Efficient People-Searching Robot

Anh Pham(✉), Mayang Parahita(✉), Andy Tsang, Mathias Chaouche, and James Rees

University of Birmingham, Birmingham, UK
{bdp414, mdp358, cyt493, mxc526, jxr227}@student.bham.ac.uk

Abstract. This paper describes a robot that performs person search in an office environment. The system brings together a number of technologies to ensure efficient and robust search in the face of varying conditions. These include face detection, face recognition, speech recognition, localization, path planning and re-planning in the presence of dynamic obstacles, and a decision theoretic search strategy. The main novel contribution of the system is an ensemble method for combining the outputs of multiple classifiers for face recognition. A second contribution is the use of a strong priority over the location of individuals in the database to guide search. The system is also able to use speech generation and recognition to interact with individuals to achieve its goals, as well as to receive goals via a mobile interface.

Keywords: Person search · Navigation · Face recognition · Ensembles

1 Introduction

Mobile robots are often required to interact with people. An important skill is the ability to efficiently and robustly search for specific people. To be efficient, the robot should search using strong prior knowledge over the likely locations of a particular person. To be robust, person recognition must be reliable in the face of varying lighting conditions, dynamic obstacles and unreliable modes of interaction (speech recognition, face recognition). In this paper we present a system that has been engineered to be robust to these challenges. In particular, the system uses strong prior knowledge over person locations and multiple face recognition methods to improve performance.

The implementation is on a P3-DX robot, controlled using speech and keyboard. Given a command, the robot guides a user through the most likely sequence of locations for the target person. The robot drives smoothly at a suitable pace and avoids dynamic obstacles. The robot can detect whether doors are closed, and request help, identifying when each door has been opened. The target person is found using state-of-the-art face recognition methods that are combined in an ensemble using plurality vote. If the target person is not found, the robot will search other possible rooms in descending order of probability.

L. Alboul et al. (Eds.): TAROS 2016, LNAI 9716, pp. 249–254, 2016.
DOI: 10.1007/978-3-319-40379-3_25

2 Background

2.1 Navigation

Before navigation takes place, often a map must be generated in order to guide planning and movement. This involves using sensors that allow the robot to perceive (such as cameras, sonar, and laser) to form a model of the environment. Since sensors are subject to noise and the environment can change over time, most state-of-the-art mapping techniques are probabilistic and simultaneous [1].

Existing navigation techniques employ two types of internal map representation: Metric-based and topological-based. Metric-based representations use a 2D coordinate systems in which each object is assigned a global position. Topological approaches usually define a space using places and the connection between them, essentially a graph with vertices as gateways, places of interest, or intermediate points, and a set of edges as paths between them. Having obtained the appropriate navigational map, a search is performed to find a path given starting and goal positions. The path is then used to navigate, complemented by localization to confirm the robot's position.

2.2 Face Recognition

There are three popular face recognizers: Fisher, Eigen and Local Binary Pattern Histograms, each operating differently. A Fisher recognizer maximizes the mean distance of different classes while minimizing the variance within the class, in order to perform better on linear discriminant analysis. An Eigen recognizer finds a linear combination of features that maximize the total variance in data and produce Eigenvectors for the recognition process. An LBPH recognizer compares the 8 neighbors of each pixels and used the result as a texture descriptor.

It is clear that each recognizer has its own advantages. For instance, Fisher has better accuracy with various facial expression, Eigen provides high efficiency in computation resource but very sensitive to light variation [2], while LBPH is invariant to monotonic gray-level changes and computational efficient [3]. However, to our knowledge, few attempts were made to combine different face recognizers [4].

3 Design

The physical set up consists of a P3-DX mobile robot connected to a laptop with an external camera and microphone mounted on a tripod (height of 150 cm). The system is made up of separate constituent components with a single node to control and manage the behavior of the robot as a whole. The software is written in Python using Rospy, a popular library providing interface to mobile robots.

3.1 Database

During this study, topological data for 9 rooms and 10 people were used. The initial probability of finding each person in each room is stored in an SQLite table, which can be queried to obtain possible locations of target person in order of descending probabilities. If different rooms have the same probability, the robot go to the closest first to ensure efficient navigation. Furthermore, during and after a task, the knowledge about the target person is updated, allowing the system to keep an accurate record of the targets, hence maximizing the efficiency of the task.

