ButlerBot: The Robotic Butler

Diana David, Zion Adika, Liu Yang, Vitaly Bokser

Additional Team Members: Edmond Hakimian (Data Integration)
Zeynep Altinbas (Speech Recognition)
Richard Spillane (Computer Vision)

Stony Brook University
Computer Science Department
Stony Brook, NY 11794-4400
ddavid@ic.sunysb.edu, yliu@cs.sunysb.edu

Abstract

ButlerBot is a robotic butler created to cater to people's questions at the International Joint Conference on Artificial Intelligence (IJCAI) 2003 conference and to escort people to desired destinations. The three main areas of research focused on in ButlerBot, the Stony Brook Robot Design Team's new robot created for the American Association of Artificial Intelligence's (AAAI) Robot Host competition, are navigation, computer vision and data server / human interaction. ButlerBot was the second place winner at the AAAI 2003 Robot Host competition.

Introduction

At the 2003 International Joint Conference on Artificial Intelligence (IJCAI), the American Association of Artificial Intelligence (AAAI) held its annual robotics competition and exhibition. Robots had the option to enter four different events including the robot host, the robot exhibition, the robot rescue and the robot challenge.

The robot host event involves serving data related to the IJCAI conference to people in attendance, as well as escorting users from their current location to a desired destination. Teams participating in the robot exhibition event are able to show interesting and unique research in the field of robotics that is perhaps not covered by any of the other events. For the robot rescue event, AAAI creates a scenario where a disaster has occurred and thus left dummy-people trapped. Robots then search through the rubble to locate potential victims they could save. The robot challenge event participants create robots capable of finding their way to the conference registration desk, registering themselves, finding the room they are meant to present in and then giving a presentation on themselves and the technologies used in their creation.

The Stony Brook Robot Design Team created ButlerBot for the AAAI robot host event and robot exhibition event. In the matter of three short months, all of ButlerBot was designed, implemented and ready to serve as a butler at the conference. The team was divided into three subgroups: navigation, computer vision and data server / human interaction. The navigation group focused on escorting people from one location to another while simultaneously tracking itself. The computer vision group worked closely with the navigation group for example to help the robot realize its position based on landmarks, which were red, green, blue and yellow balloons. The data server / human interaction group devised a way to store relevant conference information to serve to individuals and created a user-friendly interface to interact with people.

System Architecture

The main driver of ButlerBot's software system is a Finite State Machine (FSM). The four main components connected to the driver are the user interface, navigation, vision and the text processing / information server. All of the components communicate via TCP/IP socket connections. Figure 1 shows how ButlerBot's system architecture was designed to interact.

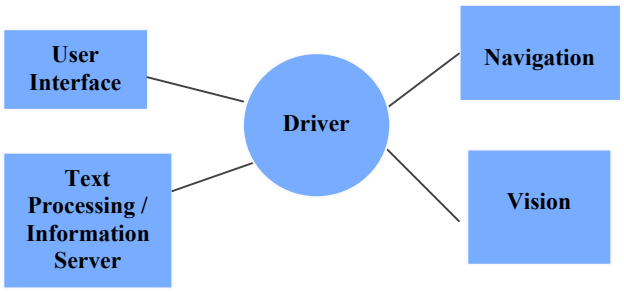

Figure 1: System Architecture Design

Navigation

Being that ButlerBot's main purpose is to serve and assist users within a designated area, such as a conference hall or

a hotel, a complex navigation system is not necessary. A simpler navigation system would reduce cost and potential error. An internal coordinate system and a means of keeping track of the robot's current coordinates suffice as the backbone for the navigation system. Ideally, this internal coordinate system should be easily replaceable with a different coordinate system structure based on a different environment. This would reduce the required effort in migrating the robot from one environment to another.

ButlerBot's navigation system consists of an internal coordinate system representation of the surrounding area. The robot keeps track of its current position on its virtual map. Currently the map is a planar map, though various floor maps can be included to create a complete representation of an entire building. Work is still needed to integrate the various floors by adding transition points on each map.

