Control of a Mobile Robot Using Spoken Commands

Jesús Savage, Emmanuel Hernández, Gabriel Vázquez, Adalberto Hernandez (1), Andrey Ronzhin (2)

(1) Laboratory of Biorobotics, University of Mexico, UNAM
savage@servidor.unam.mx
(2) Speech Informatics Group of St. Petersburg Institute for Informatics and Automation (SPIIRAS)
ronzhin@iias.spb.ru

Abstract

There are several architectures that control mobile robots, the architecture that it is proposed in this paper is formed by several layers, see Figure 1, each one having a specific function that in whole control the behavior of the robot. The Virtual and Real Robot (ViRBot) System is used to control the movements of virtual and real mobile robots, and one of the features of this system is that an user can use spoken commands to communicate with the robots. In this paper it is described the Human/Robot Interface module that contains a speech recognition and natural language understanding system used to control the operation of a mobile robot.

Keywords: Speech Recognition, Human/Machine Interfaces, Mobile robots

Introduction

One of the goals to use speech recognition for the control of movements of a mobile robot is for a user to be able to communicate with it using natural language understanding. A user gives a command to the robot and according to what it was asked the robot should act accordingly to do what it was asked. In this paper is presented a speech recognition and natural language understanding system that has been developed between the Laboratory of Intelligent Interfaces and SPIIRAS in the last years. The communication between the robot and a user is done by several means: speech recognition and text provided by keyboard.

1. Mobile Robot Operation

There are several architectures to control mobile robots, this architecture is proposed based on the Aura architecture developed by Ronald Arkin [6,7]. In our approach the mobile robot’s operation it is formed by several layers, see Figure 1, each one having a specific function that in whole control the behavior of the robot.

Each of the layers are described briefly in the following sections, given more details on the Human/Robot interface.

1.1. Virtual Environment

The virtual environment is created using a 3D motor called Sense8 and the robots’ simulator accept the same commands as the real robot. Figure 2 shows a view of the virtual environment.
1.2. Internal Sensors

Each of the sensors is represented by a data structure that has the following elements: sensor's name, sensor's type, sensor's position in the robot and a set of the sensor's values. A B14 robot is used, developed by Real World Interfaces, that it has two wheel encoders and a battery level detector.

1.3. External Sensors

The robot has an array of 8 infrared sensors, 8 sonar sensors, 20 tactile sensors and a video camera.

1.4. Simulator

In the ViRBot system, it is possible to simulate each of sensors' signals according to a physical model of the sensors. Thus, when the system is tested using the virtual robot, the simulator provides each of the sensors' values. A user may include his own simulation algorithms [4].

1.5. Robot's Tasks

The robot is programmed to perform certain tasks, like picking objects and delivering them to another place, during the day at a certain time.

1.6. Human Robot Interfaces

The communication between a robot and a user can be done by several means: speech recognition, text provided by keyboard and control pads. In this paper is presented a speech recognition and natural language understanding system that has been developed between the Laboratory of Intelligent Interfaces and SPIRAS in the last years.

1.7. Perception

One of the main objectives of the perception module is to obtain a symbolic representation of the data coming from the robot's internal and external sensors, from the Human/Robot interface, from the robot's tasks and if chosen from the simulator. The symbolic representation is generated after applying digital signal processing algorithms on the data generated by the sensors. With this symbolic representation a belief is generated that later with the world representation a situation recognition is created.

1.8. World Model

The world model module uses the belief generated by the perception module and together with the information provided by the cartographer and the knowledge representation it generates a situation that needs to be solved.

1.9. Cartographer

For every room of the working environment there is a representation of how each of the rooms are interconnected between them, as well as each of the known obstacles included in them. From the obstacle representation forbidden areas are created, also free areas with landmark points are created. The landmarks form the nodes of a connectivity network that it is used for the planner to find a path from an origin to a destination. Figure 3 shows the forbidden areas of one of the experiments’ rooms as well as the connectivity network or topological map.

![Figure 3. Symbolic and topological representation of one of the experiments' rooms](image)

1.10. Knowledge Representation

An expert system is used to represent the knowledge that the robot has. In an expert system, the knowledge is represented by rules, each one contains the encoded knowledge of an expert, that is, the actions that the robot would do if certain conditions were met. The environment was defined as facts in an expert system, an expert system shell called CLIPS was used, developed by NASA [2].