3.2 Input and Output

Speech recognition was implemented to provide a natural control method. The acoustic signal is first converted to an actual sentence using Google's REST API, which is then parsed using a set of regular expressions. For instance, if the parser receives "Find Andy", the task will start with "Andy" as the target person. As the module involves sending the voice recording via internet to Google, the system is prone to wireless disconnectivity. Hence, the module emits regular beeps between which the user could speak, and emitting a double beep to signal a connection error.

Despite many benefits, speech recognition was found to be unreliable. Therefore a keyboard input, which uses the same parser - was implemented for a more reliable and robust approach which also caters for users with speech impairment. A mobile interface to send commands to the robot was also developed.

A speech synthesizer was implemented to provide user friendly feedback by vocalizing the current progress. With Ubuntu, a text-to-speech application Espeak is provided natively, and a Python wrapper module Pyttsx has been used to utilize it with a speech rate of 200 words per minute.

3.3 Localization and Navigation

The implementation uses Adaptive Monte Carlo Localization [5] and Grid-based navigation. At initialisation, an overlay grid is generated, with cell marked as either an obstacle or a free space based on the map data. After that, A* search is run using the grid to find the shortest path to goal. To avoid dynamic obstacles, each free cell is given an occupancy probability, which is then increased or decreased based on the laser scan. Cells exceeding 50 % will be marked as occupied. If needed, a new path will be computed and followed, allowing the robot to avoid unexpected obstacles such as bags and people very well.

Initially, the robot moved points to points, pausing to rotate when needed. Despite guaranteeing the shortest distance travelled, a large amount of time was spent to pause and turn. Hence, the angular and linear speeds of the robot are changed smoothly according to different factors, allowing smooth and natural movement. In addition, a feedforward vector representing momentum is updated based on the robot's movement – the faster the robot moves, the bigger its momentum becomes. This feedforward

control is used to predict the robot's future pose and steer the robot according to the extrapolation of its position.

To address localization failures, a relocalize mode is activated if the average weighting of particles remains too low for too long. In such cases, the robot pauses its current task and starts spinning around while adding a number of random particles on the map every frame. After a period of time, the right position will be found and the average weighting will increase enough to stop the relocalize mode, allowing the robot to return to its current task.

3.4 Face Detection and Recognition

Using OpenCV library, the face detector extract faces from a video feed and creates grayscale images of 256×256 pixels, which are then fed into face recognition, avoiding unnecessary computation spent on non-face pixels.

The face recognition module controls the recognizers provided by OpenCV including LBPH, Fisher and Eigen. The training set includes 10 faces, 7 of which from the AT&T Face Database [5]. Training data of the remaining 3 faces was taken in different places within the testing area, some of which were lit by artificial sources from various directions. Whenever new data is added, the whole training set is reprocessed. Otherwise the recognizer will load a saved training file (.yml) to minimize computation time.

After the training process, whenever a face is given to the recognizer, it will return the most likely person, along with the Euclidean distance between the testing image and the closest found image. Therefore, the lower the returned value is, the higher the chance for the prediction to be correct. A threshold is set for every recognizer in order to ignore faces with low probability.

Each recognizer has different performance in reference to the size of training set, light variance of testing image and the variation of facial expressions. Therefore, instead of using a single recognizer, our design uses all three recognizers to improve the result. The combined method will recognize a face only when all three recognizers agree on the same ID (comparable to using AND operators). The combination was found to be more robust with lower false positive rate, justifying the extra computational resource required.

4 Preliminary Results

4.1 Speech Recognition

The success frequency of speech commands was recorded. Using 5 different voices, command sentences were pronounced a total of 100 times. The results shows a very low True Positive Rate of 0.561, indicating how the speech command is only successfully processed after several attempts.

4.2 Path Planning and Driving

Choosing different starts and goals, the performance of pathfinding is found to be 0.001177 s on average, with the maximum time recorded being 0.03298 s, which means the robot can quickly re-plan to avoid obstacles without using up resource and affecting the system. To show the improvements of the driving code, tests with each feature turned off were compared against the complete driving code, using the same 31-m path. In the first set of tests, the current position given by localisation is used without extrapolation. In the second set, the robot can only do one type of movement at a time, either rotating or moving forward. In the last set, the complete code was used. Moreover, to keep the results deterministic and reliable, a static environment was used with no dynamic obstacles and detection (Fig. 1).

