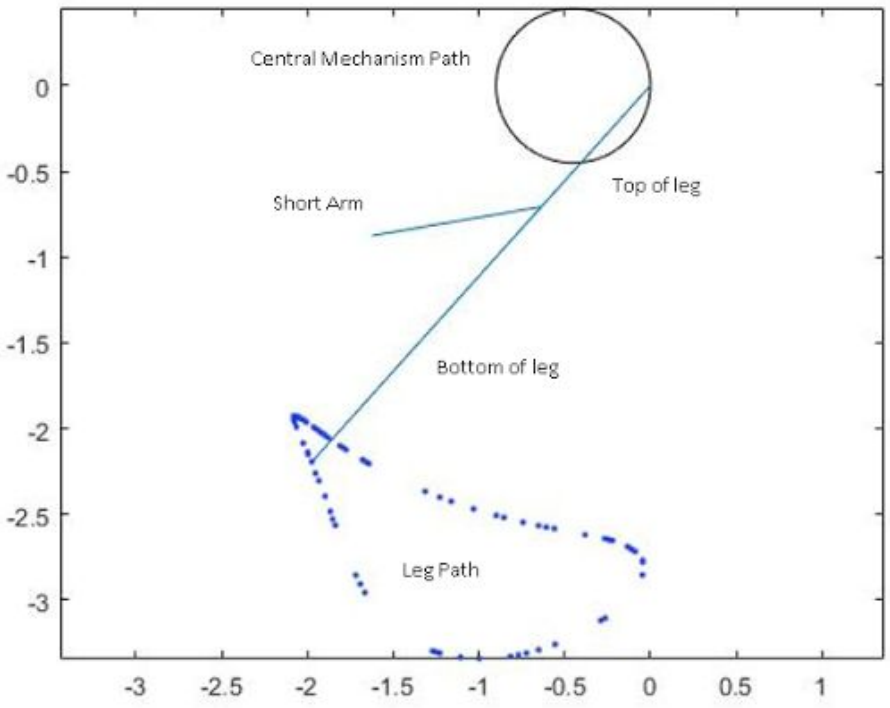

Figure VIII. - Leg 1 Diagram

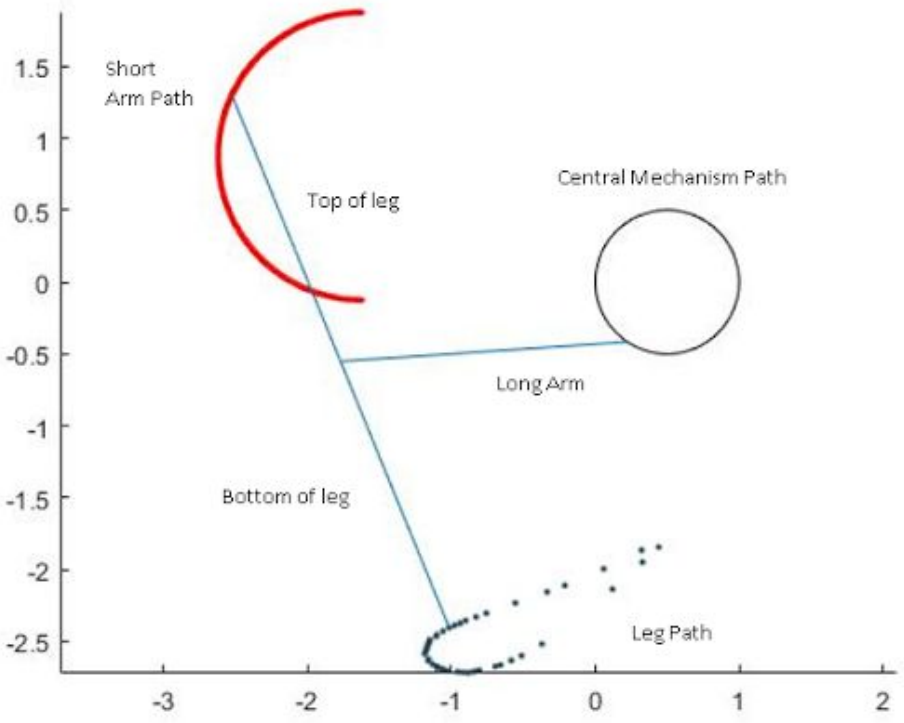

Figure IX. - Leg 2 Diagram

With this geometry of the two legs in mind, we can now discuss the variables that the simulation takes into account. The simulation for the first leg considers: the length of the short arm, the offset (x and y) of the short arm attachment point with respect to the central mechanism, the radius of the central mechanism path, the length of the top of the leg, and the length of the bottom of the leg. The simulation for the second leg considers