Each map data structure consists of three main components: an outer boundary, a list of inner static obstruction boundaries, and a list of predefined paths to reach various destinations from the robot's patrol route. The outer boundaries define walls and other stationary obstructions adjacent to the walls. Each individual boundary is stored as a vector of line segments. Internal obstruction polygons, each defined as its own vector of line segments, are also stored as part of ButlerBot's internal mapping structure. Whenever moving about, ButlerBot will never cross one of these line segments. The last component of the internal map is a list of paths that the robot can take to reach one of the valid destinations on its valid destination list. Each path must start at a point on the robot's "patrol path", which will be discussed shortly. When the robot decides to move to a new destination (based on its user interaction), it will first return to its "patrol path" and follow it to the starting point of the "destination path", after which it will follow the "destination path" until the destination.

When not en route to a particular destination, ButlerBot will be in "patrol" mode. In this mode, the robot will traverse a predefined series of line segments, occasionally stopping to look around for an "interesting" event. An "interesting" event can be defined as a motion, or the detection of an image resembling a human figure. If an "interesting" event is discovered, ButlerBot will temporarily leave its patrol route and head towards the interesting event in an attempt to seek out and approach potential users. In order to increase ButlerBot's chances of finding a person to assist, if no person has been detected for a predetermined amount of time while patrolling, ButlerBot will begin making random dips into areas not specifically listed as part of its patrol route. As part of ButlerBot's next design phase, work will be done to improve the patrol mode. In particular, the route will be traversed in a less mechanical manner, using a stronger heuristic algorithm for determining where in the path to head next and where to spend the most amount of time. The "dips" out of the patrol path will be less random and more heuristic-based as well. This enhanced algorithm will add a memory component to ButlerBot's decision-making engine. Areas in which more individuals are found will receive higher priority in the decision of where to move next and for what period of time to move to the new location. A weighted random number generator with dynamically modified weights can be used to implement this enhancement. With a heuristic navigation engine as such, it is anticipated that ButlerBot's effectiveness will be substantially improved.

As part of the ButlerBot navigation system, a set of six infrared sensors (distance sensors), were used to detect non-stationary obstacles and prevent collisions. Initially, a simple wall-hugging algorithm was used to attempt to clear the obstacle. Such a collision detection system was crucial to the effectiveness of the robot's autonomous navigation.

ER1 Hardware Communication Channel

The ButlerBot Navigation hardware, provided by Evolution's ER1 hardware platform, involved a command line interface to communicate in order to control the movement of the robot. The hardware API is accessed by opening a TCP/IP socket to the ER1 program running as a background task. The API contains simple commands that allow autonomous control of the robot, for example "move 10 meters" in order to move forward 10 meters or "move 90 degrees" in order to turn. These commands would then be converted by the ER1 software into a serial format recognized by the robot control hardware. The robot control hardware uses a Universal Serial Bus to interface the host PC. The stepper motors that drive the robot's motions are controlled by the robot control hardware, which translates the signals from the PC into appropriate stepper signal descriptions.

Limitations of the Autonomous Navigation System

The autonomous navigation system of ButlerBot depended too much on a map based environment, as well as its initial position. Thus it could not easily be placed in an arbitrary environment.

Another problem is that it did not account for slippage of the motors, particularly on different type of floor surfaces. These errors would accumulate and ultimately distort the robot's knowledge of its actual location.

With a high-precision odometer added to the system, the robot will be able to more effectively maintain an accurate record of its current position. However, this is only one part of the ultimate solution, as the wheels slipping on difficult surfaces would still be able to accumulate error in keeping track of the current position. Another method for increasing the accuracy of the current coordinates that are maintained by the navigation system, which is still being developed, is an auto-calibration system that would

periodically recalibrate the robots position. The current method of auto-calibration makes use of the vision analysis from the stereo camera. Utilizing beacons such as colored balloons or intense lights, ButlerBot can determine its actual current coordinates and readjust its position to counteract any inaccuracies.

