総合演習　６回目

豊橋技術科学大学　松田基

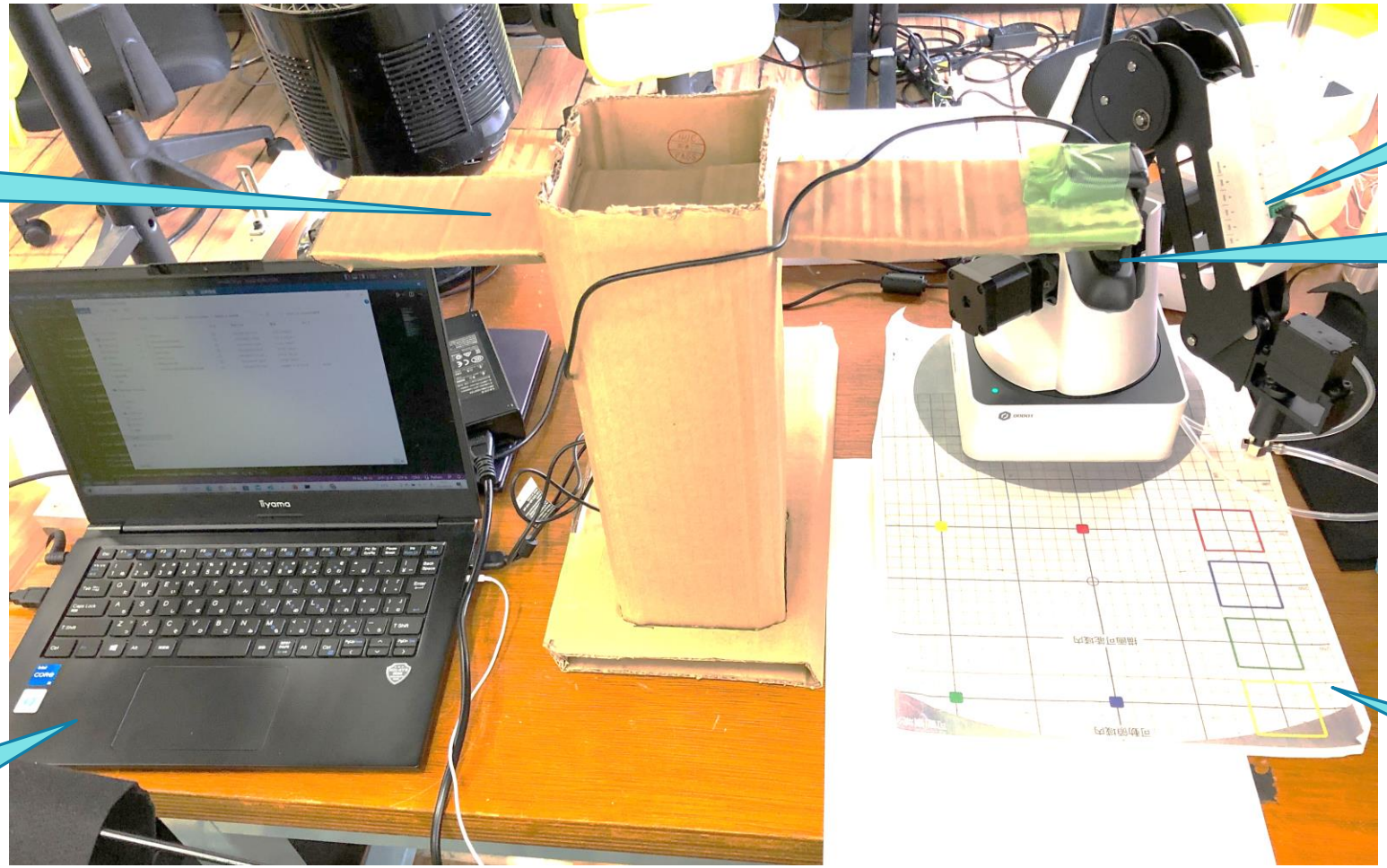
準備（グリッパへの付け替え）15分

1. 吸引カップのネジを緩める
   ネジは<u>完全には抜かない</u>

2. グリッパを差し込む

3. グリッパの<u>**上の**</u>ネジを締める
   <u>かなり強めに締める</u>

チロルチョコの仕分け（実演）15分



6, 7回目の内容

- ロボットアーム（DOBOT）を**Pythonプログラムから制御**する。
- ロボットアームを使って、**指定位置にある対象物を運搬**する。

- AI技術の一つである「ディープラーニング」を使って**AI技術の基本的な考え方**を学ぶ。
- 「ディープラーニング」を使って**画像認識からロボットアームを制御**する。