all of these variables as well as the length of the long arm and the location (x and y) of a point that the long arm must pass through. Now we will examine how changes to these variables affect the x and y variation of the "step" of the robot.

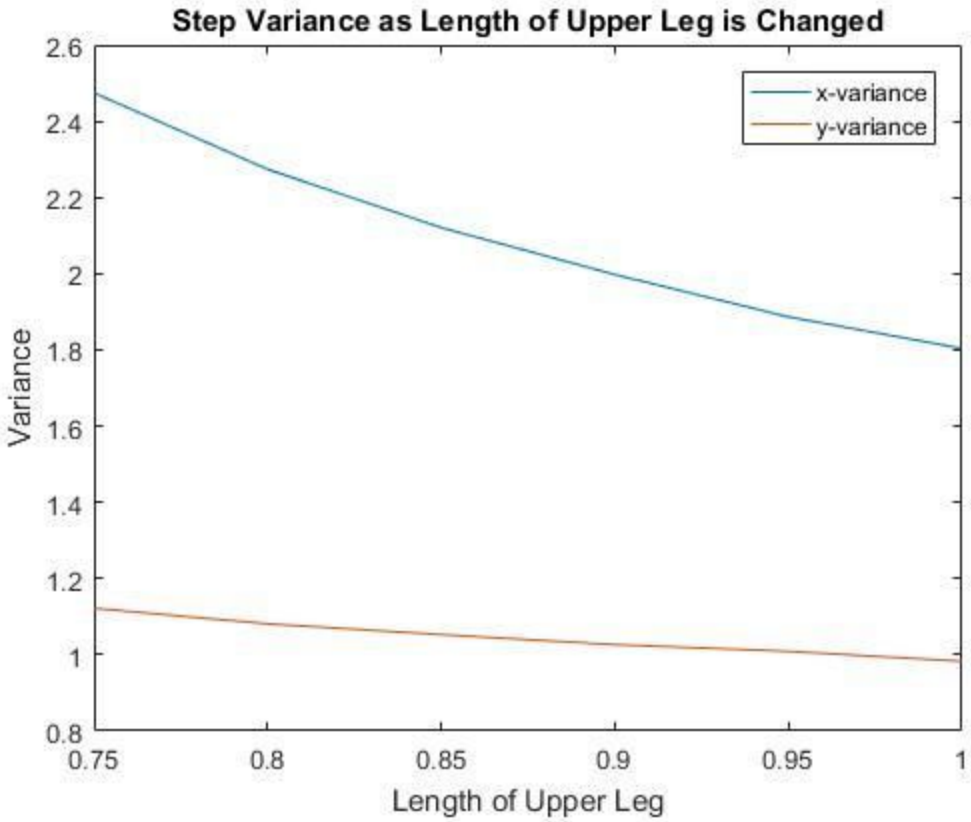

Figure X. - Leg 1 Step Variance vs Upper Leg Length

Figure X shows that, as the length of the top of the leg grows, the x-variance and y-variance decrease, though the y-variance stays relatively constant. In order to maximize the step length and height, the lowest upper leg length should be used.

