RoboCup@ホームリーグ ハンディマンタスクに向けての開発

2023 年度プロジェクトリサーチ

2023年度プロジェクトリサーチ活動報告書

～RoboCup　@ホームリーグ　ハンディマンタスクに向けての開発～

Y210260　中嶋 洸介
Y210233　清田 樹
Y210300　山本 雄也
Y210229　北川 和真
Y210280　藤原 孝太
Y210282　堀 浩大
Y210243　白井 秋河
Y210266　長島 健留
Y200301　安田 尚平

＜活動目的＞

本プロジェクトではRoboCup@ホームシミュレーションオープンプラットフォームリーグへの出場を目的としてチームを結成する．RoboCup@ホームは日常生活の場で役に立つ仕事を行う実用的なロボットの実現を目指す部門[1)]で，シミュレーションオープンプラットフォームリーグではシミュレーション内の人間（NPC）が何かアクションしたときにシミュレーション環境のロボットがそのアクションに対応した動きをすることで日常生活支援をし，そのロボットの能力を競い合う競技となる．前回大会は2023年5月3日～5月7日に行われ，次の大会は2024年の4月末になる．プロジェクトリサーチでは来年の大会に向けて技術開発を行い，その成果を8月26日，27日のオープンキャンパスで発表することを目標とする．

＜活動計画＞

RoboCup@ホームシミュレーションオープンプラットフォームリーグには，ハンディマンタスク，インタラクティブクリーンナップタスク，ヒューマンナビゲーションタスクという3つのタスクがある．今回のプロジェクトリサーチではハンディマンタスクに必要な技術の開発を計画している．ハンディマンタスクは動いている人や動かない人形，コップなどの物があるシミュレーションの環境内で「寝室にある人形を人に渡して」や「ロビーに行ってテーブルの上にあるコップをゴミ箱に捨てて」といった人間の曖昧な指示がテキストベースで与えられ，それに対してロボットが自律的に対応した動きをするタスクである．このタスクには大きく分けて2つの技術が必要となっている．

一つ目は入力された指示から言葉を抽出し，その言葉に対してロボットが行動する技術である．この技術には形態素解析を用いて入力された指示から単語を抽出し，品詞の判別を行う．そして解析した単語ごとに対応した動作するための情報を入れることでロボットを指示通り行動できるようにする予定である．形態素解析は自然言語で書かれた文を形態素ごとに分解し，それぞれの品詞を判別するものである[2)]．入力された指示を形態素解析し，分解した単語ごとに行動する情報を入れてロボットを動かすことが今回の開発課題である．

二つ目はロボット自身が経路を選択する自律移動の技術である．競技が始まるとプレイヤーはロボットを操作することができず，ロボット自身が経路を選択しなければならないためこの技術が必要になる．この技術にはポテンシャル法と動的物体に対応できないポテンシャル法を補填するためにロボットにセンサーをつけて動的物体の座標位置を得て経路選択する予定である．ポテンシャル法は，自律移動ロボットが目標地点に向かって移動する際に，障害物を避けるための制御アルゴリズムの一つである．この方法は，ロボットが目標に引かれる「引力」ポテンシャルと，障害物から押し戻される「斥力」ポテンシャルを組み合わせて使用する．この引力と斥力を合わせることで，ロボットは障害物を回避しながら目標地点に移動する．ポテンシャル法が活用できる場合は事前にマップデータが分

かっている時である．@ホームリーグ　ハンディマンタスクでは事前にマップデータを得ることができるので，ポテンシャル法を活用することができる．しかし，動いている物体の情報は先に入力できないため，ポテンシャル法の活用は難しい．そのため，センサーの情報から相手の位置を把握し，動的物体に斥力を与えることで経路生成を行う．センサーの情報から相手の位置を把握し，動的物体に斥力を与えるアルゴリズムをポテンシャル法と組み合わせることで経路選択することが開発課題である．

以上，入力された指示を形態素解析し，分解した単語ごとに行動する情報を入れてロボットを動かす開発とポテンシャル法を応用し動作物にも対応した経路生成の開発がプロジェクトリサーチの開発課題であり，ここまでの課題を達成しようと考えている．

