ROS を用いた移動ロボットの知覚制御

1610581 堀田 大地

2018/11/29

1　目的

本実験では，複数のセンサを統合したロボットシステム開発を行う．加えて，ミドルウェア，ROS，を用いることでロボットシステム開発が容易に行えることを体験する．

2　課題 1

本章では，人の声を認識してロボットがそれに対する受け答えをするようなプログラムを作成した．それを実現するために，Publish として音声を流して，それを Subscribe して，テキストデータに変換してプログラムが応答する実験を行った．また，Roscore は localhost で立ち上げた．ソースコードは Appendix-A に記載した．しかしながら，音声を認識できるのは我々が定義した辞書ファイルに書かれている日本語のみであった．

2.1　手法

まずは，RosClient インスタンスの GetLastMsg メソッドを用いて，SpeechInfo を Subscribe した．その情報の中に我々が喋った言葉が含まれていた．次に，そのデータを文字列へ変換し，我々が定義した言葉に一致するならば，ある言葉を喋らすようなプログラムを作成した．

2.2　実験

辞書に定義されている日本語はしっかり認識されていた．しかしながら，それ以外の言葉はランダムで 1 つ喋っているようであった．この問題を解決するために，色々な対話を試すときは辞書ファイルを更新して Roscore から起動し直した．認識率は高く，ROS を用いることで簡単にメッセージを流すことができた．

3　課題 2

本章では，ロボットが見ているシーンを音声で説明できるようにした．例えば，人が手を挙げているのを認識すると ‘人が手を挙げている ‘と状況を説明することである．

3.1　手法

課題 1 と同様に，Skeltons 情報を Subscribe して，条件にて場合分けを行って，どのポーズを取っているかを評価した．また，課題 1 で使用したソースコードの Subscribe した言葉に対して応答する部分を利用し，音声発話を行った．具体的には，右手を挙げていると ‘われわれのしょうりだ ‘と発話するようなプログラムを作成した．

3.2　実験

右手をあげるとしっかり発話するプログラムを作成できた．Skeltion 情報の場合分けの部分では，頭の位置より手の高さが高いならば手を挙げたとしていたが，実際には頭の高さより大きく手を挙げなければ挙げたと判定されなかった．ROS で容易に Speech 情報だけでなく Skelton 情報を簡単に特別なプログラムを書くこと

なく扱えることが確認できた.

4　課題 3

本章では，音声を用いてロボットを制御できるようにした．例えば，‘まえにすすんで‘と発話すれば，実際にロボットが前進するようなプログラムを作成した.

4.1　手法

まず，課題 1 と同様にして音声情報を文字列データに変換した．そして，受け取りたい言葉を辞書ファイルに定義し，if 文を使って言葉によって次のロボットの動きが変わるようにした．ソースコードを Appendix-3 に記載した.

4.2　実験

課題 1 で辞書に定義している文字列は認識できると評価できていたので，問題なく認識できた．しかしながら，実際はロボットの動作のときに想定の綺麗な動きをすることは稀であった．これは，実際のロボットの動作だとノイズや抵抗があり，制御に対して誤差が生じていたからであると考えられた.

5　課題 4

本章では，独自の複雑な動きをロボットに実行させた．例えば，ジグザクにロボットを動かすことを意味する.

5.1　手法

直進，60 度回転の動作を繰り返して正三角形の軌跡を辿るようなプログラムを実装した．ソースコードを Appendix-4 に記載した.

5.2　実験

実験 3 と同様にノイズや抵抗のせいで綺麗な正三角形を描くような軌跡を描くことはできなかったが，定量的な評価ではあるが，三角形は描けていた．しかしながら，このような一連の動作を記載するようなプログラムが本当に良いのかは評価しなければならない．なぜならば，ロボットの状態を Publish，Subscribe するのが ROS の良い部分なのに，ロボットが動いているかどうかだけで次の動作を実行しているソースコードとなっているからだ.