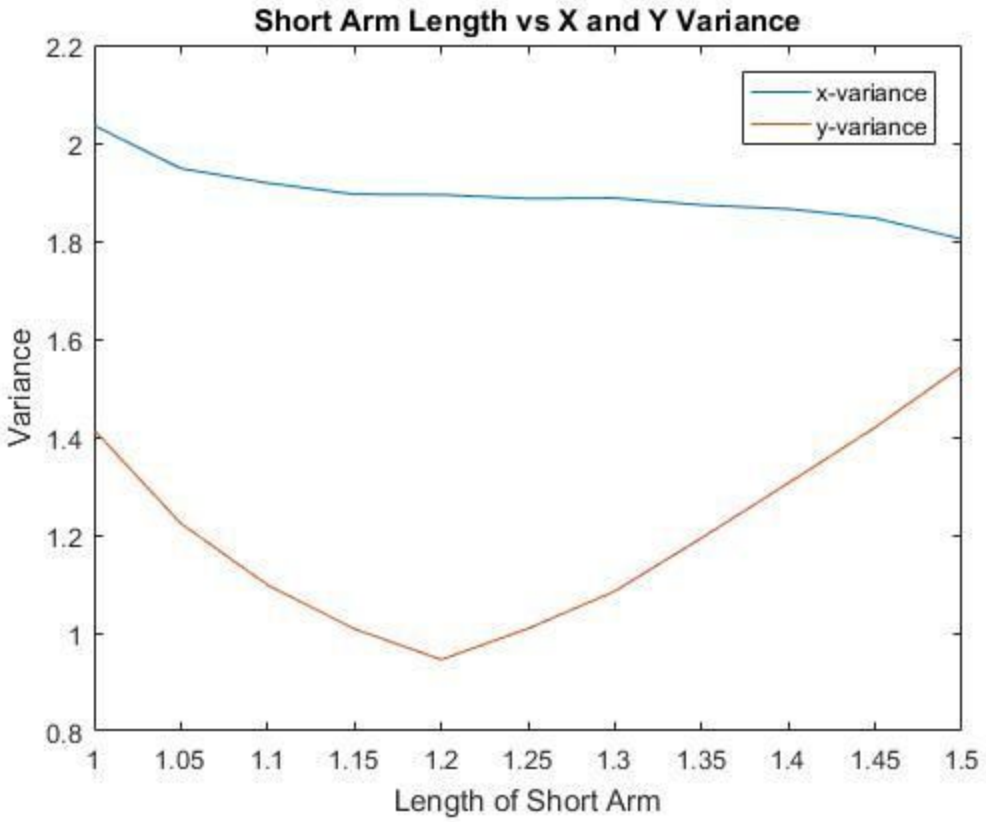

Figure XI. - Leg 1 Step Variance vs Short Arm Length

As the length of the short arm grows, the x-variance only decreases, but the y-variance decreases, reaches a minimum, and then increases past its original variance. This could mean that as the length of the short arm increases, the y-variance reaches a maximum and the x-variance reaches a minimum so a design could be envisioned which maximizes step height at the cost of step length.