Computer Vision

The two main tasks for ButlerBot's vision system are to detect human activity and to track its own location accurately.

Motion Detection

A statistical model was implemented to detect human motion by identifying non-static portions of an image's background that contain only small motions (Elgammal, Harwood and Davis 2000). Figure 2 shows a series of images where only the person's location changes between images, followed by a black and white image output from the motion detection algorithm to be discussed.

Image-processing techniques are used to enhance the quality of images being sent into the motion detection algorithm. Poisson filters and median filters are used to return large blobs containing objects with motion. In order to simplify the motion model and achieve more accurate results when navigating, larger motions are focused on while smaller motions are ignored. The final object of interest is chosen to be the largest blob in the resulting image. The Digiclops Camera from Point Grey Research (PGR) uses a stereo algorithm to retrieve depth information for each pixel in the reference image. The average depth for all of the pixels under the mask (where the mask is the largest blob detected) is used as the depth for the target.

Landmark detection

The location of ButlerBot is continually tracked, but error tends to accumulate as the robot moves around. The information pertaining to the robot's position is thus updated from time to time. Balloons are used as a landmark to help the robot adjust its knowledge of its current location. The first step is color detection since the balloons are vivid colors including red, green, blue and yellow. The image of the scene is first transformed into HSV (Hue, Saturation, Value) space. Next, only the pixels within a specific threshold are retained as possible pixels for a landmark. Again, the image is cleaned and noise is eliminated using image processing and enhancement techniques. Large blobs are then considered targets. The problem of determining if a target with a desired color is a landmark or not still remains though. Using an edge detection algorithm, only the edge of a target is outlined and highlighted. If the target is a balloon, all of the pixels creating the boundary of the balloon should be at the same distance from its center of mass based on the spherical geometry of a balloon. Due to the possible noise present in real data, the variance of the distance for each boundary pixel to the center of mass is calculated. If the variance is within a preset threshold, it is a balloon; otherwise it is considered an outlier object. Figure 3 shows an input image at the top and the binary output images created by the landmark detection algorithm.