＜活動経過＞

本プロジェクトでは形態素解析による指示文の分解とポテンシャル法による経路選択を用いて室内で人々の便利屋をするロボットの作成を行なった．それぞれの技術について私たちの取り組みを説明する．

まず、RoboCup@ホームリーグ　ハンディマンタスクのルールブック[3)]を読み，必要な技術を理解した．ロボットの開発ツールやライブラリが含まれたオープンソースソフトウェアで多くのロボット開発で使われている Robot Operating System(ROS)[4)]を書籍[5)]や web サイト[6)]から勉強し，ロボットを制御する技術の活用方法を学んだ．RoboCup@ホームシミュレーションオープンプラットフォームリーグはシミュレーション上で行う競技であるためシミュレーションの環境構築や使い方について学んだ．そして，二つの必要な技術の開発を試みた．

入力された指示から言葉を抽出し，その言葉に対してロボットが行動する技術では形態素解析による指示文の分解を行った．ロボット自身が経路を選択する自律移動の技術ではポテンシャル法による経路選択を行った．

～形態素解析～

私たちが想定している場面としてはテキストベースの英語での指示がロボットにあった場合である．このテキストベースの指示をロボットが読み取るための手法として，私たちは形態素解析を用いて行なった．

まず，形態素解析がどういったものであるのかを論文[7)]を参考に学習した．そして送られてくる文章（今回の場合は大会に出場したわけではないため，過去の大会のタスク内容から自分たちで設定した文章）に対して形態素解析を行うプログラムを web サイト[8)]を参考にして作成した．以下が作成した形態素解析のプログラムである．このコードでどのような言語処理が行われたのかは以下の＜結果・成果＞の部分で示す．

