Robô Recepcionista Pioneer P5-DX

Pioneer 5
Pedro Isidro - 67220, Diogo Silva - 75136, João Pedrosa - 79833

***Abstract*—Abstract (summary of the work)**

***Index Terms*—Pioneer P5-DX, ROS, autónomo.**

I. Introdução

Os requisitos para do nosso projecto exigiam ter um robô Pioneer P3-DX capaz de executar tarefas modeladas por uma máquina de estados finitos. Essas tarefas incluem o robô ser capaz de mapear, auto-localiza-se e navegar o seu ambiente e deslocar-se para diferentes localizações indicadas através de interação com utilizadores.

A nossa motivação foi ter um robô recepcionista capaz de receber visitantes num edifício, e.g. um bloco de escritórios. O robô seria capaz de mapear o edício *a priori* para futura utilização, de receber um conjunto de coordenadas já conhecidas (e.g. escritórios de certas pessoas) associadas a palavras-chave, de interagir com utilizadores através de síntese e reconhecimento de voz e, por fim, de guiar os mesmos aos seus destinos.

Durante a implementação do projecto simplificámos algumas destas funcionalidades, nomeadamente o mapeamento e o reconhecimento de voz. O mapeamento foi executado apenas no quinto piso da Torre Norte do IST. O reconhecimento de voz foi restringido a um pequenos dicionário para reduzir a complexidade e a necessidade de adaptar modelos acústicos e treinamento.

December 6, 2013

II. Algoritmos e Implementação

A. Mapeamento

To do the Mapping of 5 floor, we use the gmapping.

Hokuyo node:
The Hokuyo node obtain the data of Hokuyo connected to the computer and publishe in topic scan.

Gmapping:
The gmapping get the information in topics tf and scan. The topic tf transforms necessary to relate frames for laser, base, and odometry. The topic scan transforms Laser scans to create the map from. The gmapping give the topic map for creating the map.

Initial Map creating in gmapping

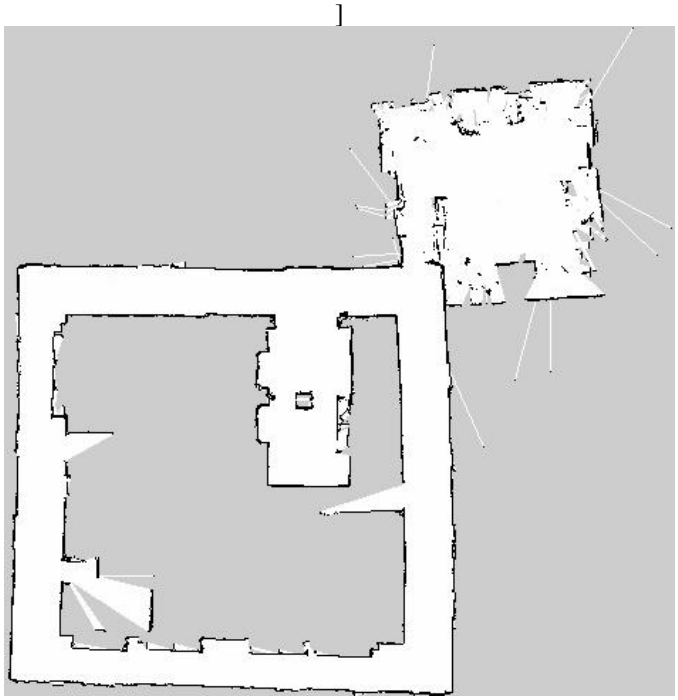

Fig. 1. Initial Map

B. Localização

C. Navegação

For doing the Navigation the used commands gived by the user

Odometry:

The pose have tree parameters: x, y , θ
Algorithm:
We use Rosaria to obtain odometry information for doing the localization in the map, the odometry information obtain in topic pose published by Rosaria. Another thing we use Rosaria is for reading the sonar, give by the topic sonar.

D. Execução do Plano Coordenado

In order to coordinate the robot's actions, an abstract representation of the task to be carried out is needed. The receptionist robot is a sequential system, *i.e.*, it performs one action at a time – either moving to a target location or interacting with a user to acquire one. For such a system,

a State Machine (SM), or Finite State Automaton (FSA) is usually used to specify the control flow through the system.

A SM is implemented using the *smach* package. The task of the robot is divided into three states:

- INITIAL
- TO_GOAL
- GET_GOAL

E. Interacção com o utilizador

A interacção com utilizadores foi feita com recurso a reconhecimento e síntese de voz. Para o primeiro, foram testados duas soluções com características muito distintas. Para o último, foi utilizado um síntetizador de som com capacidade de síntetizar voz a partir de um texto recebido.

