Universidade da Beira Interior

Departamento de Informática

Inteligência Artificial

Professor: Luís Alexandre
Licenciatura em Engenharia Informática

Carlos Esteves nº 37491

1. Introdução

Este projeto foi realizado no âmbito da unidade curricular de Inteligência Artificial com o objetivo de programar um robô capaz de responder a um número de questões pré-definidas pelo docente. Coloca assim em prática os conteúdos lecionados tanto nas aulas práticas assim como nas aulas teóricas, aprofundando deste modo os conhecimentos adquiridos.

2. Desenvolvimento

2.1 Objetivos alcançados

Neste projeto foi dada uma resposta com sucesso ás seguintes perguntas:

1. How many different types of objects did you recognize until now?

2. Which objects were in the room you visited before this one?

6. How many different paths can you take to go from the current room, back to the start room?

7. In what type of room is Mary in?

9. Is the room occupied?

Apesar da tentativa de responder ás restantes perguntas propostas, não foi possível concretizar o esperado.

2.2 Detalhes da implementação

NOTA: É necessário a instalação do networkx para o completo funcionamento do projeto desenvolvido.

2.2.1 Pergunta 1

Para responder a esta pergunta foi criado um array global "obj_count" e na função "callback1" depois de ser feito o split do objeto lido é verificado se este já existe no array e caso não se verifique é adicionado.

```
# Question 1
if(((data.data)[x]).split("_")[0] not in obj_count and ((
data.data)[x]).split("_")[0] != 'person'):
    obj_count.append(((data.data)[x]).split("_")[0])
```

A resposta final é obtida fazendo len() do "obj_count".

2.2.2 Pergunta 2

De forma a dar uma resposta correta a esta pergunta foram criadas três variáveis globais: “roomsobjs” , “room_ant” e “objs”.

O “objs” é um array que é preenchido na função callback1, ou seja, sempre que o robô encontra um objeto novo, se este ainda não está no “objs” fica guardado.

```
# Question2
if(((data.data)[x]) not in objs):
    objs.append(data.data[x])
```

Mais abaixo na mesma função é preenchida a variável roomsobjs, um array de arrays de objetos, da seguinte forma:

```
# Adiciona o objs ao array de quartos roomobjs
if objs[0] != None and objs[0] != '':
    if totobj[room-1] != 0: # se a posição do quarto não
    estiver a 0 faz append
        for y in range(len(objs)):
            if(objs[y] not in roomsobjs[room-1]):
                roomsobjs[room-1].append(objs[y])
                totobj[room-1] += 1
    # Se a posição do quarto estiver a 0 faz insert
    else:
        roomsobjs.insert(room-1,objs)
        totobj[room-1] = totobj[room-1] + len(objs)
        objs =[]
```

Primeiramente é verificado se o ”roomsobjs” está vazio na posição daquele quarto, ou seja, room – 1 devido ao array ter inicio na posição 0. Se estiver é feito insert do array “objs” na posição do respetivo quarto , incrementado a quantidade de objetos daquele quarto e o “objs” é posto a NULL. Se o “roomsobjs” á tiver algo naquela posição é feito o append dos objetos em “objs” que ainda não estiverem guardados.

Para apresentar a resposta final é utilizada a variável “room_ant” que indica o quarto anterior e é atualizada sempre que o robô muda de quarto, fazendo assim print do “roomsobjs[room_ant-1]”.

2.2.3 Pergunta 6

A resposta a esta pergunta foi feita de maneira eficaz utilizando o networkx.

```
# odometry callback
def callback(data):
    global x_ant, y_ant,room,room_ant,objs,G
    x=data.pose.pose.position.x
    y=data.pose.pose.position.y
    # show coordinates only when they change
    if x != x_ant or y != y_ant:
        print " x=%.1f y=%.1f" % (x,y)
    x_ant = x
    y_ant = y

    # verifica se o quarto mudou
    room1 = room
    room_check()
    if(room1 != room):
        G.add_node(room)# adiciona o quarto ao grafo
        G.add_edge(room,room1)# liga dois quartos, arestas
        print "Welcome to room number %d" % room
        objs = []
        room_ant = room1
```

Assim sendo sempre que o robô muda de quarto é acrescentado no grafo G um nodo com o número do quarto e uma ligação entre esse quarto e o anterior(aresta).

Na resposta final é utilizada uma função do nx: all_simple_paths() e passados como parâmetros G, o quarto e None (devido a que neste caso não é preciso limitar a pesquisa da função).