～ソースコード～

```
import nltk
import json
```

```
# with open('C:¥Users¥itsuk¥third¥project¥EnvironmentInfo06.json') as f:
#         print(f.read())
with open('C:/Users/itsuk/third/project/EnvironmentInfo04.json',encoding="utf-8_sig") as f:
        d = json.load(f)
        s = d.get('taskMessage')

#  最終的に送られてきた文字列をここに入れる。
# s = "Go to the lobby, grasp the rubik's_cube on the white_table and put it in the trash_box_for_recycle."
# s = "Go to the living room, grasp the doll between the low_table and wagon and throw it in the
trash_box_for_bottle_can in the living room."
# s = "Go to the lobby, grasp the doll and give it to me."
# s = "Go to the living room, grasp the ketchup on the white_table and throw it in the
trash_box_for_bottle_can in the living room."

#  指定されたものがある場所、指定されたもの、それをどこに置いたりするのか
#  仮に今は辞書型でやっている。()
place = {"lobby":101, "bedroom":201, "living":301, "kitchen":401} #  場所の変数
#  問題点1：形態素解析でやると living room だけ取り出せないかも？(split と replace で解決できるかも)

thing = {"rubik's_cube":102, "doll":202, "cup":302, "ketchup":402} #  物の変数
#  置く場所、置いてある場所の変数
Dest_list = {"low_table":103, "wagon":203, "white_table":303, "trash_box_for_recycle":403,
"trash_box_for_burnable":503, "trash_box_for_bottle_can":603}

pos = {"on":104,"in":204,"next":304,"close":404,"between":504,"front":604,"under":704} #  前置詞の変数
#  問題点 2、between の時のみ and があるため、後述の"and"による分割ができない。
mot = {"put":105,"throw":205,"give":305} #  動作の変数

# thing_num = 0
# place2_num = 0
# pos_num = 0

#  関数1（between 以外）
def normal(s_thing):
        s_split2 = s_thing.split('and ')
```

```
    s_thing2 = s_split2[0] # "grasp the rubik's_cube on the white_side_table "
    s_place2 = s_split2[1] # "put it in the trash_box_for_recycle."
    thing_nltk = nltk.word_tokenize(s_thing2)
    place2_nltk = nltk.word_tokenize(s_place2)
    # つかむ対象の情報
    # 物の情報取得
    for i in thing_nltk:
        if thing.get(i) != None:
            thing_num = thing.get(i) # この値を、動かすためのプログラムに送信していく。(他の場合も一緒。)
    # 前置詞の情報
    for i in thing_nltk:
        if pos.get(i) != None:
            pos_num = pos.get(i) # この値を、動かすためのプログラムに送信していく。(他の場合も一緒。)
            break
        else:
            pos_num = None
    # どこに置いてあるか？
    for i in thing_nltk:
        if Dest_list.get(i) != None:
            place_num = Dest_list.get(i) # この値を、動かすためのプログラムに送信していく。(他の場合も一緒。)
            break
        else:
            place_num = None

    # どこに何をするかの情報
    # 行動
    for i in place2_nltk:
        if mot.get(i) != None:
            mot_num = mot.get(i)

    if mot_num in mot.values():
        for i in place2_nltk:
            if pos.get(i) != None:
                pos2_num = pos.get(i)
```

```
                    break
                else:
                    pos2_num = None

    for i in place2_nltk:
        if Dest_list.get(i) != None:
            place2_num = Dest_list.get(i)
            break
        else:
            place2_num = None

    for i in place2_nltk:
        if place.get(i) != None:
            room_num = place.get(i)
            break
        # elif thing.get(i) != None:
        #     room_num = thing.get(i)
        #     break
        else:
            room_num=place1_num
    #
    return thing_num, place_num, pos_num, mot_num, pos2_num, place2_num, room_num

# 関数2（between）
def extra(s_thing):
    s_split2 = s_thing.split('and ')
    s_thing2 = s_split2[0] # "grasp the doll between the low_table "
    s_thingPlc = s_split2[1] # "wagon "  ※これはそのまま送る。→最後の空白が邪魔
    s_thingPlc = s_thingPlc.replace(' ','')
    s_place2 = s_split2[2] # "throw it in the trash_box_for_bottle_can in the living room."
    thing_nltk = nltk.word_tokenize(s_thing2)
    place2_nltk = nltk.word_tokenize(s_place2)
    # つかむ対象の情報
    # 物の情報取得
    for i in thing_nltk:
        if thing.get(i) != None:
```

```
                thing_num = thing.get(i) # この値を、動かすためのプログラムに送信していく。(他の場合も一緒。)
    # 前置詞の情報
    for i in thing_nltk:
        if pos.get(i) != None:
            pos_num = pos.get(i) # この値を、動かすためのプログラムに送信していく。(他の場合も一緒。)
        else:
            pos_num = None
    # どこに置いてあるか？
    for i in thing_nltk:
        if Dest_list.get(i) != None:
            place_num = Dest_list.get(i) # この値を、動かすためのプログラムに送信していく。(他の場合も一緒。)
        else:
            place_num = None
    place2_num = Dest_list.get(s_thingPlc)

    # どこに何をするかの情報
    # 行動
    for i in place2_nltk:
        if mot.get(i) != None:
            mot_num = mot.get(i)
    if mot_num in mot.values():
        for i in place2_nltk:
            if pos.get(i) != None:
                pos2_num = pos.get(i)
            else:
                pos2_num = None
    for i in place2_nltk:
        if Dest_list.get(i) != None:
            place3_num = Dest_list.get(i)
        else:
            place3_num = None
    for i in place2_nltk:
        if place.get(i) != None:
            room_num = place.get(i)
```

```
                break
            else:
                room_num=place1_num

    return thing_num, place_num, place2_num, pos_num, mot_num, pos2_num, place3_num, room_num




# 場所指定
s_split1 = s.split(', ')#"Go to the~,"とそれ以降を分ける
s_place = s_split1[0] # "Go to the lobby"
#"部屋断定"
s_place_nltk = nltk.word_tokenize(s_place)
for i in s_place_nltk:
    if place.get(i) != None:
        place1_num = place.get(i) # この値を、動かすためのプログラムに送信していく。(他の場合も一緒。)
#後ろ側の文
s_thing = s_split1[1] # "grasp the rubik's_cube on the white_side_table and put it in the
trash_box_for_recycle."

#
check_thing = nltk.word_tokenize(s_thing) # between の有無判別用
e = 0
for i in check_thing:
    if i == 'between':
        e += 1
        break
if e == 1:
    ex = extra(s_thing)
    # 最終的に送信する形になる。
    print(place1_num) # 最初に行く部屋
    print(ex[0]) # 把持対象
    print(ex[1]) # オブジェクト1(前半)
    print(ex[2]) # オブジェクト2(前半)
    print(ex[3]) # 前置詞(前半)
    print(ex[4]) # 動詞
```

```
        print(ex[5]) # 前置詞（後半）
        print(ex[6]) # オブジェクト（後半）
        print(ex[7]) # 部屋（後半）
    else:
        c = normal(s_thing)
        # 最終的に送信する形になる。
        print(place1_num) # 最初に行く部屋
        print(c[0]) # 把持対象
        print(c[1]) # 前置詞（前半）
        print(c[2]) # オブジェクト（前半）
        print(c[3]) # 動詞
        print(c[4]) # 前置詞（後半）
        print(c[5]) # オブジェクト（後半）
        print(c[6]) # 部屋（後半）
print(type(s))
```