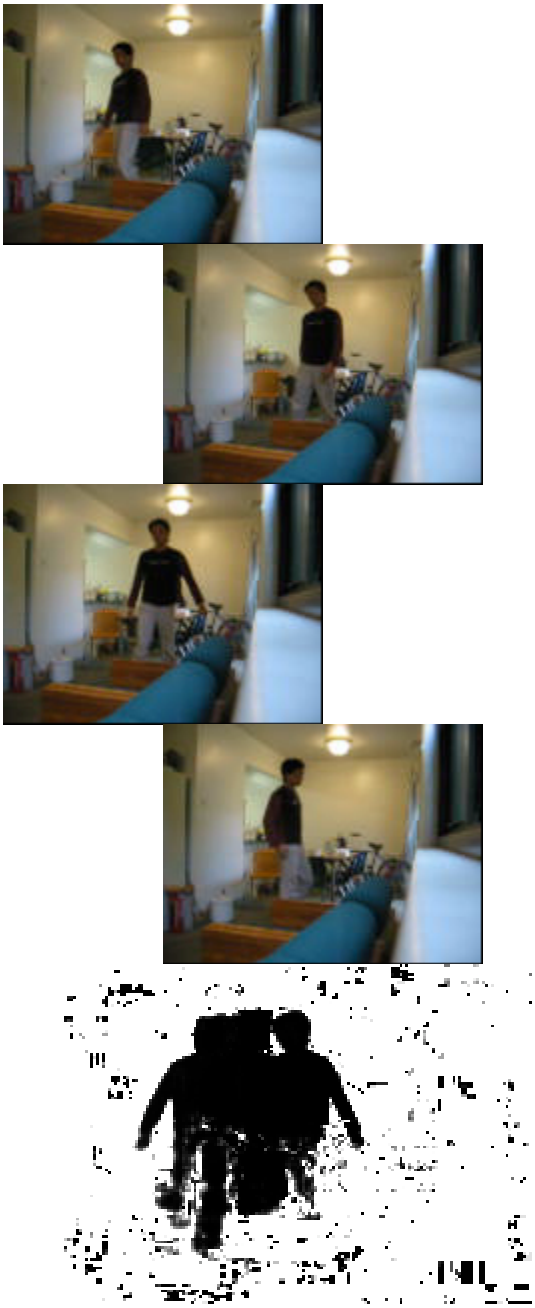

Figure 2: Motion Detection Inputs and Result

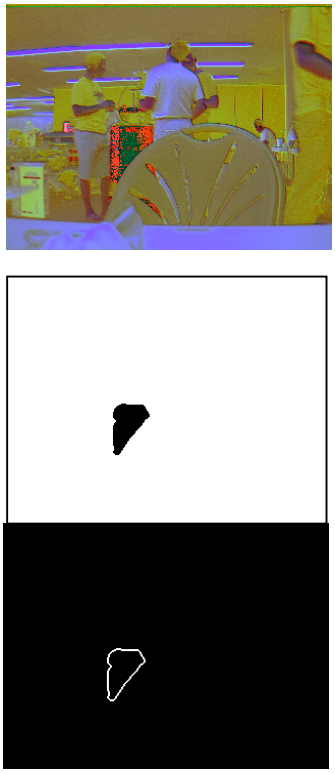

Figure 3: Landmark Detection

Once the position for two landmarks is determined, triangulation is used to determine the current location of the robot. A moving human figure identification module implemented by Richard Spillane and Liu Yang is not described here for reasons of brevity.

Data Server

A MySQL database was created to store conference data for ButlerBot to serve. When searching for an answer in the MySQL database, another database is used to substitute synonyms in the case that an answer is not found for the original question. Unfortunately, due to the time constraints of three months, the database was not fully integrated with the user interface. Rather a Perl script was written to convert the data saved in the MySQL database into text files with a specialized format. Figure 4 shows an example of a data entry in the new text file.

```
AAAI:
8
* AAAI is the American Association for Artificial Intelligence
fox:!sly
3
* Foxes scare me!!
* Do you like foxes?
```

Figure 4: Example of Data Storage

The first line of each data entry contains keyword combinations. The first row in Figure 4 would match to any sentence containing the word AAAI while the second row would match to any sentence containing the word fox but not the word sly. The intent is to eliminate the case where a person may enter a phrase such as "I am as sly as a fox" and to only match a statement to the results shown in Figure 4 if the person is truly speaking about a fox in terms of an animal. However, a dilemma arises when a sentence matches multiple sets of keywords. Therefore, on the second line of each data entry is the priority that keyword is given. From Figure 4 it can be seen then that if a sentence used both the word AAAI and fox, but not the word sly, the data server would match the result to the keyword AAAI since it has a higher priority than the keyword set "fox:!sly". Each line beginning with an asterisk after the second line in the data entries contains results that match the given keywords. One of the results is randomly selected when a sentence is matched up to a given keyword. Much of the code to handle these decisions was adapted from Carnegie Mellon University's Duane Fields' Splotch Program.

Human Interaction

There are several options for human interaction with ButlerBot. The interface is shown in Figure 5. A person may simply click on a tabbed pane to view a general schedule of the entire IJCAI conference or a map of the conference building. Another option is to communicate with the robot by manually entering a question into a search field and having ButlerBot search for the answer. Microsoft Agent has been incorporated into ButlerBot's interface so that a butler avatar converses with the user, increasing user-friendliness.

A wireless headset in combination with IBM's ViaVoice speech recognition software allows users to verbally command the robot. The user's verbal command is then automatically entered into the same search field as if the user had manually entered it.