今日の内容

- ロボットアーム（DOBOT）を**Pythonプログラムから制御**する。
- ロボットアームを使って、**指定位置にある対象物を運搬**する。

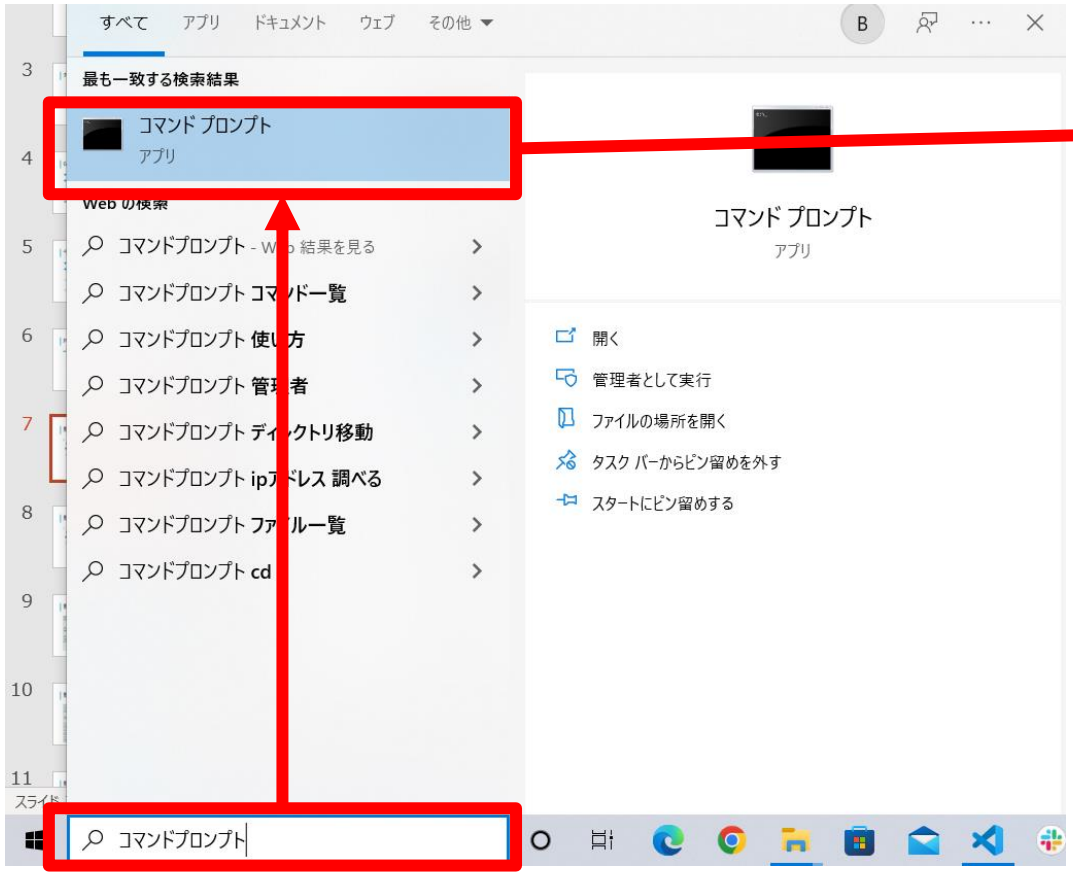
- AI技術の一つである「ディープラーニング」を使って**AI技術の基本的な考え方**を学ぶ。
- 「ディープラーニング」を使って**画像認識からロボットアームを制御**する。