1) Reconhecimento de voz: Para compreender como é que o reconhecimento de voz é realizado é necessário introduzir alguns conceitos importantes. A unidade básica da fala é entendida como um *fone*. Contudo, as características acústicas correspondentes a um fone variam conforme o contexto em que esse fone aparece, a pessoa, etc. Devido a estas variações utilizam-se subestados dentro de um fone para melhorar o reconhecimento. Usa-se, também, o contexto em que os fones aparecem, o que se irá traduzir num problema de procura do contexto que melhor se aproxima dos dados recebidos. Os fones constróem sílabas. Dependendo das condições de fala, a mesma sílaba correponde a diferenes fones. Palavras restringem considerávelmente a quantidade de fones que se têm de comparar e quanto menor for o dicionário mais rápido será o reconhecimento.

Uma das soluções testadas foi a livraria *Pocketsphinx* da plataforma Sphinx. Esta livraria, disponível num pacote do ROS com o mesmo nome, está optimizada para portabilidade que é exactamente o que pretendemos na implementação num robô. A plataforma Sphinx utiliza algoritmos extensivamente utilizados em investigação baseados em *Hidden Markov Models* (HMM). Os principais componentes do processo de reconhecimento de voz esão representados na Figura 2.

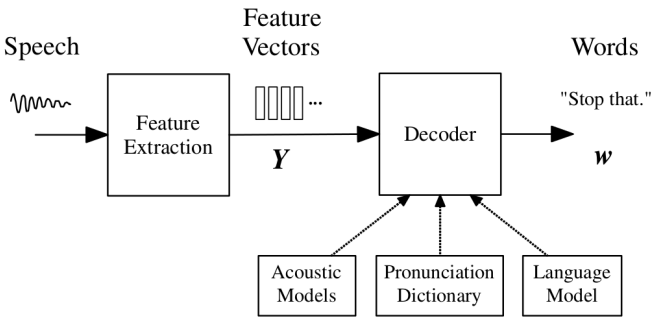

Fig. 2. Principais componentes do processo de reconhecimento de fala.

O audio recebido pelo microfone é convertido em numa sequência de vectores acústicos $Y_{1:T} = y_1, ..., y_T$, num processo denominado de extração de *features*. Os números dos vectores são baseados nas propriedades acústicas da fração correspondente na sequência. Depois desta extração, o descodificador tenta encontrar a sequência de palavras $w_{1:L} = w_1, ..., w_L$ que é mais provável ter estado na origem de Y, ou seja:

$$\hat{w} = arg_w max\{P(w|Y)\} \quad (1)$$

No entanto, $P(w|Y)$ é difícil de modelar directamente, pelo que se utiliza a regra de Bayes para transformar 1 no equivalente:

$$\hat{w} = arg_w max\{p(Y|w)P(w)\} \quad (2)$$

A verossimilhança $p(Y|w)$ é determinada pelo *modelo acústico* e a probabilidade $P(w)$ é determinada pelo *modelo de linguagem*. Para qualquer palavra w, é criado um modelo acústico pela concatenação de modelos fonéticos para criar palavras como elas são definidas num dicionário de pronunciação. Os modelos fonéticos podem ser treinados especificamente para dicionários, ambientes ou contextos específicos. O modelo de lingagem é um modelo em que a probabilidade da palavra N é dependente apenas nos seus $N-1$ antecessores. Os parâmetros no modelo de línguagem são determinados através da contagem da ocorrência das palavras no $corpus$ do texto.

Assim, para o reconhecimento de voz é feito pelo nó *pocketsphinx*, recebendo como argumentos um dicionário de pronunciações e um modelo de linguagem. Para o nosso projecto modificámos o código original para também aceitar outros modelos acústicos que não o original. Desta forma, podemos carregar um modelo que melhor se adapte à nossa utilização.

2) Síntese de voz: A síntese de voz foi feita com recurso ao pacote *sound_play* da *stack audio_common*. O algoritmo trata qualquer som (ficheiro *WAV* ou *OGG* ou texto sintetisado) como algo que pode ser reproduzido ou parado. O estado da reprodução de um som pode ser mudado através da publicação para um tópico específico *robotsound*. O nó *soundplay_node* é o nó responsável pela reprodução do som. Os restantes nós do pacote servem para fornecer o som a ser reproduzido. O nó que nos dá acesso à síntese de voz é o *say* que recebe texto por argumento da linha de comandos ou de um texto. O código deste nó foi alterado por forma a receber texto por subscrição a um tópico. A síntese de voz é feita através do *Festival*.

F. Visualização

The project was developed on the Robot Operating System framework, which is open-source and contains a myriad of different off-the-shelf packages ready for use.

III. Resultados

IV. Conclusão

Appendix A
Title of Appendix A

Appendix A text goes here. desired:

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.