CS4758: Moving Person Avoider

Yi Heng Lee, Sze Kiat Sim

***Abstract*—We attempt to have a quadrotor autonomously avoid people while moving through an indoor environment. Our algorithm for detecting people uses a Haar classifier to detect a person's upper body in the image feed from the front camera of the quadrotor. We then localize the person in a 5 x 5 grid based on the location and size of his detected feature. We use this grid to form a Markov decision process (MDP) problem and use reinforcement learning to determine the optimal control policy for the quadrotor so as to avoid colliding with the person.**

I. Introduction

Robots are increasingly required to navigate through crowded and dynamic environments in order to perform their tasks. The ability to detect and avoid moving people is thus crucial to the successful completion of these tasks. The task of avoiding people is not entirely simple for a robot, however, as it requires the robot to detect a person using its sensors, such as an onboard camera, and to execute planning and control algorithms to successfully avoid the moving person.

In this project, we attempt to navigate a quadrotor through a series of indoor environments that are largely free of obstacles while avoiding moving people. The quadrotor should maintain a safe distance of at least 1 m from an oncoming person, adjusting its flight path as required. We attempt to do so by using the Haar classifier to detect a person, and running a reinforcement learning algorithm to determine the optimal path to be taken by the quadrotor.

We test our algorithms by navigating the quadrotor through a number of indoor environments. The results show that the quadrotor performs well in open indoor environments, and fairly well in enclosed areas.

II. Related Work

In the area of obstacle and person detection, previous works include papers on navigating aerial robots in an indoor environment and avoiding obstacles using vision-based sensors. Bills, Prakash and Leung [1] developed a vision-based algorithm, using a support vector machine (SVM) to detect obstacles in the image stream from the front camera of an indoor helicopter. Control algorithms are then used to avoid the detected obstacles in an indoor environment. Work related to object or face detection using Haar-like features include papers by Lienhart and Maydt [2], and Viola and Jones [3].

In the area of avoidance and control, Michels, Saxena and Ng [4] modeled a remote-controlled (RC) car control problem as a Markov decision process, and used reinforcement learning techniques to develop a control policy that would steer the RC car to avoid obstacles. Similarly, Bou-Ammar, Voos, and Ertel [5] used reinforcement learning to control and stabilize a quadrotor.

We develop a vision-based algorithm based on these related works, using a Haar classifier to detect people and reinforcement learning to determine the optimal control policy for the quadrotor.

III. Hardware Platform

The robot used in the project is the Parrot AR.Drone, a WiFi-controlled quadrotor. A laptop computer runs the various vision and learning algorithms, and provides control instructions to the quadrotor through a wireless connection. The quadrotor has a front camera and a camera aimed downwards. The front camera provides an image stream with a resolution of 320 x 240 pixels, which is used as the input for the people detection algorithm. The quadrotor also has downward facing sonar sensors which measure the altitude of the quadrotor.

IV. Person Detection and Localization

We used a Haar classifier implemented in the OpenCV library to detect people. We first evaluated several different Haar cascade files in the OpenCV library to find the optimal cascade for our vision algorithm. Cascades that detected a person's face, such as "haarcascade_frontalface_alt.xml" and "haarcascade_frontalface_default.xml" had a high accuracy of detection, giving a low rate of false positives (falsely classifying objects that are not faces as being faces), but were unsuitable for the quadrotor due to their short detection range. These cascades were often only able to detect the person's face when the person was within 1 m of the quadrotor, and often at these distances the person's face would be too high to be captured by the quadrotor's front camera, depending on the altitude that the quadrotor was flying at.

We tried to solve this problem by running the "haarcascade_lowerbody.xml" simultaneously with the face detector so that the person's lower body would be detected if the quadrotor was flying at too low an altitude to see the face. However, the "haarcascade_lowerbody.xml" cascade worked poorly and often failed to detect the person, while registering chairs in the room as false positives. We eventually chose the "haarcascade_upperbody.xml" cascade as it had a longer detection range of about 3 m. However, this cascade also gave a higher rate of false positives. For example, it would sometimes classify the doors in the Upson hallway as positives. We ran tests to determine the localization accuracy of this upper-body classifier and the results are shown in a subsequent section.