Fig. 1. Average performance of each set of tests in seconds

The difference was tremendous: feedforward vector and continuous movement improve driving time by roughly 45 % and 27 % respectively. In addition, the code performed similarly when tested with different routes, showing its consistency. Furthermore, Fig. 2 shows how the robot can avoid dynamic obstacles using very narrow pathways in between.

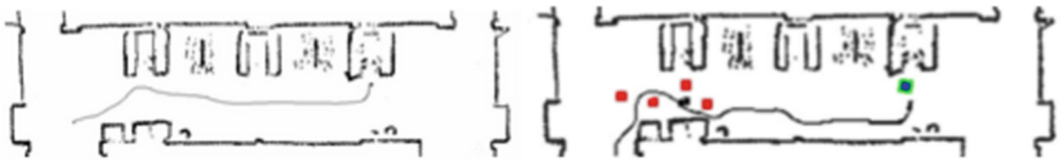

Fig. 2. Navigation path through empty corridor (left) and obstructed by obstacles (right)

4.3 Face Recognition

For face recognition, we conducted several tests and found that 800, 95 and 10000 were the optimal distance thresholds for Eigen, LBPH and Fisher respectively.

Figure 3 shows that the ensemble of three recognizers performs better than single ones, giving an accuracy of 0.85. Moreover, it has a False Positive Rate of 0 which is extremely desirable in our system since even with a large database and a constant video feed, it will never wrongly recognize the target person.

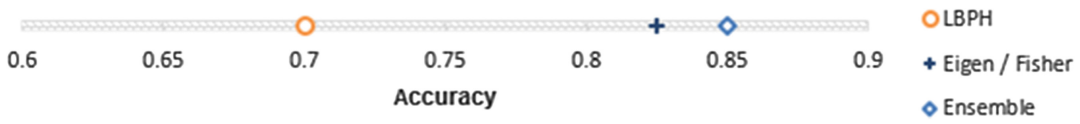

Fig. 3. Accuracy of individual face-recognizers and the ensemble of all three

4.4 Systems Tests

The system was tested as an integrated unit at multiple stages throughout the development process to ensure that all components work well together to fulfill the overall task, and to give guidance to necessary changes. In the final iteration, the system was able to receive keyboard commands, while speech recognition was not reliable. The robot could then navigate to the destination with a 95 % success rate. The few trial failures were due to obstacles that could not be navigated around. The robot proved capable of stopping in a suitable position that will provide sufficient space to perform face detection. The face recognition was mostly successful with a success rate of 85.7 % and an average recognition time of 22.8 s, however the variance of light and facial expression relative to the quality and quantity of the training set had a strong negative effect on the result. Overall, the system takes around 2-5 min to find a person, depending on where he or she is.

5 Conclusion

In conclusion, although it is still work in progress, the project has successfully combined various techniques, including both standard and novel methods. In future work, components will be improved in various ways: a denser grid will be used to smoothen the robot's movement and detect obstacles more precisely, while the face recognizers will be optimized further to give better results. Moreover, the tracking of people will be fully autonomous. Once refined, it can be widely deployed, especially in an office environment such as a university building.

References

1. Thrun, S.: Robotic mapping: a survey. In: Lakemeyer, G., Nebel, B. (eds.) Exploring Artificial Intelligence in the New Millennium, pp. 1–35. Morgan Kaufmann Publishers Inc., San Francisco (2003)
2. Jaiswal, S., Bhadauria, S., Jadon, R.: Comparison between face recognition algorithms - eigenfaces, fisherfaces, and elastic bunch graph mapping. JGRCS **2**(7), 187–193 (2011)
3. Ahonen, T., Hadid, A., Pietikainen, M.: Face description with local binary patterns: application to face recognition. IEEE Trans. Pattern Anal. Mach. Intell. **28**(12), 2037–2041 (2006). doi:10.1109/TPAMI.2006.244
4. Zhao, W., Chellappa, R., Phillips, P.J., Rosenfeld, A.: Face recognition: a literature survey. ACM Comput. Surv. **35**(4), 399–458 (2003). doi:10.1145/954339.954342
5. Database of Faces. AT&T Laboratories, Cambridge. http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html (2002). Accessed 11 Jan 2016