～ポテンシャル法～

私たちは経路選択を行うためにポテンシャル法を使った．まず，ポテンシャル法がどういったものかを理解するために論文[9]を参考に学習した．そしてポテンシャル法による経路選択のプログラムを web サイト[10]を参考に作成した．

以下が作成したポテンシャル法のプログラムである．このコードでどのような経路選択が行われたのかは以下の＜結果・成果＞の部分で示す．

～ソースコード～

```
"""

Potential Field based path planner

author: Atsushi Sakai (@Atsushi_twi)

Ref:
https://www.cs.cmu.edu/~motionplanning/lecture/Chap4-Potential-Field_howie.pdf

"""

from collections import deque
import numpy as np
```

```python
import matplotlib.pyplot as plt

# Parameters
KP = 5.0  # attractive potential gain
ETA = 100.0  # repulsive potential gain
AREA_WIDTH = 30.0  # potential area width [m]
# the number of previous positions used to check oscillations
OSCILLATIONS_DETECTION_LENGTH = 3

show_animation = True


def calc_potential_field(gx, gy, ox, oy, reso, rr, sx, sy):
    minx = min(min(ox), sx, gx) - AREA_WIDTH / 2.0
    miny = min(min(oy), sy, gy) - AREA_WIDTH / 2.0
    maxx = max(max(ox), sx, gx) + AREA_WIDTH / 2.0
    maxy = max(max(oy), sy, gy) + AREA_WIDTH / 2.0
    xw = int(round((maxx - minx) / reso))
    yw = int(round((maxy - miny) / reso))

    # calc each potential
    pmap = [[0.0 for i in range(yw)] for i in range(xw)]

    for ix in range(xw):
        x = ix * reso + minx

        for iy in range(yw):
            y = iy * reso + miny
            ug = calc_attractive_potential(x, y, gx, gy)
            uo = calc_repulsive_potential(x, y, ox, oy, rr)
            uf = ug + uo
            pmap[ix][iy] = uf

    return pmap, minx, miny


def calc_attractive_potential(x, y, gx, gy):
```

```
    return 0.5 * KP * np.hypot(x - gx, y - gy)


def calc_repulsive_potential(x, y, ox, oy, rr):
    # search nearest obstacle
    minid = -1
    dmin = float("inf")
    for i, _ in enumerate(ox):
        d = np.hypot(x - ox[i], y - oy[i])
        if dmin >= d:
            dmin = d
            minid = i

    # calc repulsive potential
    dq = np.hypot(x - ox[minid], y - oy[minid])

    if dq <= rr:
        if dq <= 0.1:
            dq = 0.1

        return 0.5 * ETA * (1.0 / dq - 1.0 / rr) ** 2
    else:
        return 0.0


def get_motion_model():
    # dx, dy
    motion = [[1, 0],
              [0, 1],
              [-1, 0],
              [0, -1],
              [-1, -1],
              [-1, 1],
              [1, -1],
              [1, 1]]

    return motion
```

```
def oscillations_detection(previous_ids, ix, iy):
    previous_ids.append((ix, iy))

    if (len(previous_ids) > OSCILLATIONS_DETECTION_LENGTH):
        previous_ids.popleft()

    # check if contains any duplicates by copying into a set
    previous_ids_set = set()
    for index in previous_ids:
        if index in previous_ids_set:
            return True
        else:
            previous_ids_set.add(index)
    return False


def potential_field_planning(sx, sy, gx, gy, ox, oy, reso, rr):

    # calc potential field
    pmap, minx, miny = calc_potential_field(gx, gy, ox, oy, reso, rr, sx, sy)

    # search path
    d = np.hypot(sx - gx, sy - gy)
    ix = round((sx - minx) / reso)
    iy = round((sy - miny) / reso)
    gix = round((gx - minx) / reso)
    giy = round((gy - miny) / reso)