PCのセットアップ（30分）

- 「DOBOT_Magician_Python_Setup.pdf」に従ってPCをセットアップします。

【動作テスト】の説明（15分）

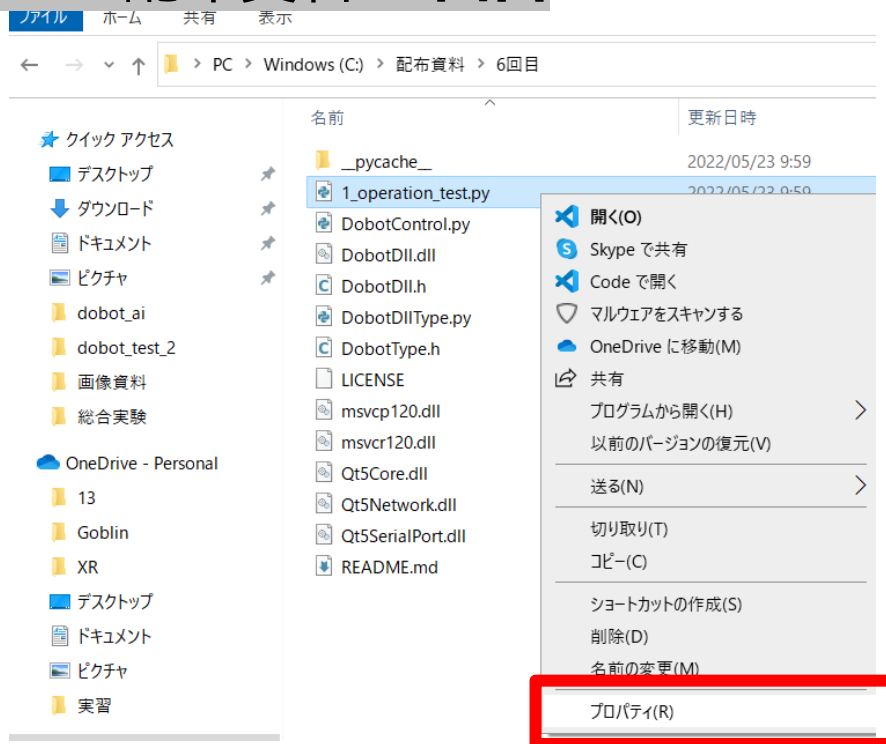
1. コマンドプロンプトを起動



これが出ればOK

【動作テスト】の説明（15分）

2. 「01_operation_test.py」があるフォルダ内に移動する。
   > *cd* **移動するフォルダ名**
   例)> cd *C:¥配布資料¥**6回目***