Whether manually entering the data or verbally communicating with the robot, ButlerBot's response is

determined in the same manner described above in the Data Server section.

Yet another way to interact with ButlerBot is through the interactive map displayed in the upper left hand corner of the user interface. Depending on the room the robot is currently in, different red stars are displayed on the map as seen in Figure 5. If a user clicks on a red star, ButlerBot will escort the user to the destination selected.

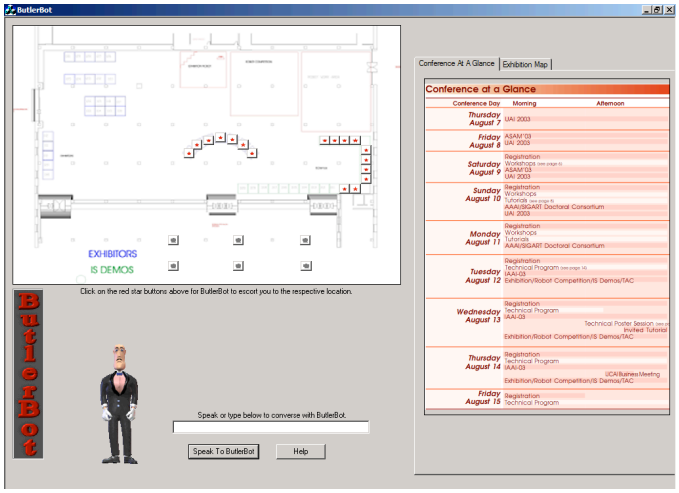

Figure 5: ButlerBot's User Interface

Future Work

In future revisions of the navigation system, ButlerBot will use Dijkstra's shortest path algorithm to automatically determine the best path to take (Dijkstra 1959). This would assist it in taking a more direct route to the destination. In order to prevent a shortest path through an obstruction within the map boundaries, a simple Bentley-Ottmann sweep can be used on the list of line segments composing the polygon shaped obstruction (Bentley and Ottmann 1979).

A Neural Network (NN) algorithm has already been implemented for human activity recognition. The NN distinguishes between different human behaviors including standing, walking, and sitting and thus allows the robot to make a better decision as the various behaviors can easily be integrated into the robot's Finite State Machine as specific states. In future work, motion estimation will also be added to enable real tracking.

A more intelligent conversation server will be created using Neural Network algorithms such that the robot can learn from speaking to people. Another goal is for ButlerBot to remember previous questions and statements entered by the user and the corresponding replies given back to the user to further simulate realistic conversation.

New methods of voice recognition are being investigated as ViaVoice had many problems at the conference including working well for wide ranges of users and it was found to not handle significant background noise gracefully. In addition, the user interface will be further enhanced graphically. While MS Agent provided a very attractive and attention grabbing character to interact with, a larger and more expressive avatar is being researched.

Acknowledgements

A huge thanks to both Amanda Stent and Dimitris Samaras for their guidance, and the College of Engineering and Applied Science for their sponsorship. In addition, this project was sponsored and made possible by Evolution Robotics. Thanks particularly to Jennifer McNally and Hans Dreyer of Evolution Robotics for their wonderful customer service and assistance in technical support.

References

Bentley, J.L., Ottmann, T.: ``Algorithms for Reporting and Counting Geometric Intersections". IEEE Trans. on Computers C 28, 643-647, 1979

Dijkstra, E. "Two Problems in Connexion with Graphs," Numerische Mathematik, 1 (1959), 269-271.

Elgammal, A., Harwood, D., and Davis L., "Non-parametric model for background subtraction" in FRAME-RATE Workshop, IEEE, 1999.

Evolution Robotics, http://www.evolution.com

IBM ViaVoice SDK for Windows, http://www-3.ibm.com/software/speech/dev/

MS Agent, http://www.microsoft.com/msagent/

MySQL, http://www.mysql.com/

Splotch, http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/ai-repository/ai/areas/classics/eliza/splotch/0.html