6　様々なロボット

6.1　対話ロボット

今回のような辞書データに含まれる言葉のみを認識するようなプログラムでは限界がある．それに加えて，全ての言葉を辞書にするという発送もあるかもしれないが，計算量的に現実的でない．近年では，SpeechRecognition も発展している [1] ので事前に深層学習を用いて音声認識モデルを学習することも有用であると思う．また，面白い対話をする取り組みとして真下らの研究がある [2]．これはテーマを与えれば自動的にネタを作る漫才ロボットを提案している．具体的には漫才データベースを保持しており，一番テーマに近いボケをセレクトするというものである．この研究では，データベースなので有限な面白さしかないのが問題である．この問題を解決するために，インターネットのボケてというサービスからテーマとボケを学習して，画像からのキャプション生成をしている吉田らの研究 [3] がある．これはボケてからデータを収集し，いいね数を考慮したデータベースを作成して，画像とボケの関係性を学習しており，いいね数はそのボケの面白さを表す指標として用いるために objectives の係数としている．[3] の実験結果によれば，アンケートで実際の投稿されているボケ，生成したボケ，MS COCO データセットを用いて学習したモデルでキャプション生成 (STAIR caption) を行った結果を評価してもらい，68% の人が前者，23% の人が提案手法，9% の人が STAIR caption が面白いと回答している．私は，面白さとは意外性から来るものだと思っており，既存の手法では人間の意外性までは表現することができていないと評価している．

6.2　質問応答ロボット

このロボットも同様にして，まずは発言を理解することが大事である．しかしながら，現在の技術であればこの問題は問題とはならないと思う．近年，自然言語処理の分野においては，Vaswani らの研究 [4] を初めとして，Attention を用いることで重要性の高い部分に注意するという手法がよく使われている．また，コンピュータビジョンの分野においても画像とその画像に関する質問が与えられたきに，その正しい答えを導くタスクである VQA が盛んである．このような取り組みから質問の本質を探る研究がなされている．しかしながら，人間は質問や説明を行うときに普遍的な知識を用いることが多い．このようなトレーニングプロセスの中では，そのような普遍的な知識を獲得することはできない．そのため，本質とはずれている回答が得られるケースがある．そのため，ある画像やテキストドメインに対しても不変である回答や提案が不可欠であると考えている．

6.3　サッカーロボット

近年のサッカーロボットでは，特にロボカップのサッカーリーグにおいてはチームプレイを行うという手段が取られていると思う．これは強化学習を行いロボット間の連携を学習して，効率よくパスを用いるため点数を獲得しやすいためだと考える．サッカーを用いてロボット間の強調タスクの精度をあげることで，実際にロボット運用のときのコミュニケーション改善に繋がると考えている．また，サッカーをするためには，このようなプレイ面の他にも，ロボットの正確な制御や異常状態にならないことが求められる．

7 Appendix

7.1 課題 1 のソースコード

ソースコード 1 AudioTest.py

```
# coding: shift-jis
import os
os.system("title AudioTest.py")
from RosClientBin import *

client = RosClient()
client.Connect( "localhost" , "11311" )
client.Subscribe[SpeechInfo]()
client.Publish[SpeechOrder]()

while True:
    info = client.GetLastMsg[SpeechInfo]()

    if info:
        print ToString(info.recSpeech), info.isSpeaking
        if ToString(info.recSpeech).find("こんにちわ")!=-1:
            order = SpeechOrder()
            order.utterace = ToBytes("かえりたい")
            client.Send( order )
        elif ToString(info.recSpeech).find("かえりたい")!=-1:
            order = SpeechOrder()
            order.utterace = ToBytes("おれも")
            client.Send(order)
        elif ToString(info.recSpeech).find("まえにすすんで")!=-1:
            order = SpeechOrder()
            order.utterace = ToBytes("まえにすすみます")
            client.Send(order)
```