The upper-body Haar classifier returns a CvRect object for each detected person, which describes a rectangle enclosing the detected feature, as shown in figure 1. We localized the person in a 5 x 5 grid, as shown in figure 2, which represents a top-down view of the environment ahead of the quadrotor. The width of each square in the grid represents approximately a person's step size. A person's x-coordinate represents his horizontal position relative to the quadrotor, such as whether the person is to the left or the right of the quadrotor, while the y-coordinate represents the person's distance from the quadrotor. The quadrotor is always located at the (2, 0) position in the grid, with the grid moving with the quadrotor. Therefore, if the quadrotor moves one cell to the left, the entire grid moves left, causing people in the grid to be shifted one cell to the right to reflect their positions relative to the quadrotor, while the quadrotor remains at square (2, 0).

Fig. 1. An image from the quadrotor's camera with two rectangles corresponding to the features of the two people detected by the Haar classifier.

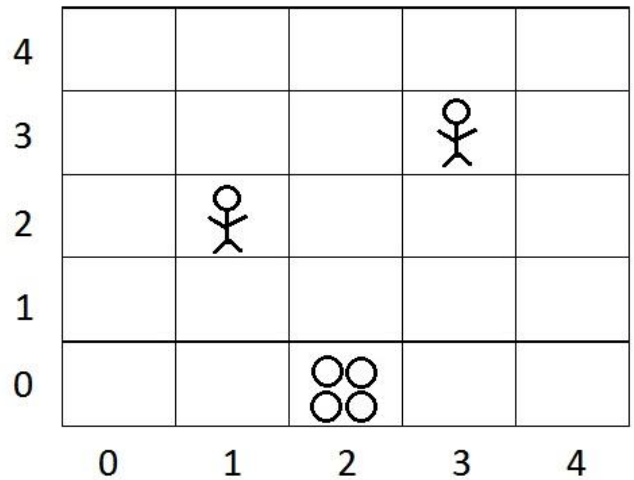

Fig. 2. An illustration of the 5 x 5 grid used to localize people relative to the quadrotor. The grid represents a top-down view of the environment ahead of the quadrotor. The quadrotor is fixed at position (2, 0) in the grid.

We localized a detected person in the grid based on the size of the CvRect object corresponding to the person's features. We note that a person further away from the quadrotor would be described by a smaller rectangle, and a person further from the center of the quadrotor would be described by a rectangle that is further from the center of the camera image. Therefore we normalized the x-coordinate of the center of the CvRect rectangle to get the x-coordinate of the person in the grid. A person with an x-coordinate within pixel 0 to 64 in the quadrotor's image would be placed in grid 0, while a person with x-coordinate within pixel 65 to 128 would be placed in grid 1, and so on. Similarly, the y-coordinate of the person in the grid, which represents the distance of the person from the quadrotor, was determined based on the size of the detected feature. A person described by a rectangle of width smaller than 60 pixels is placed in grid 3, while a person described by a rectangle of width smaller than 90 pixels is placed in grid 2, and so on. These parameters were chosen based on a set of test images.

We tried other person detection methods as well, but finally settled on the Haar classifier. Prior to the midterm presentation, we implemented person detection using Histograms of Oriented Gradients (HOG) descriptors together with a Support Vector Machine (SVM). We divided the 320 x 240 pixel image from the quadrotor's camera into a 5 x 3 grid of 64 x 64 pixel cells, similar to the grid used by Bills, Prakash and Leung [1]. HOG descriptors were used to extract the features in each cell and a SVM classifier determined if the cells contained a person's face and upper torso. The SVM was trained on a total of 150 positive (contains person) and 200 negative (does not contain person) 64 x 64 pixel images that we took using the quadrotor's front camera in the Upson 317 room. Testing the SVM classifier on images of group members standing in the room gave an 84% accuracy of classifying a positive image (identifying a person when the person is in the picture).

Based on the positions of people localized in the 5x5 grid shown above in figure 2, a reinforcement learning algorithm is then used to determine the optimal control policy.

V. Avoidance Using Reinforcement Learning

The reinforcement learning algorithm takes as input an array representation of the 5 x 5 grid of detected persons, such as the grid shown in figure 2. A grid of rewards is then created as shown in figure 3. The end states with y-coordinate 4, that is, the states furthest from the quadrotor are each given a positive reward of 10, since we want the quadrotor to move forward. Grid squares containing detected persons are given rewards of -100, since we want the quadrotor to avoid these squares. All other states have a reward of 0, and we do not localize a person in an end state.