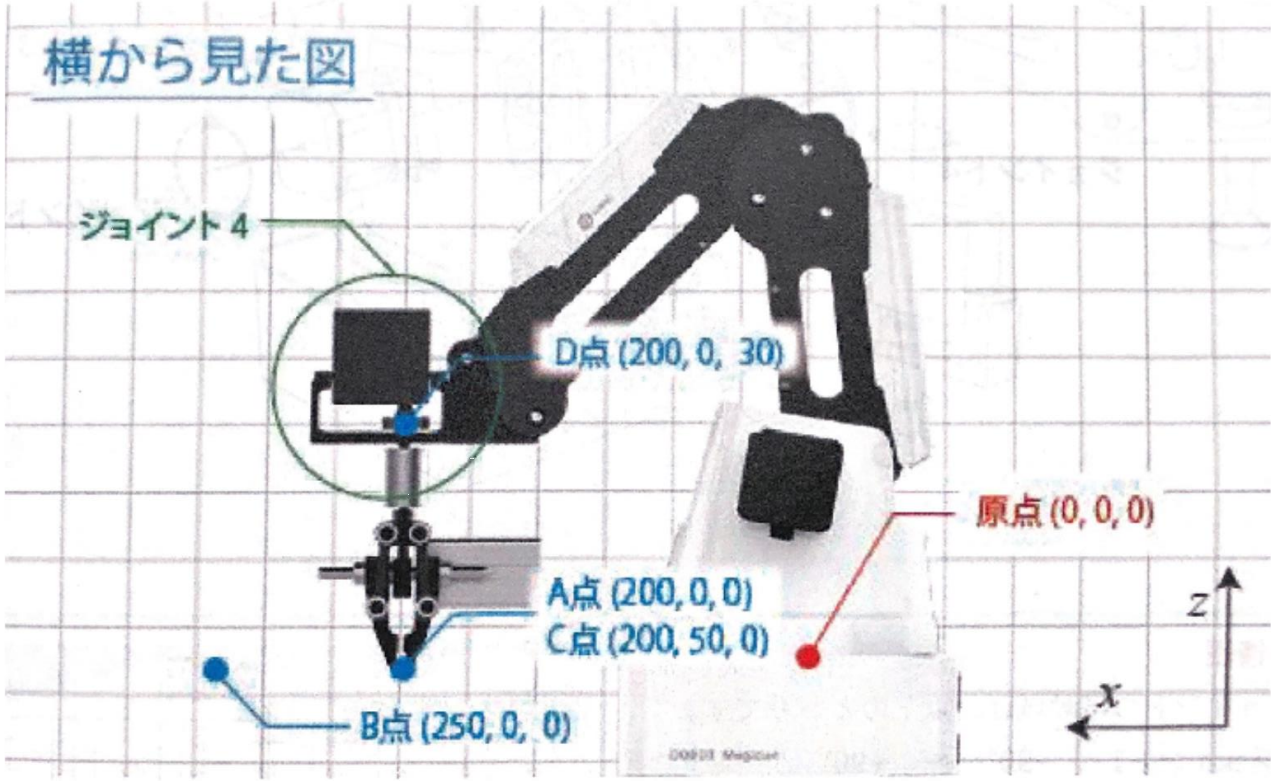
【動作テスト】の説明（15分）

3. DOBOTの電源を入れ、「01_operation_test.py」を実行（一旦待機）
   > python 01_operation_test.py

！注意！

可動域外でDOBOTの電源を入れると、正常に動かない場合があります。

右図のように、手動でアームの先端を少し前に出してから、電源を入れましょう。



【動作テスト】の説明（15分）

01_operation_test.py

```
import DobotDllType as dType
from DobotDllType import PTPMode, JC, DobotConnect


# APIをロードし、DOBOTに接続する
api = dType.load()

# ご使用に応じてCOMポート番号を変更ください
state = dType.ConnectDobot(api, "COM4", 115200)[0]
if not state == DobotConnect.DobotConnect_NoError:
    print("Could not connect to DOBOT")
    exit()
```

ライブラリのインストール

PCによって違う
「DobotStudio」の左上などから
確認できる

【動作テスト】の説明（15分）

01_operation_test.py

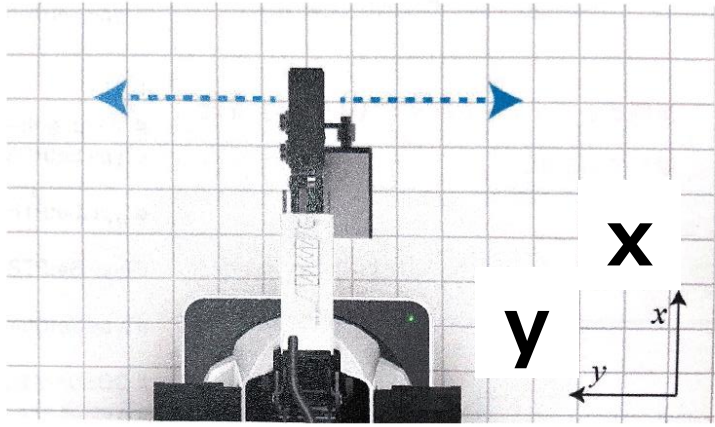
特に変更の必要なし

```
# 初期設定
dType.SetCmdTimeout(api, 3000)
dType.SetQueuedCmdClear(api)
dType.SetQueuedCmdStartExec(api)

deviceName = "DOBOT Magician"
dType.SetDeviceName(api, deviceName)

dType.SetJOGJointParams(api, 50, 50, 50, 50, 50, 50, 50, 50, True)
dType.SetJOGCoordinateParams(api, 50, 50, 50, 50, 50, 50, 50, 50, True)
dType.SetJOGCommonParams(api, 100, 100, True)
dType.SetPTPJointParams(api, 100, 100, 100, 100, 100, 100, 100, 100, True)
dType.SetPTPCoordinateParams(api, 100, 100, 100, 100, True)
dType.SetPTPJumpParams(api, 20, 100, True)
dType.SetPTPCommonParams(api, 30, 30, True)
dType.SetHOMEParams(api, 200, 0, 0, 0, True)
dType.SetEndEffectorParams(api, 59.7 ,0, 0, 0)
```