1.11. Goal Activations

Given a situation recognition, a set of goals is activated in order to solve them.

1.12. Options

Set of hardwired procedures that solve, partially, specific problems.

1.13. Planner

Given a set of goals, the planner module finds a procedure or guide for accomplishing them. This requires searching a space of configurations to find a set of operations that will solve the problem. In the case that the command is to move an object from one place to another, this problem is solved by finding a sequence of operations that leads from an initial state to a goal state. The Robot is able to perform operations like grasping an object, moving itself from on place to
another, etc. Then the objective of planning is to find a sequence of physical operations to achieve the desired goal. These operations can be represented by a state-space graph. This graph is a set of nodes connected together with a set of arcs. The nodes represent states of a space of possible combinations. The arcs help for going from one state to another. One of the tasks that planning solves is how to choose the best set of operations from all of these. If we impose preconditions for going from one state to another the planning problem becomes more tractable. Tanimoto [8] explains that it is often possible to identify sub tasks for which plans have been developed in advance. Then these plans are combined making adjustments to obtain a plan of the task.

In the case that the planner needs to move the robot from one room to another it finds the best sequence of movements between rooms until it reaches the final destination. Inside each room it finds also the best movement path taking into account the know obstacles, that represent some of the objects in the room. The best path is found, among several paths, using the Dijkstra algorithm, that uses the topological maps of the rooms.

1.14. Navigator

When the robot moves from one room to another the planner gives a set of points \((x_1, y_1, x_2, y_2... x_n, y_n)\) and a time \(t\) to visit them, the navigator finds the angle of rotation \(\theta\) a distance \(d_i\) and a speed \(v_i\) to reach each of them. The robot’s position is described, in time \(i\), by \((x_i, y_i, \theta)\) where \(\theta\) represents the angle of the robot with respect of x axis. The system of coordinates \(x’, y’\) represents the new axis of coordinates after a rotation and displacement of the robot. Figure 4 shows these systems of coordinates.

![Figure 4. Coordinate system](image)

1.15. Pilot

The pilot takes the trajectories generated by the navigator and executes them. Basically on each step it has to move the robot a distance \(d_i\) and turn \(\theta_i\) degrees. It checks for unknown obstacles by the planner and tries to avoid them using behavior algorithms, that are based on potential fields.

1.16. Controller

The controller controls the Robot’s motors and reads data coming for the motors and sensors.

We use a mobile robot model B14 fabricated by Real World Interface [1]. It is a cylindrical mobile robot equipped with a wheeled base that allows the robot to move in two axes: translation (movement parallel to the robot head alignment) and rotation (movement perpendicular to the robot head alignment). The robot have motion controllers that control the movements of the robot, it also have odometers to keep track of the position of the robot. Figure 6 shows the robot with a video camera on the top.

![Figure 6. The Robot B14](image)

1.17. Learning

The system can learn to solve new problems by using genetic algorithms, probabilistic methods, Markov chains, etc.

1.18. Behaviors

The behaviors algorithms are use for the navigation of the robot.

1.18.1. Behavior Algorithms Using Potential Fields

Under this idea the robot it is considered as a particle that is under the influence of an artificial potential field \(U\) whose local variations reflects the free space structure and it depends on the obstacles and the goal point that the robot needs to reach [5].
potential field function is defined as the sum of an attraction field that push the robot to the goal and a repulsive field that take it away from the obstacles. The movement planning is done by iterations, in which and artificial force is induced by
\[ F(q) = -\nabla U(q) \]
that forces the robot to move to the direction that the potential field decrees, where \( \nabla \) is the gradient in \( q \) and \( q(x,y) \) represent the coordinates of the robot position.

The potential field is generated by adding the attraction field \( U_{\text{attr}} \) and the repulsive filed \( U_{\text{rep}} \)
\[ U(q) = U_{\text{attr}}(q) + U_{\text{rep}}(q) \]
thus
\[
F(q) = F_{\text{attr}}(q) + F_{\text{rep}}(q)
\]
\[
F_{\text{attr}}(q) = -\nabla U_{\text{attr}}(q)
\]
\[
F_{\text{rep}}(q) = -\nabla U_{\text{rep}}(q)
\]

1.18.2. Repulsive potential field

The goal of the repulsive potential field is to create a force that take away the robot from the obstacles this is obtained using a potential value that tends to infinite in the surface of the obstacle and decreases as the robot goes away from it. The following equation shows a field with the previous characteristics
\[
U_{\text{rep}}(q) = \frac{1}{2} \frac{1}{|q - q_{\text{obstacle}}|^2}
\]
where \( q_{\text{obstacle}} \) represents the coordinates of the obstacle. For several obstacles, the total field potential is the superposition of the individual potential field of each obstacle,
\[
U_{\text{rep}}(q) = \sum_{i=0}^{k} U_{\text{rep}}^i(q)
\]

We used these areas to find the repulsion forces that act on the robot. There are several approaches to find the repulsion force that a forbidden area acts on the robot, one of them is to find the centroid of the forbidden area and consider that all the mass is concentrated in that point.