    if show_animation:
        draw_heatmap(pmap)
        # for stopping simulation with the esc key.
        plt.gcf().canvas.mpl_connect('key_release_event',
                lambda event: [exit(0) if event.key == 'escape' else None])
        plt.plot(ix, iy, "*k")
        plt.plot(gix, giy, "*m")
```

```
rx, ry = [sx], [sy]
motion = get_motion_model()
previous_ids = deque()

while d >= reso:
    minp = float("inf")
    minix, miniy = -1, -1
    for i, _ in enumerate(motion):
        inx = int(ix + motion[i][0])
        iny = int(iy + motion[i][1])
        if inx >= len(pmap) or iny >= len(pmap[0]) or inx < 0 or iny < 0:
            p = float("inf")  # outside area
            print("outside potential!")
        else:
            p = pmap[inx][iny]
        if minp > p:
            minp = p
            minix = inx
            miniy = iny
    ix = minix
    iy = miniy
    xp = ix * reso + minx
    yp = iy * reso + miny
    d = np.hypot(gx - xp, gy - yp)
    rx.append(xp)
    ry.append(yp)

    if (oscillations_detection(previous_ids, ix, iy)):
        print("Oscillation detected at ({},{})!".format(ix, iy))
        break

    if show_animation:
        plt.plot(ix, iy, ".r")
        plt.pause(0.01)
```

```
        print("Goal!!")

        return rx, ry


def draw_heatmap(data):
        data = np.array(data).T
        plt.pcolor(data, vmax=100.0, cmap=plt.cm.Blues)


def main():
        print("potential_field_planning start")

        sx = 0.0    # start x position [m]
        sy = 10.0    # start y positon [m]
        gx = 30.0    # goal x position [m]
        gy = 30.0    # goal y position [m]
        grid_size = 0.5    # potential grid size [m]
        robot_radius = 5.0    # robot radius [m]

        ox = [15.0, 5.0, 20.0, 25.0]    # obstacle x position list [m]
        oy = [25.0, 15.0, 26.0, 25.0]    # obstacle y position list [m]

        if show_animation:
                plt.grid(True)
                plt.axis("equal")

        # path generation
        _, _ = potential_field_planning(
                sx, sy, gx, gy, ox, oy, grid_size, robot_radius)

        if show_animation:
                plt.show()
if __name__ == '__main__':
        print(__file__ + " start!!")
        main()
        print(__file__ + " Done!!")
```

＜各々の活動＞

（中嶋　洸介）：

・毎週行うプロジェクトリサーチのミーティングの司会進行

・目標の RoboCup @ホームシミューレーションリーグのルールブック[2)]を読み、必要な技術の理解

・書籍[5)]や web サイト[6)]で ROS の勉強

・シミュレータ（gazebo,Unity）の勉強

・web サイト[7)]で形態素解析の勉強

・論文[9)]でポテンシャル法の勉強

・web サイト[10)]を参考にポテンシャル法のプログラム作成

・オープンキャンパスでの発表に向けて，購入物の検討

・オープンキャンパスでの発表

・ポスターセッションのポスター作成

・ポスターセッション

（清田　樹）：

・目標の RoboCup @ホームシミューレーションリーグのルールブック[2)]を読み、必要な技術の理解

・書籍[5)]や web サイト[6)]で ROS の勉強

・web サイト[7)]で形態素解析の勉強

・論文[9)]でポテンシャル法の勉強

・json ファイル(“命令文”や”オブジェクトの座標や角度”などの情報が含まれているファイル)の理解

・web サイト[8)]を参考に形態素解析のプログラムの作成

・オープンキャンパスでの発表に使う形態素解析の説明資料作成

・オープンキャンパスでの発表

（山本　雄也）：

・目標の RoboCup @ホームシミューレーションリーグのルールブック[2)]を読み、必要な技術の理解

・書籍[5)]や web サイト[6)]で ROS の勉強

・シミュレータ（gazebo,Unity）の勉強

・web サイト[7)]で形態素解析の勉強

・論文[9]でポテンシャル法の勉強
・Oculus Quest（VR ヘッドセット）のアカウント設定とセットアップ
・オープンキャンパスでの発表に使うロボカップの説明資料作成
・オープンキャンパスでの発表
・ポスターセッション