【動作テスト】の説明（15分）



01_operation_test.py

```
# アームをホームポジションへ移動
dType.SetHOMECmdEx(api, 0, True)

dType.SetPTPCmdEx(api, PTPMode.PTPMOVLXYZMode, 200, 100, 0, 0, True)

dType.SetPTPCmdEx(api, PTPMode.PTPMOVLXYZMode, 200, -100, 0, 0, True)

dType.SetQueuedCmdStopExec(api)#削除

# DOBOT接続解除
dType.SetQueuedCmdStopExec(api)
dType.DisconnectDobot(api)
```

０点合わせ（オフセット）

アームをx=200, y=100, z=0へ移動

【動作テスト】の実行（15分）

◆DOBOTの電源を入れ、「01_operation_test.py」を実行

> python 01_operation_test.py

◆こんなエラー出たら、①DOBOTの電源入れ忘れ ②COM番号の不一致 を確認してください。

```
您用的dll是64位，为了顺利运行，请保证您的python环境也是64位
python环境是：（'64bit', 'WindowsPE'）
>>>>>>>>>>>>>>>>>>>>Can not connected!
Could not connect to DOBOT
```

◆移動させる位置を変えてみましょう。

注）可動域外に無理に動かそうとすると故障の原因となります。

$115<\sqrt{x^2+y^2}<320$（測定マットの可動域内の範囲）で位置を設定してください。

【絶対座標と相対座標】の説明（10分）

◆PTPコマンド
エンドエフェクタ（グリッパなど）をある座標まで移動させたいときに使用するコマンド

モード	移動先の指定方法	モーション
PTPJUMPXYZ	XYZ 絶対座標指定	JUMP
PTPMOVJXYZ		MOVJ
PTPMOVLXYZ		MOVL
PTPMOVJINC	INC 相対座標指定	MOVJ
PTPMOVLINC		MOVL

後で説明

【絶対座標と相対座標】の説明（10分）

◆PTPコマンド
エンドエフェクタ（グリッパなど）をある座標まで移動させたいときに使用するコマンド

```
dType.SetPTPCmdEx(api, PTPMode.PTPMOVLXYZMode, 200, 100, 0, 0, True)
```