7.2 課題 2 のソースコード

ソースコード 2 VisionTest.py

```
# coding: shift-jis
import os
os.system("title VisionTest.py")
from RosClientBin import *
import time

client = RosClient()
```

```
8  client.Connect( "localhost" , "11311" )
9  client.Subscribe[Skeltons]()
10 client.Subscribe[Objects]()
11 client.Subscribe[Faces]()
12 client.Subscribe[ColorBlobs]()
13 client.Subscribe[SpeechInfo]()
14
15 while True:
16     time.sleep(5)
17     skeltons = client.GetLastMsg[Skeltons]()
18     if skeltons:
19         if skeltons.data.Count:
20             rightUp = False
21             leftUp = False
22             rightFront = False
23             leftFront = False
24             rightSide = False
25             leftSide = False
26
27             if skeltons.data[0].joints[Skelton.SKEL_RIGHT_HAND].y - skeltons.data[0].
                   joints[Skelton.SKEL_HEAD].y > 0: rightUp = True;
28             if skeltons.data[0].joints[Skelton.SKEL_LEFT_HAND].y - skeltons.data[0].
                   joints[Skelton.SKEL_HEAD].y > 0: leftUp = True;
29
30             if skeltons.data[0].joints[Skelton.SKEL_HEAD].z - skeltons.data[0].joints
                   [Skelton.SKEL_RIGHT_HAND].z > 300: rightFront = True;
31             if skeltons.data[0].joints[Skelton.SKEL_HEAD].z - skeltons.data[0].joints
                   [Skelton.SKEL_LEFT_HAND].z > 300: leftFront = True;
32
33             if skeltons.data[0].joints[Skelton.SKEL_RIGHT_HAND].x - skeltons.data[0].
                   joints[Skelton.SKEL_RIGHT_SHOULDER].x < -300: rightSide = True;
34             if skeltons.data[0].joints[Skelton.SKEL_LEFT_HAND].x - skeltons.data[0].
                   joints[Skelton.SKEL_LEFT_SHOULDER].x > 300: leftSide = True;
35
36             if rightUp:
37                 order = SpeechOrder()
38                 order.utterace = ToBytes("われわれのしょうりだ")
39                 client.Send( order )
40
41             if rightFront: print "前",
42             if rightSide: print "横",
43             print
44
45             if leftUp: print "上",
46             if leftFront: print "前",
47             if leftSide: print "横",
```

```
48              print
49
50      objects = client.GetLastMsg[Objects]()
51      if objects:
52          for i in range( objects.data.Count ):
53              print "物体検出 id:", objects.data[i].id, " pos:",objects.data[i].pos.x,
                    objects.data[i].pos.y,objects.data[i].pos.z
54
55      faces = client.GetLastMsg[Faces]()
56      if faces:
57          for i in range( faces.data.Count ):
58              print "顔検出 id:", faces.data[i].id, " pos:",faces.data[i].pos.x, faces.
                    data[i].pos.y,faces.data[i].pos.z
59
60      color = client.GetLastMsg[ColorBlobs]()
61      if color:
62          for i in range( color.data.Count ):
63              print "色検出 id:" , color.data[i].id, " pos:",color.data[i].pos.x, color
                    .data[i].pos.y, color.data[i].pos.z
```

7.3 課題 3 のソースコード

ソースコード 3　RobotTest.py

```
1  # coding: shift-jis
2  import os
3  os.system("title RobotTest.py")
4  from RosClientBin import *
5  from msvcrt import *
6
7  client = RosClient()
8  client.Connect( "localhost" , "11311" )
9  client.Subscribe[RobotInfo]()
10 client.Subscribe[SpeechInfo]()
11 client.Publish[RobotOrder]()
12 client.Publish[SpeechOrder]()
13 import sys
14 import time
15 info_speech = None
16 while 1:
17     c = 0
18
19     order = RobotOrder()
20     info_speech = client.GetLastMsg[SpeechInfo]()
21     info = client.GetLastMsg[RobotInfo]()
22     order_speech = SpeechOrder()
```

```
23
24      if info_speech:
25          print("info speech is True")
26          if ToString(info_speech.recSpeech).find("まえ")!=-1:
27              order.kind = RobotOrder.ORDER_MOVE_FORWARD;
28              order.data.Add( 0.1 );
29          elif ToString(info_speech.recSpeech).find("うしろ")!=-1:
30              order.kind = RobotOrder.ORDER_MOVE_FORWARD
31              order.data.Add( -0.1 );
32          elif ToString(info_speech.recSpeech).find("ひだり")!=-1:
33              order.kind = RobotOrder.ORDER_ROTATE;
34              order.data.Add( 0.5 );
35          elif ToString(info_speech.recSpeech).find("みぎ")!=-1:
36              order.kind = RobotOrder.ORDER_ROTATE;
37              order.data.Add( -0.5 );
38          elif ToString(info_speech.recSpeech).find("とまれ")!=-1:
39              if info.ismoving:
40                  order.kind = RobotOrder.ORDER_STOP;
41
42          client.Send(order)
```