(北川　和真)：
・目標の RoboCup @ホームシミューレーションリーグのルールブック[2]を読み、必要な技術の理解
・書籍[5]や web サイト[6]で ROS の勉強
・web サイト[7]で形態素解析の勉強
・json ファイル(“命令文”や”オブジェクトの座標や角度”などの情報が含まれているファイル)の理解
・web サイト[8]を参考に形態素解析のプログラムの作成
・オープンキャンパスでの発表に使う形態素解析の説明資料作成
・オープンキャンパスでの発表
・ポスターセッション補助資料調査
・ポスターセッション

(藤原　孝太)：
・目標の RoboCup @ホームシミューレーションリーグのルールブック[2]を読み、必要な技術の理解
・書籍[5]や web サイト[6]で ROS の勉強
・web サイト[7]で形態素解析の勉強
・論文[9]でポテンシャル法の勉強
・過去の RoboCup @ホームシミューレーションリーグのタスク調査
・シミュレータの勉強
・オープンキャンパスでの発表に向けて，購入物の検討
・オープンキャンパスでの発表
・ポスターセッションのポスター作成
・ポスターセッション

(堀　浩大)：
・目標の RoboCup @ホームシミューレーションリーグのルールブック[2]を読み、必要な技術の理解
・書籍[5]や web サイト[6]で ROS の勉強

・論文[9]でポテンシャル法の勉強
・Oculus Quest（VR ヘッドセット）のセットアップ
・シミュレータ（gazebo）の勉強
・オープンキャンパスでの発表
・ポスターセッション

（白井　秋河）：
・目標の RoboCup @ホームシミューレーションリーグのルールブック[2]を読み、必要な技術の理解
・書籍[5]や web サイト[6]で ROS の勉強
・論文[9]でポテンシャル法の勉強
・過去の RoboCup @ホームシミューレーションリーグのタスク調査
・シミュレータの勉強
・オープンキャンパスでの発表に向けて，購入物の検討
・補助申請書の作成
・オープンキャンパスでの発表
・ポスターセッションのポスター作成
・決算報告書の作成

（長島　健留）：
・目標の RoboCup @ホームシミューレーションリーグのルールブック[2]を読み、必要な技術の理解
・書籍[5]や web サイト[6]で ROS の勉強
・論文[9]でポテンシャル法の勉強
・シミュレータ(gazebo)の勉強
・オープンキャンパスでの発表に使うポテンシャル法の説明資料作成
・オープンキャンパスでの発表
・ポスターセッション

＜結果・成果＞
今回，入力された指示を形態素解析し，分解した単語ごとに行動する情報を入れて ロボットを動かす開発とポテンシャル法を応用し動作物にも対応した経路選択の開発という 2 つの開発課題を掲げた.
一つ目の開発では，形態素解析によって文章から単語を抽出し，抽出した単語や部分単語ごとにタグを付けるプログラムをつくることができた．「Go to the living room, grasp the doll and put it on the low_table」という文章では「living」，「doll」，「put」，「on」，「low_table」