1.18.3. Attractive potential field

Attractive field potential creates an attraction force through the goal configuration of the robot. It can be considered a parabolic field of the following form
\[
U_{\text{attr}}(q) = \frac{1}{2} \epsilon |q - q_{\text{destination}}|^2
\]
where \( q_{\text{destination}} \) represents the coordinates of the destination.

There are several methods for planning using potential field, one of them is to use the gradient vector to guide the robot from the initial position to the goal. Defining the unitarian vector pointing to the gradient direction
\[ \hat{F}(q) = \frac{F(q)}{||F(q)||} \]
in this way the movement in discrete times is defined by
\[ q_{i+1} = q_i + \delta_t \hat{F}(q) \]

As we can see in Figure 5 the growing structure represents an obstacle that generates a repulsive field, and the goal generates an attractive field in parabolic form close to it and linear far away that it is represented by an inclination in all the generated surface.

![Figure 5. Potential Field created by an obstacle](image)

2. Speech Recognition and Natural Language

To build an agent such as the VirBot, the system should use various levels of understanding and knowledge representation, each based on those below it and each progressively more knowledge intensive. One way to represent meaning is by describing the relationships of objects mentioned in the input sentence. During this process the main event described in the sentence and participants are found. The roles the participants play in the event are determined, as are the conditions under which the event took place. The key verb in the sentence can be used to associate the structure to be filled by the event participants, objects, actions, and the relationship between them. Conceptual Dependency approach uses this type of technique to represent the meaning contained in a sentence.

Conceptual Dependency (CD) is a theory developed by Schank in the early '70s for representing meaning [3]. This technique finds the structure and meaning of a sentence in a single step. CDs are especially useful when there is not a strict sentence grammar. One of the main advantages of CDs is that they allow expert systems to be built which make inferences from a natural language system in the same way humans beings do. CDs facilitate the use of inference rules because many inferences are already contained in the representation itself. The CD representation uses conceptual primitives and not the actual words contained in the sentence.

These primitives represent thoughts, actions, and the relationships between them. Some of the more commonly used CD primitives are:
ATRANS Transfer of an abstract relationship (e.g. give.)

PTRANS Transfer of the physical location of an object (e.g. go.)

ATTEND Focus a sense organ (e.g. point.)

MOVE Movement of a body part by its owner (e.g. kick.)

GRASP Grasping of an object by an actor (e.g. take.)

SPEAK Production of sounds (e.g. say.)

Each primitive represents several verbs which have similar meaning. For instance give, buy, steal, and take have the same representation, i.e., the transference of an object from one entity to another. Each primitive is represented by a set of rules and a data structure containing the following components:

An Actor: The entity that performs the ACT.
An ACT: Performed by the actor, done to an object.
An Object: The entity the action is performed on.
A Direction: The location that an ACT is directed towards.

The user's spoken input is converted into a CD representation using a two step process. The CDs are formed first by finding the main verb in the spoken sentence and choosing the CD primitive associated with that verb. Once the CD primitive has been chosen the other components of the sentence are used to fill the CD structure. For example, in the sentence "Robot, go to the kitchen", when the verb "go" is found a PTRANS structure is issued. PTRANS encodes the transfer of the physical location of an object, has the following representation:

(PTRANS (ACTOR NIL) (OBJECT NIL) 
 (FROM NIL) (TO NIL))

The empty (NIL) slots are filled by finding relevant elements in the sentence. So the actor is the robot, the object is the robot (meaning that the robot is moving itself), and the path is from the living room to the kitchen (assuming the robot was initially in the living room). The final PTRANS representation is:

(PTRANS (ACTOR Robot) (OBJECT Robot) 
 (FROM living-room) (TO kitchen))