7.4 課題 4 のソースコード

ソースコード 4　RobotTestDistAngle.py

```
1
2   # coding: shift-jis
3  import sys
4  import os
5  os.system("title RobotTestDistAngle.py")
6  from RosClientBin import *
7  import time
8  import System
9  import msvcrt
10
11
12
13 client = RosClient()
14 client.Connect( "127.0.0.1" , "11311" )
15 client.Publish[RobotOrder]()
16 client.Subscribe[RobotInfo]()
17
18 ###################
19 order = RobotOrder()
20 order.kind = RobotOrder.ORDER_MOVE_DIST
21 order.data.Add(0.5)
```

```
client.Send(order)

while True:
    time.sleep(2)
    info = client.GetLastMsg[RobotInfo]()
    if info:
        if info.ismoving==False:
            break

###################
order = RobotOrder()
order.kind = RobotOrder.ORDER_ROTATE_ANGLE
order.data.Add(3.14 * 2 / 3)
client.Send(order)

while True:
    time.sleep(2)
    info = client.GetLastMsg[RobotInfo]()
    if info:
        if info.ismoving==False:
            break

###################
order = RobotOrder()
order.kind = RobotOrder.ORDER_MOVE_DIST
order.data.Add(0.5)
client.Send(order)

while True:
    time.sleep(2)
    info = client.GetLastMsg[RobotInfo]()
    if info:
        if info.ismoving==False:
            break

###################
order = RobotOrder()
order.kind = RobotOrder.ORDER_ROTATE_ANGLE
order.data.Add(3.14 * 2 / 3)
client.Send(order)

while True:
    time.sleep(2)
    info = client.GetLastMsg[RobotInfo]()
    if info:
        if info.ismoving==False:
```

```
            break
        else:

###################
order = RobotOrder()
order.kind = RobotOrder.ORDER_MOVE_DIST
order.data.Add(0.5)
client.Send(order)

while True:
    time.sleep(2)
    info = client.GetLastMsg[RobotInfo]()
    if info:
        if info.ismoving==False:
            break
        else:

###################
order = RobotOrder()
order.kind = RobotOrder.ORDER_ROTATE_ANGLE
order.data.Add(3.14 * 2 / 3)
client.Send(order)

while True:
    time.sleep(2)
    info = client.GetLastMsg[RobotInfo]()
    if info:
        if info.ismoving==False:
            break
```

参考文献

[1] 久保 陽太郎，音声認識のための深層学習 (連載解説: Deep Learning(深層学習，第 5 回〕)Deep Learning for Speech Recognition(¡Survey Papers¿Deep Learning(5))，人工知能学会，2014.

[2] R. Mashimo, T. Umetani, T. Kitamura and A. Nadamoto, Generating Funny Dialogue between Robots based on Japanese Traditional Comedy Entertainment, In Proc. of the Conference on Interactive Entertainment(IE), 2014.

[3] K. Yoshida, M. Minoguchi, K. Wani, A. Nakamura and H. Kataoka, Neural Joking Machine : Humorous image captioning, In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition of Language & Vision Workshop(CVPR), 2018.

[4] K. Wani.Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin, Attention Is All You Need, arXiv 1706.03762, 2017.