02_XYZ.py

ジョイント4の角度の指定

X座標の指定

y座標の指定

z座標の指定

◆絶対座標指定（XYZ）
原点を基準として、移動先の座標（X, Y, Z, R）を指定し、アームを制御。

◆相対座標指定（INC）
現在のエンドエフェクタ位置を基準として、移動先の座標（X, Y, Z, R）を指定し、アームを制御。

※Rはジョイント4の角位置を指定（Z軸方向に回転）

【絶対座標と相対座標】の実行（10分）

◆DOBOTの電源を入れ、「02_XYZ.py」と「03_INC.py」を実行

> python 02_XYZ.py

> python 03_INC.py

◆エラー出たら、①DOBOTの電源入れ忘れ ②COM番号の不一致 を確認してください。

◆「02_XYZ.py」（絶対座標）と「03_INC.py」（相対座標）の違いを見ましょう。

◆移動させる位置を変えてみましょう。測定マットの可動域内の範囲で位置を設定してください。

【移動中の軌道】の説明（10分）

◆PTPコマンド
エンドエフェクタ（グリッパなど）をある座標まで移動させたいときに使用するコマンド

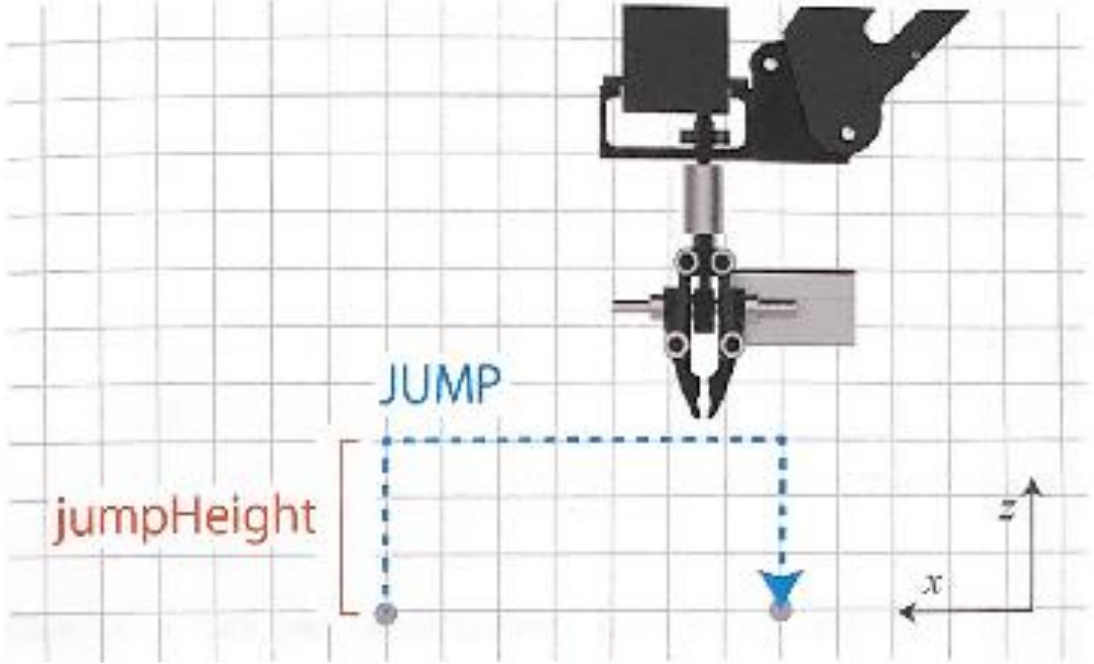
モード	移動先の指定方法	モーション
PTPJUMPXYZ	XYZ 絶対座標指定	JUMP
PTPMOVJXYZ		MOVJ
PTPMOVLXYZ		MOVL
PTPMOVJINC	INC 相対座標指定	MOVJ
PTPMOVLINC		MOVL