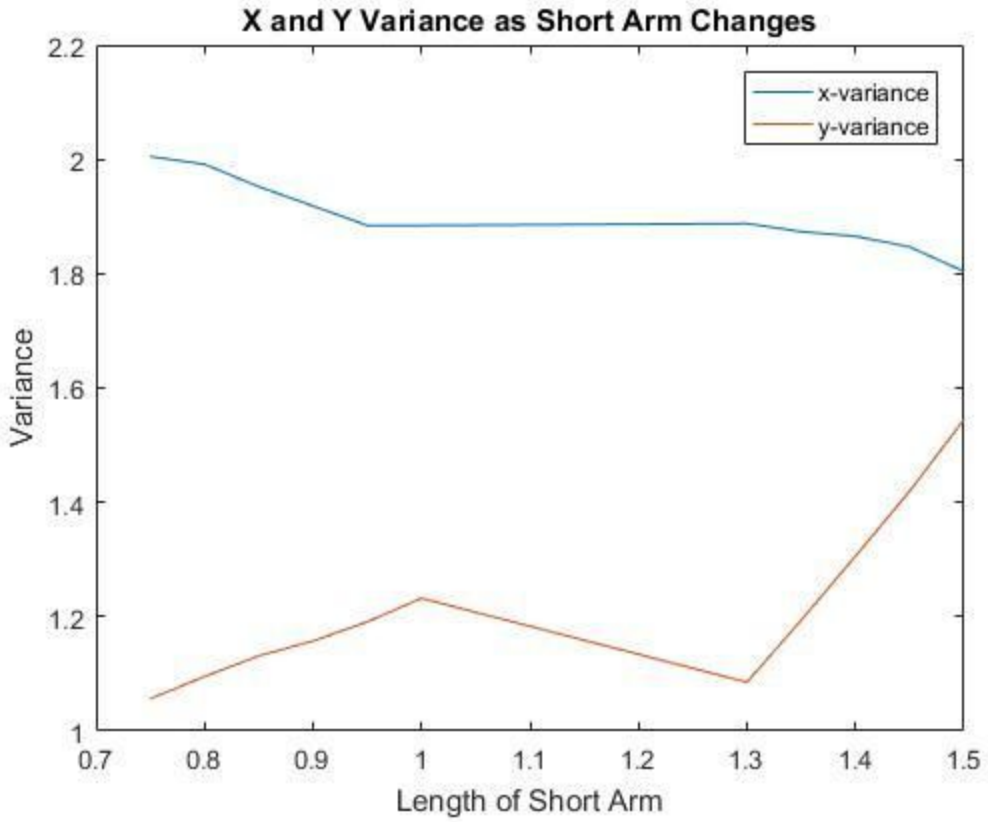

Figure XII. - Leg 2 Step Variance vs Short Arm Length

This graph shows how the increased complexity that the extra second leg shape has over the first. With a similar variation in the length of the short arm, the second leg has both a local maximum around 1 and a local minimum around 1.3 followed by a steep increase for the y-variance. Despite this difference in the shape of the y-variance curve between the first and second leg shapes, the x-variance curves are very similar. This difference reiterates the importance of doing an analysis for both leg shapes.