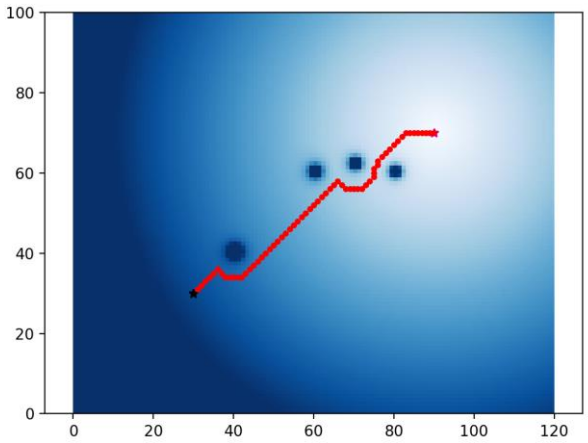
という単語を抽出し，抽出した単語がどういった情報であるかを判別するタグを付けた．しかし形態素解析から分解した単語や部分単語ごとに正確に行動する情報を入れてシミュレーション上の自律移動ロボットを動かす部分ができなかった．以下に形態素解析のプログラムの出力結果を示す．

```
Taskmessage：(Go to the living room, grasp the doll and put it on the low_table.)

場所のリスト：{'lobby': 101, 'bedroom': 201, 'living': 301, 'kitchen': 401}
把持対象のリスト：{"rubik's_cube": 102, 'doll': 202, 'cup': 302, 'ketchup': 402}
置く場所、置いてある場所のリスト：
{'low_table': 103, 'wagon': 203, 'white_table': 303, 'trash_box_for_recycle': 403, 'trash_box_for_burnable': 503, 'trash_box_for_bottle_can': 603}
前置詞のリスト：{'on': 104, 'in': 204, 'next': 304, 'close': 404, 'between': 504, 'front': 604, 'under': 704}
動作のリスト：{'put': 105, 'throw': 205, 'give': 305}

最初に行く部屋の情報：(living),301
把持対象の情報：(doll),202
前置詞（前半）の情報：(None),None
置いてある場所の情報：(None),None
動作内容の情報：(put),105
前置詞（後半）の情報：(on),104
置く場所の情報：(low_table),103
動作を行う部屋の情報：(living),301
```

二つ目の開発では，ポテンシャル法のプログラムを python で書くことができた．シミュレータ上で動かすことができなかったため python 内で変数シミュレートした．Python では黒い点をロボット，赤がロボットの経路，青い点が障害物として，経路選択の手法としてポテンシャル法を用いたロボットが障害物に対してどのような経路を選択するかを確認することができた．しかし，センサーの情報から相手の位置を把握し，動的物体に斥力を与え，ポテンシャル法に組み合わせるという部分ができなかった．以下にポテンシャル法のプログラムの出力結果を示す．



本プロジェクトの目標として RoboCup@ホームシミュレーションオープンプラットフォームリーグで使うロボット技術の開発結果をオープンキャンパスで発表することを掲げた．オープンキャンパスの発表では主に高校生向けに分かりやすく発表を行った．発表としては私たちが出場を目指しているロボカップの詳細，ポテンシャル法の詳細，形態素解析の詳細と形態素解析プログラムによる実際の動作の 3 つの内容についてそれぞれのポスターを

作成し，発表を行った．

ロボカップの発表ではロボカップがどういう競技会であるかを説明し，私たちが出場を目指している RoboCup@ホームシミュレーションオープンプラットフォームリーグの競技内容や課されるタスク内容について説明を行った．

ポテンシャル法の説明では障害物に与えられる「斥力」と目標位置に与えられる「引力」によってどのようにロボットが経路選択を行うのかを説明し，斥力と引力の強さを表現したポテンシャル場やロボットが移動可能な位置や点を指すノードについての説明を行った．また，ポテンシャル法によってロボットが障害物を避けながら目標位置に到達するまでの経路をポスターで表示し，ロボットの動きの軌跡を分かるようにした．

形態素解析の説明では形態素解析の概要を説明し，実際に RoboCup@ホームシミュレーションオープンプラットフォームリーグで想定される指示内容を形態素解析のプログラムで分解した．また，ポスターにて文章がどのように分解され，ロボットに送るかというプログラムの流れを説明した．

来場者にはロボカップの存在や技術内容，開発内容について知ってもらうことができたと感じた．

今回のプロジェクトリサーチの活動では RoboCup @ホームシミューレーションリーグで使えるロボットの作成まではできなかったが，自律移動ロボットに使われる技術について学習を行うことができた点，技術開発の困難さを理解することができた点，プロジェクトを進める上での検討事項や課題に直面しながらも，チームで活動できたという点で私たちにとって大きな糧になったと感じた．