ここの説明

【移動中の軌道】の説明（10分）

◆JUMP

　現在位置から指定する高さだけアームを持ち上げて、目標位置の真上に移動し、目標位置にアームを降ろす。

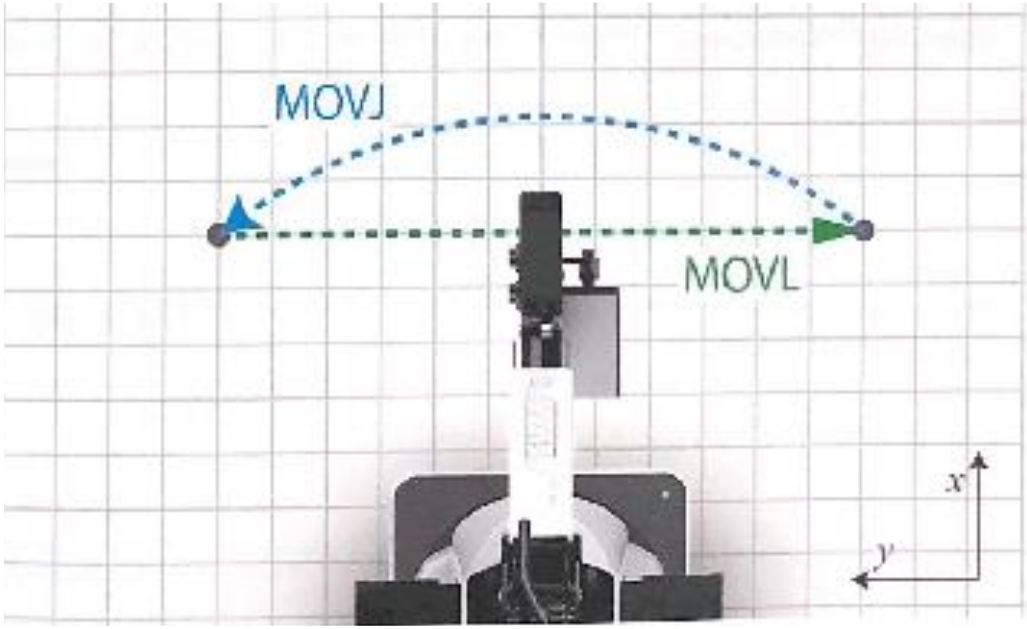


◆MOVL

　現在位置から目標位置まで、XYZ座標上における直線軌道を通るように移動する。

◆MOVL

　現在位置から目標位置に向かって、各ジョイントを回転させて移動する。

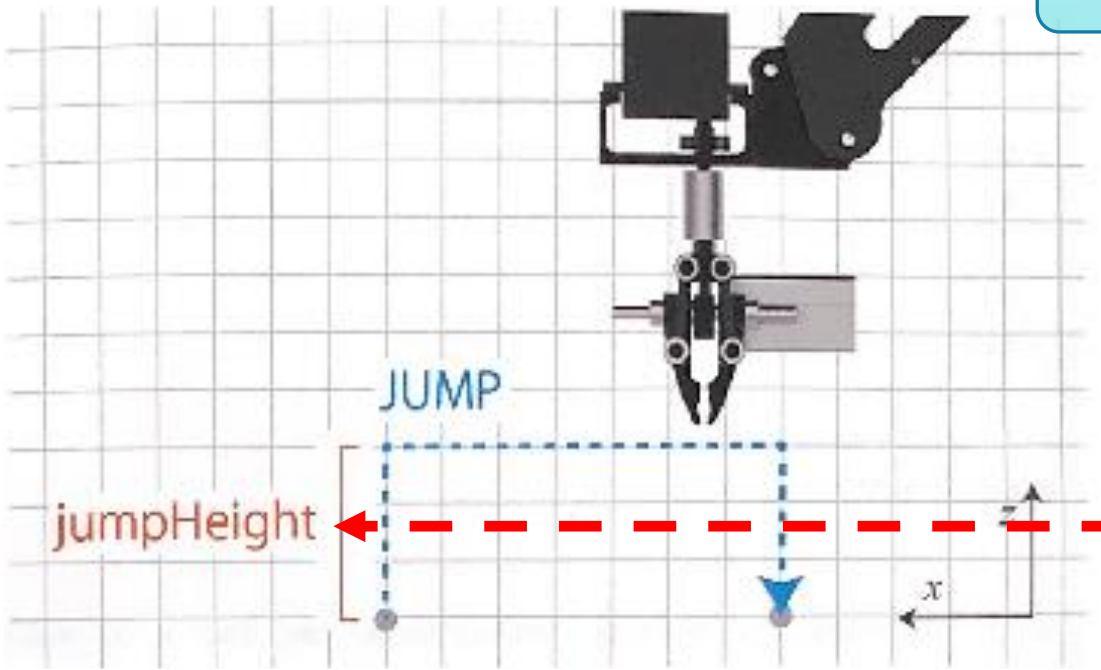


【移動中の軌道】の説明（10分）

04_movement_locus.py

```
dType.SetPTPJumpParams(api, 20, 100, True)
dType.SetPTPCmdEx(api, PTPMode.PTPJUMPXYZMode, 150, -200, 0, 0, True)
```