Esta função retorna todos os caminhos possíveis e é apenas necessário fazer len() desta. Também seria possível mostrar os caminhos além de os contar.

```
if(data.data == '6'):
    if(room == 1):
        print "I already am at the start room"
    else:
        paths = nx.all_simple_paths(G,room,1,None)
        print "I can take %d different path(s)" % len(list(paths))
```

2.2.4 Pergunta 7

Nesta questão é inicialmente guardado numa variável global se o robô já encontrou a Maria e o quarto desta.

```
# Question 7
if(((data.data)[x]).split("_")[1] == 'mary'):
    met_mary = 1
    mary_room = room
```

Se já o fez é chamada a função Mary() com o seguinte aspeto:

```
for x in range(len(objects)):

    #print objects[x].split('_')[0]

    if(objects[x].split('_')[0] == 'table'):
        table = table + 1
    if(objects[x].split('_')[0] == 'chair'):
        chair = chair + 1
    if(objects[x].split('_')[0] == 'book'):
        book = book + 1
    if(objects[x].split('_')[0] == 'computer'):
        computer = computer + 1

room_type = 'generic room'

#waiting room
if(chair >= 1 and table == 0 and book == 0 and computer == 0):
    room_type = 'waiting room'

# study room
if(chair > 0 and table > 0 and book > 0 and computer == 0):
    room_type = 'study room'

# computer lab
if(chair > 0 and table > 0 and computer > 0):
    room_type =' computer lab'

#meeting room
if(chair > 0 and table == 1 and book == 0 and computer == 0):
    room_type = 'meeting room'


print "Mary is in a %s" % room_type
chair = 0
table = 0
computer = 0
```

Nesta função é passado para a variável “objects” o conteúdo da “roomsobjs[mary_room-1]”, obtendo-se assim todos os objetos reconhecidos pelo robô até ao momento. De seguida é feito um split verificando o tipo dos objetose incrementado as respetivas variáveis.

No fim é apenas verificado a qual quarto este conjunto de objetos corresponde, de acordo com o enunciado do projeto.

2.2.5 Pergunta 9

Nesta pergunta primeiramente é verificado na função callback1 se o objeto que o robô leu é uma pessoa:

```
# Question 9
if(((data.data)[x]).split("_")[0] == 'person'):
    o_room[room-1] = 1
```

Caso seja é alterado o valor da variável global “o_room”, inicialmente toda a zeros.

No fim é verificado que posições da “o_room” não estão a 0 e é feito print do respetivo quarto.

```
if(data.data == '9'):
    for x in range(11):
        if o_room[x] == 1:
            print "Room %d is occupied" % (x+1)
```

2.2.6 Função extra

Para determinar o quarto em que o robô se encontra foi definida a função room_check() que verifica através das coordenadas atuais na posição do robô e atribui á variável global "room" o respetivo quarto.

```
def room_check():
  global room,x_ant,y_ant

  if y_ant <= 1.5 and x_ant >= -1:
      room = 1
  if y_ant <= 6.5 and y_ant > 1.5 and x_ant >= -1:
      room = 2
  if y_ant > 6.5 and x_ant >= -1:
      room = 3
  if y_ant <= 1.5 and x_ant >=-6 and x_ant < -1:
      room = 4
  if y_ant > 1.5 and x_ant >=-6 and x_ant < -1:
      room = 5
  if y_ant <= 1.5 and x_ant >= -11 and x_ant < -6:
      room = 6
  if y_ant <= 6.5 and y_ant > 1.5 and x_ant >=-11 and x_ant < -6:
      room = 7
  if y_ant > 6.5 and x_ant >=-11 and x_ant < -6:
      room = 8
  if x_ant < -11 and y_ant <= 1.5:
      room = 9
  if x_ant < -11 and y_ant > 1.5 and y_ant<=6.5:
      room = 10
  if x_ant < -11 and y_ant > 6.5:
      room = 11
```

3. Conclusão

Este projeto foi realizado com empenho para que este fosse entregue dentro do prazo e com o máximo de parâmetros propostos resolvidos. Apesar deste ser simples conseguimos ultrapassar todos os problemas que foram surgindo apesar de não ter sido respondido a quatro perguntas.
Ao longo desenvolvimento deste projeto foi conseguido assim aprofundar conhecimentos na matéria da unidade curricular, nomeadamente no uso da linguagem Python.

4. Bibliografia

1. Apontamentos das aulas Praticas e Teóricas da unidade curricular Inteligência Artificial.

2. https://pt.stackoverow.com

3. https://networkx.github.io/

4. http://www.numpy.org/