Conceptual dependencies can also be used with multimodal input. If the user said "Put the block over there", while pointing at the table top, separate CDs will be generated for the speech and gesture input with empty slots for the unknown information (assuming the block was initially on the floor):

Speech:

(PTRANS (ACTOR User) (OBJECT Block) 
 (FROM Floor) (TO NIL))

Gestures:

(ATTEND (ACTOR User) (OBJECT Hand) 
 (FROM NIL) (TO Table\'^\prime top))

Empty slots can be filled by examining CDs generated by other modalities at the same time, and combining then to form a single representation of the desired command:

(PTRANS (ACTOR User) (OBJECT Block) 
 (FROM Floor) (TO Table\'^\prime top))

CD structures facilitate the inference process, by reducing the large number of possible inputs into a small number of actions. The final CDs encode the users commands to the intelligent agent and they are passed to the perception module.

3. Perception and Conceptual Dependency

With the CD representation a belief is generated that later with the world model a situation recognition is created. The situation recognition needs to be accomplish thus a goal is generated. The planner takes the CD representation, and using the state of the world it generates other CDs and micro-instructions to accomplish the command. The micro-instructions are primitive operations acting directly on the robot.

For example when the user says "Robot, go to the kitchen" the following CD is generated:

(PTRANS (ACTOR Robot) (OBJECT Robot) 
 (FROM Robot\'^\prime s-place) (TO Kitchen))

All the information required for the system to represent the command is contained in the CD. The planner needs to find the best global path between the Robot's place and the Kitchen, thus the following command is issued:

(PLANNER get-best-global-path Robot\'^\prime s-place to Kitchen)

And the answer of the planner is the following:

(best-global-path Robot\'^\prime s-place place_1 place_2 
 ... place_n Kitchen)

Now a new set of PTRANS are generated asking the robot to move to each of the places issued by the planner:
(PTRANS (ACTOR Robot) (OBJECT Robot) (FROM Robot's-place) (TO place_1))
(PTRANS (ACTOR Robot) (OBJECT Robot) (FROM place_1) (TO place_2))

and finally

(PTRANS (ACTOR Robot) (OBJECT Robot) (FROM place_i) (TO place_j))

For each of these PTRANS it is asked to the planner to find the best local path from place_i to place_j:

(PLANNER get-best-local-path place_i to place_j)

And the answer of the planner is the following:

(best-local-path node_1 node_2 ... node_m)

in which each node_i, landmark, is formed by the xi and yi coordinates that the robot needs to reach. With these nodes the navigator finds the trajectories that needs to follow:

(NAVIGATOR get-trajectories node_1 node_2 ... node_m)

And the answer of the navigator is the following:

(trjectories d_1, theta_1 ... d_j, theta_j ... d_m, theta_m)

Where theta_j is the angle that the Robot needs to turn after it moves a distances d_j. The pilot uses these data to move to Robot using behavior algorithms.

A more complex example, such as the user saying "Robot, give the newspaper to the father" is represented as:

(ATRANS (ACTOR Robot) (OBJECT newspaper) (TO father) (FROM newspaper's-place))

The planner needs to do the following to accomplish the requested order, first it needs to command the Robot to go for the object, to pick it up, and to deliver it to the place in which the father is. These actions are represented by the following CDs:

(PTRANS (ACTOR Robot) (OBJECT Robot) (TO newspaper's-place) (FROM Robot's place))

(GRASP (ACTOR Robot) (OBJECT newspaper) (TO Robot's-hand) (FROM newspaper's-place))

(PTRANS (ACTOR Robot) (OBJECT Robot) (TO father's-place) (FROM newspaper's-place))

After these CD primitives are issued some other CDs and micro-commands are issued by the system, until the main CD is accomplish. The system was tested using a keyword input and a continuous speech recognition system to give the commands to a virtual robot.

Insertion words (words incorrectly recognized) in a continuous speech recognition system may cause the process of finding the meaning or obeying a command to make the system fail to perform the operation asked. In the testing set we obtain 80% of insertion errors. That means that only 30% of the spoken sentences would be correct recognized, but by combining standard continuous speech recognition systems with CD representation of the sentences the system was able to overcome this problem and to find a right representation 70% of the time.

4. Conclusions

As we can see by using a good representation of the problem domain through CDs planning becomes a more tractable problem and easier to implement. The most significant contribution in this research is that combined successfully digital signal processing, statistical methods and AI techniques together to enhance a continuous speech recognition system to control a mobile robot.

5. References