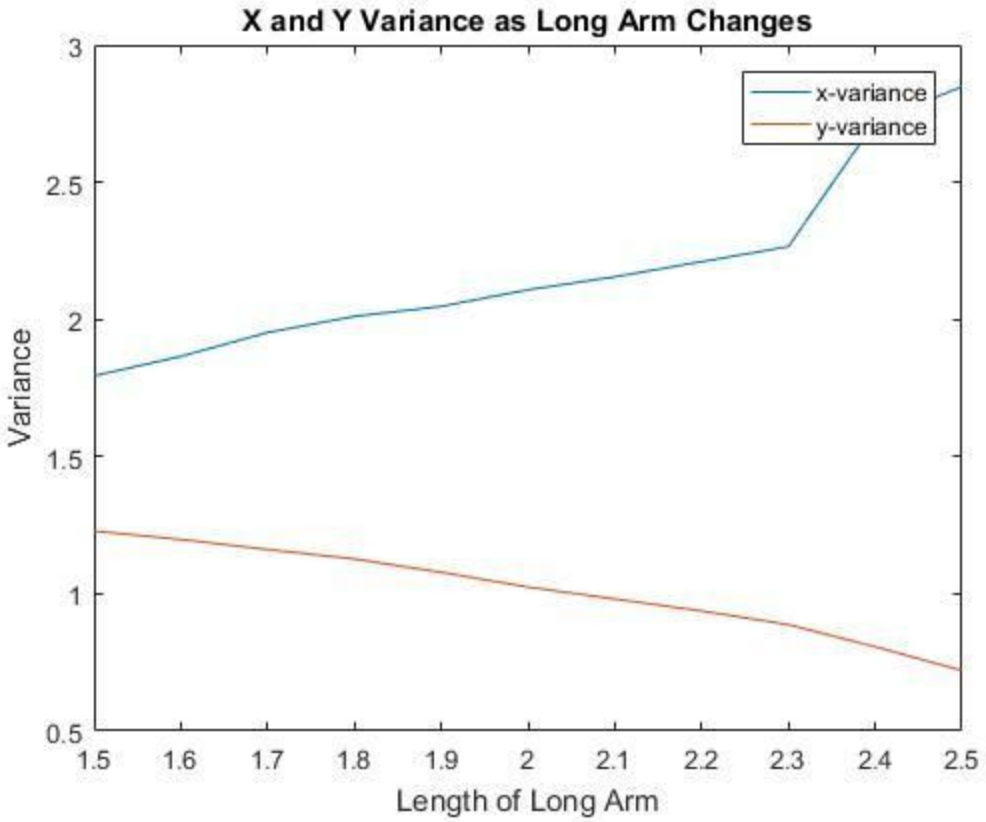

Figure XIII. - Leg 2 Step Variance vs Long Arm Length

In this figure, the x-variance for leg 2 increases dramatically after the long arm length reaches 2.3. This relationship is misleading though because when the actual shape of the leg path is observed as in the figure below, one can see that the leg shape completely changes after the long arm length reaches 2.3 and does not form a complete path anymore. This shows the importance of not only quantitatively analyzing the variance of the path but also qualitatively analyzing the shape of the path itself.

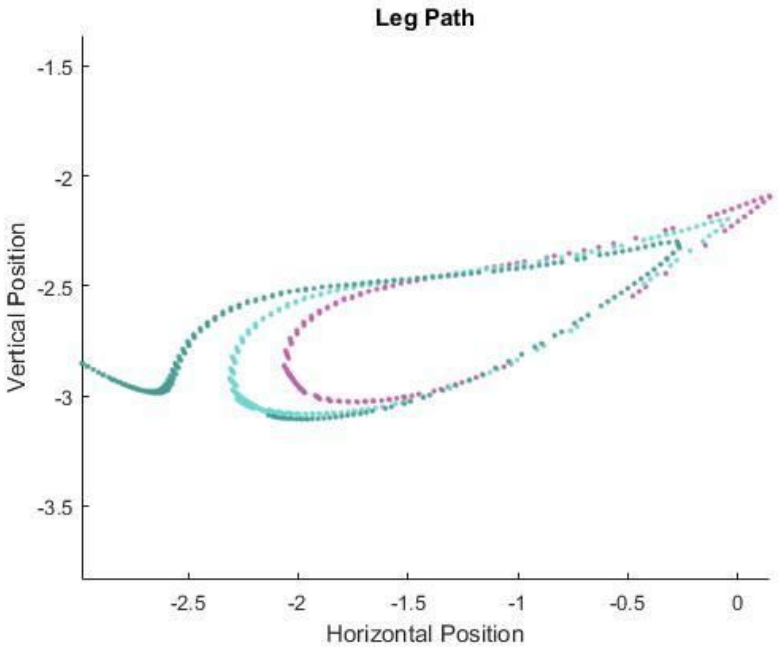

Figure XIV. - Leg 2 Path with Changing Long Arm Length

This analysis only shows results from some of the many variable that can be tested. The length of the bottom of the leg and the radius of the central mechanism path do not affect the actual path very much. Those two variables mainly just scale the size of the path up or down. The x and y position of the short arm pivot with respect to the central mechanism and the position of the point that the long arm of leg 2 must pass through are both important variables that still need to be tested. These variables are the most important factors in the actual shape of the path and can have a tremendous impact on the variance of the robot's step.