Value iteration is then performed on the given grid of rewards to determine an optimal control policy. As seen in figure 3, the optimal action for the quadrotor, assumed to be in grid square (2, 0), is to move forward. The value iteration algorithm was determined to be able to converge in 60 iterations and 100 ms for all test runs with up to 3 people in the grid. Therefore, we decided to run this reinforcement algorithm real-time, instead of simulating the possible scenarios and storing the results beforehand.

The parameters of the Markov decision process (MDP) and reinforcement learning problem were calibrated based on our desired actions, forward, left or right, given a particular scenario. For instance, we determined that if a person were detected in grid square (2, 2), the quadrotor should avoid it by moving to the left or right. However, if a person is detected is square (1, 2) and is therefore not directly in the path of the quadrotor, we determined that the quadrotor should continue moving forward, and only avoid the person when he is in grid square (1, 1). Based on these desired actions and several test runs, we set the parameters of the MDP as follows:

$$A = \{\text{forward}, \text{left}, \text{right}\}$$
$$R(s) = \begin{cases} 10 & \text{if } s \text{ is an end state} \\ -100 & \text{if } s \text{ has a person} \\ 0 & \text{otherwise} \end{cases}$$
$$P_{\text{right}} = 0.85, P_{\text{wrong}} = 0.075$$
$$\gamma = 0.8$$

The probability P_{right} represents the probability of the quadrotor moving into the correct state given an action, while P_{wrong} represents the probability that the quadrotor moves into the wrong state (on either side of its desired action) given an action. For instance, if the quadrotor moves forward in grid square (2, 0), with probability P_{right}, it will move forward, and with probability P_{wrong}, it will instead move to the left or right.

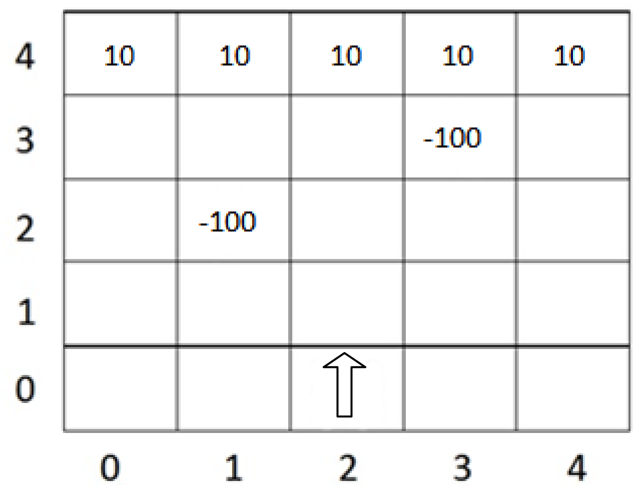

Fig. 3. An illustration of the 5 x 5 grid of rewards based on the input grid of detected persons from figure 2. The end states with y-coordinate 4 are given a reward of 10, while the grid squares with detected people are given a reward of -100. All other states have reward 0. The optimal action in grid (2,0) is to move forward.

Based on the optimal action given by the reinforcement learning algorithm, that is, to move forward, left or right, the control parameters of the quadrotor are then determined. We selected a pitch value of -2000 for a forward action (all other values 0), a roll value of -1000 for a left action, and a roll value of 1000 for a right action. These control parameters were determined to be optimal based on a number of test runs, as they allowed the quadrotor to move sufficiently fast forward while allowing sufficient time to detect and avoid an oncoming person.

VI. Experiments

We performed a series of tests to determine both the accuracy of the person detection and localization algorithm, and the performance of the reinforcement learning algorithm.

To determine the accuracy of the person detection and localization algorithm, we first took 20 images of a person in each possible grid square from (0, 1) to (3, 4) using the front camera of the quadrotor, for a total of 300 images. We labeled each of these images, ran the detection and localization algorithm on them, and determined if the algorithm both detected the person and placed him in the correct grid square. The results are shown in the following table.

y \ x	0	1	2	3	4
1	95%	95%	100%	95%	90%
2	100%	95%	100%	90%	85%
3	90%	90%	90%	85%	85%
Overall accuracy = 277/300 = 92.33%					

Fig. 4. A table of the accuracy of the detection and localization algorithm for the various grid squares.