今回の活動を踏まえて，今後行っていく研究活動にも活かしていきたい．

＜参考＞

1)
ロボカップ日本委員会．“シミュレーションオープンプラットフォームリーグ(S-OPL)”．ロボカップ@ホーム．https://www.robocup.or.jp/robocup-athome/s-opl/（参照 2023-06-09）

2)
@Home Simulation(OPL)．“RoboCup@HomeSimulation RoboCup@Home competition in virtual reality ”．Home．https://sites.google.com/view/robocup-athome-sim/home（参照

2023-06-23）

3）
Ledge.ai 編集部．“形態素解析とは｜意味・用途・3 種のツール・ライブラリを解説”．Ledge.ai．2018-06-21．https://ledge.ai/articles/morpho_analysis_japan（参照 2023-06-19）

4）
OSTECH GROUP．“ロボット開発に不可欠なオープンソースソフトウェア「ROS」とは”．2018-12-25．https://solutions.ostechnology.co.jp/blog/ros/（参照 2023-06-16）

5）
Morgan Quigley, Brian Gerkey, William D. Smart 著，河田 卓志 監訳，松田 晃一，福地 正樹，由谷 哲夫 訳，プログラミング ROS :Python によるロボットアプリケーション開発，オライリー・ジャパン, 2017

6）
Crescent．“Tutorials”．ROS.org．2016-08-16．https://wiki.ros.org/ja/ROS/Tutorials（参照 2023-06-16）

7）
田中　省作．“形態素解析ツール：英語と TreeTagger を中心に”．九州大学情報基盤センター広報：学内共同利用版．2002-07．Vol. 2．No. 2．p.108-118.

8）
m_k．“NLTK の使い方をいろいろ調べてみた”．Qiita．2019-09-23．https://qiita.com/m__k/items/ffd3b7774f2fde1083fa（参照 2023-06-26）

9）
彌城 祐亮，江口 和樹，岩崎 聡，山内 由章，中田 昌宏．“ポテンシャル法によるロボット製品の障害物回避技術の開発”．三菱重工技報．新製品・新技術特集．2014．Vol. 51．No. 1．p.40-45

10）
koichi_basebali．“【Python】PythonRobotics の potential_field_planning の理解”．Qiita．2020-04-16．https://qiita.com/koichi_baseball/items/988aa072b8d9fe382e80（参照 2023-

2023-06-23）

3)
Ledge.ai 編集部．“形態素解析とは｜意味・用途・3 種のツール・ライブラリを解説”．Ledge.ai．2018-06-21．https://ledge.ai/articles/morpho_analysis_japan（参照 2023-06-19）

4)
OSTECH GROUP．“ロボット開発に不可欠なオープンソースソフトウェア「ROS」とは”．2018-12-25．https://solutions.ostechnology.co.jp/blog/ros/（参照 2023-06-16）

5)
Morgan Quigley, Brian Gerkey, William D. Smart 著，河田 卓志 監訳，松田 晃一，福地 正樹，由谷 哲夫 訳，プログラミング ROS :Python によるロボットアプリケーション開発，オライリー・ジャパン, 2017

6)
Crescent．“Tutorials”．ROS.org．2016-08-16．https://wiki.ros.org/ja/ROS/Tutorials（参照 2023-06-16）

7)
田中　省作．“形態素解析ツール：英語と TreeTagger を中心に”．九州大学情報基盤センター広報：学内共同利用版．2002-07．Vol. 2．No. 2．p.108-118.

8)
m_k．“NLTK の使い方をいろいろ調べてみた”．Qiita．2019-09-23．https://qiita.com/m__k/items/ffd3b7774f2fde1083fa（参照 2023-06-26）

9)
彌城 祐亮，江口 和樹，岩崎 聡，山内 由章，中田 昌宏．“ポテンシャル法によるロボット製品の障害物回避技術の開発”．三菱重工技報．新製品・新技術特集．2014．Vol. 51．No. 1．p.40-45

10)
koichi_basebali．“【Python】PythonRobotics の potential_field_planning の理解”．Qiita．2020-04-16．https://qiita.com/koichi_baseball/items/988aa072b8d9fe382e80（参照 2023-06-30）