Discussion

Our findings are significant because we have found what we deem the optimal coverage route that first visits predetermined highest priority danger areas and remains there for long periods of time to find potential threats as opposed to quickly navigating and cycling through any high priority area. However, further studying of different algorithms could improve Moreover, our camera, using Google's API, was able to detect many different objects at various ranges, and as object recognition technology advances, we would like to see new tests with newer algorithms. As the algorithmic complexity increases, the demand on the system increases. Our current use of the trained Tensorflow model can run at a limited speed and accuracy on a previous-model mobile GPU. An increase in computing power would increase the number of

frames that the software could process, but usually comes with increasing the size and weight of the robot. While a larger robot could carry a series of high-end GPUs and professional grade cameras, this would not be feasible in a crowded environment such as a train station. This is also not necessary as high-end GPUs are being minimized to fit into very small bodies. The amount of graphical processing power required for our application to run smoothly could be fit into a small notebook laptop, and will only decrease in size over time.

For the physical robot our results showed the effect that various variables have on the shape of the path of the robot. These trends can be used to design the optimal configuration based not just on the current project, but on any design requirements for future projects. Our findings also show that additional research is necessary to fully understand the effects of different variables on the path of the robot. Mainly, the two leg paths must be combined to form a full model for the movement of the robot. Additional tests also need to be done on the effects of changing where the pivot points are located relative to the central mechanism. The program that defines the paths for the robot could also be refined to only analyze closed paths. This would eliminate problems such as those seen in Figure XII and XIV where there is a “jump” in x-variance due to the program finding an open path.

Acknowledgements

I thank Dr. Derrick Yeo, my Research Educator, for his assistance and advice as I was developing this research project. I would also like to thank Robert Paradiso and Ian Delk, fellow FIRE researchers, for their assistance in this project.

References

Alterovitz, Ron, Sven Koenig, and Maxim Likhachev. “Robot Planning in the Real World: Research Challenges and Opportunities.” *AI Magazine*. 37.2 (2017): 1-13. Web. 3 Dec. 2017.

Armada M.A., Cobano, J.A., Estremera, J., Garcia, E., Santos, P. 2007. A six-legged robot-based system for humanitarian demining missions. Mechatronics. 17, 417-430.

Becker, Stefan et al. “Detecting Abandoned Objects Using Interacting Multiple Models.” *Proceedings.* 9652 (2015): N.pag. *SPIE.* Web. 5 Nov. 2017.

Beynon, Michael D. “Detecting Abandoned Packages in a Multi-Camera Video Surveillance System.” *Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance*. N. vol. (2003): 221. *ACM.* Web. 5 Nov. 2017.

Chan, Sewell. “New Cameras to Watch over Subway System.” *New York Times*. 24 Aug. 2004. Web. 5 Nov. 2017.

Department of Homeland Security and the Federal Bureau of Investigation. “Potential Terrorist Attack Methods.” NSA Archive at George Washington University. 23 Apr. 2008. Web. 5 Nov. 2017.

Gasparetto, A., Vidoni, R. February 2011. Efficient force distribution and leg posture for a bio-inspired spider robot. *Robotics and Autonomous Systems*. 59.2, 142-150.

Larcher, Martin et al. “Effectiveness of Finite-Element Modelling of Damage and Injuries for Explosions inside Trains.” *Journal of Transportation Safety and Security*. 8 (2016): 83-100. *Web of Science.* Web. 5 Nov. 2017.

MTA New York City Transit. “TransitTrax.” MTA N.p. N.d. Web. 5 Nov. 2017.

Prassler, S., Scholz, J., Fiorina, P. July 1999. Navigating a railway station in a wheelchair during rush hour. *The International Journal of Robotics Research*. *18*(7). 711-727.

Pratihar, D.K., Roy, S.S. September 2011. Effects of turning gait parameters on energy consumption and stability of a six-legged walking robot. Robotics and Autonomous Systems. 60, 72-82.