We note that the detection and localization algorithm is quite accurate, giving an overall accuracy of 92.33%. In particular, the grid squares nearer to the quadrotor, such as those immediately in front of the quadrotor (in squares (1, 1), (2, 1), (2, 2) and (3, 1)) have at least 95% accuracy. This allows the quadrotor to accurately map out the people ahead of it, and to determine the optimal action in these scenarios. We also note that the grid squares further from the quadrotor, such as those with y-coordinate 3, have accuracy at most 90%, as the Haar classifier is occasionally unable to detect the person at a far range. However, this is not as crucial, as the quadrotor should be able to detect the person as it gets nearer to the person.

Next, we tested the person detection and avoidance algorithm in three different environments, namely the narrow corridor outside Upson 360, the wide corridor outside Upson 317 and the room environment in Upson 317, as shown in figure 5.

Fig. 5. Images of the three environments where we tested the quadrotor, taken using the front camera of the quadrotor. On the top left is Upson 317, on the top right is the narrow corridor and on the bottom is the wide corridor.

We ran a set of 10 test runs of the quadrotor with a moving person in each environment. Each test run involved the person standing 3 m from the quadrotor initially, with the quadrotor placed in the center of the environment. The person would then walk straight towards the quadrotor at a normal walking pace, and the quadrotor is determined to have successfully avoided the person if it passes the person with a gap of at least 20 cm between both the wall and the person, and moves on for a distance of at least 1 m. Therefore the quadrotor is considered to have failed the run if it touches or goes too close to the wall or the person at any point. Based on these criteria, we obtained the following results.

Environment	No. of test runs	Success rate
Narrow corridor	10	60%
Wide corridor	10	80%
Room	10	90%

Fig. 6. Table of results of the test runs and the success rate in the different environments.

We see that the quadrotor performs well in an open environment like the room, and fairly well in a more enclosed area such as the corridor. In most instances, the quadrotor failed because it moved to close to the wall. This is unfortunately not accounted for in our algorithm, and future work could involve both moving people and stationary obstacles.

We ran a few tests of avoiding two moving people in the wide corridor and the quadrotor did fairly well, being able to avoid the two people in most of these tests. Some of these test runs can be seen in the attached video.

VII. Conclusion

We were able to successfully implement people detection and avoidance using the quadrotor in the Upson 317 room and corridor environments. Our approach of using a Haar classifier to detect and localize people, and reinforcement learning to determine the optimal control policy performed fairly well in these indoor environments.

Future work could involve incorporating obstacle detection perhaps by using the SVM classifier described by Bills, Prakash and Leung [1], and localization of the quadrotor within the learning grid based on the image from the front camera or with the use of additional sonar sensors. Additional actions could be also considered in the MDP, such as remaining still when the path ahead is blocked, or turning to observe other parts of the environment. Some form of memory, perhaps with the use of hidden Markov models (HMMs), could also be incorporated so that the quadrotor can keep track of the recent movements of the person, or the relative motion of two oncoming people.

Acknowledgments

We would like to thank Prof. Ashutosh Saxena and the course staff of CS4758, in particular, Cooper Bills, for their invaluable guidance and feedback throughout the project.

References

[1] C. Bills, A. Prakash, and T.S. Leung. “Vision-based obstacle detection and avoidance”. http://www.cs.cornell.edu/Courses/cs4758/2010sp/final%20project/Cooper%20Bills,%20Arjun%20Prakash,%20and%20T.S.%20Leung.pdf

[2] R. Lienhart, and J. Maydt, “An extended set of Haar-like features for rapid object detection,” *International Conference on Image Processing*, vol. 1, pp 900-903, December 2002.

[3] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp 511-518, 2001.

[4] J. Michels, A. Saxena, and A.Y. Ng, "High Speed Obstacle Avoidance using Monocular Vision and Reinforcement learning," *Proceedings of the Twenty-first International Conference on Machine Learning (ICML)*, 2005.

[5] H. Bou-Ammar, H. Voos, and W. Ertel, "Controller design for quadrotor UAVs using reinforcement learning". *Proceedings of the IEEE International Conference on Control Applications*, pp 2130-2135, 2010.