アームを持ち上げる高さ(jumpHight)を指定



【移動中の軌道】の実行（10分）

◆DOBOTの電源を入れ、「04_movement_locus.py」を実行
> python 04_movement_locus.py

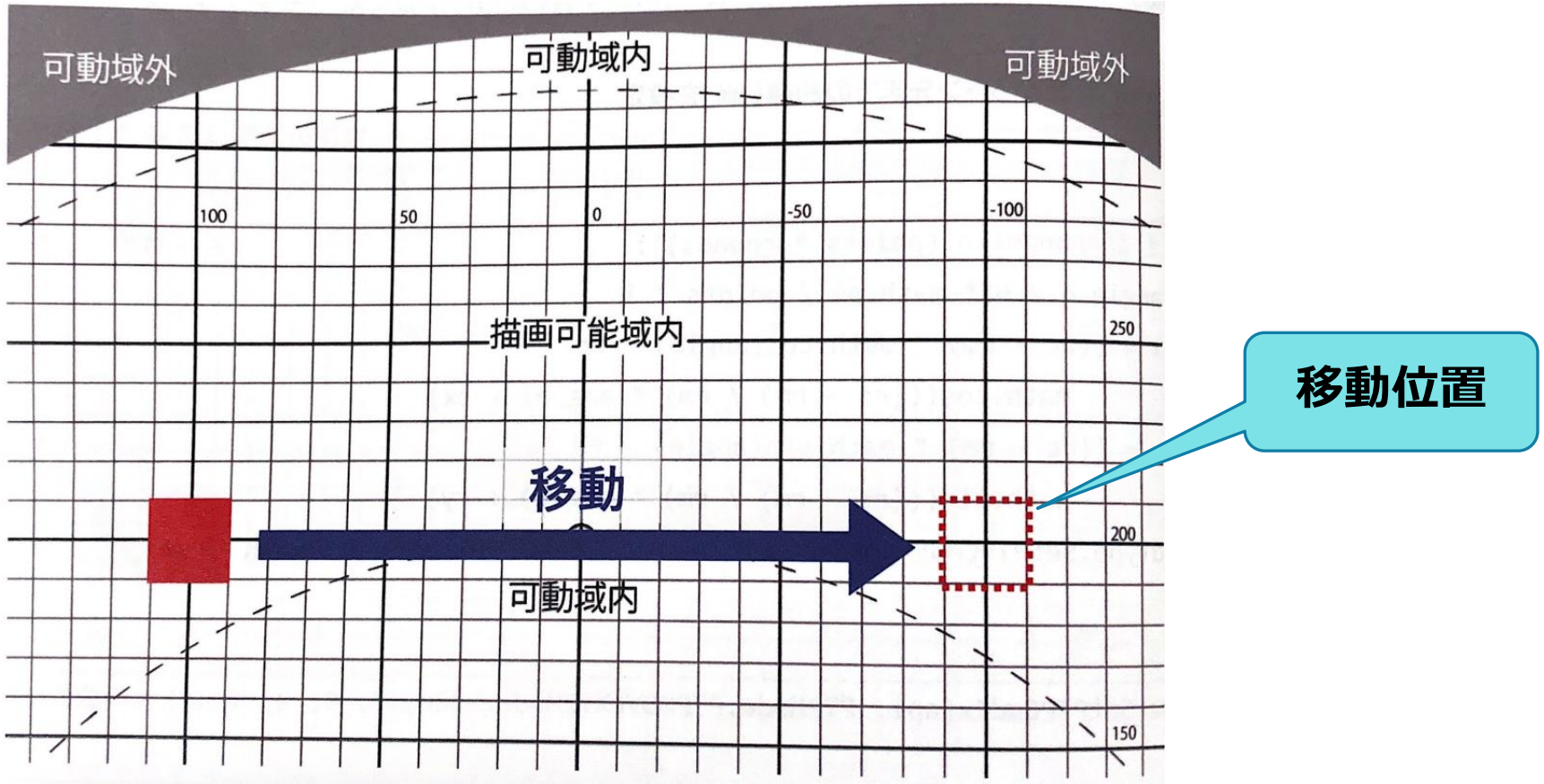
◆エラー出たら、①DOBOTの電源入れ忘れ ②COM番号の不一致 を確認してください。

◆モーションの違いを見て、長所・短所、どんな用途に適しているか、を考えてみましょう。
（レポート：10/100）

◆移動させる位置を変えてみましょう。測定マットの可動域内の範囲で位置を設定してください。

【チロルチョコの運搬】の説明（10分）

◆グリッパを使って、指定座標に置いたチロルチョコを別の指定座標に移動させる。



【チロルチョコの運搬】の説明（10分）

05_carry_TIROL.py

◆グリッパを使って、指定座標に置いたチロルチョコを別の指定座標に移動させる。

```
dType.SetHOMECmdEx(api, 0, True)

dType.SetEndEffectorGripperEx(api, True, False, True)
dType.dSleep(1000)

dType.SetPTPCmdEx(api, PTPMode.PTPMOVJXYZMode, 200, 100, 0, 0, True)
dType.SetPTPCmdEx(api, PTPMode.PTPMOVJXYZMode, 200, 100, -35, 0, True)

dType.SetEndEffectorGripperEx(api, True, True, True)
dType.dSleep(1000)
```

アームをホームポジションへ移動

エアポンプ起動

グリッパを開く

エアポンプ起動

グリッパを閉じる

チョコの真上に移動
アームを降ろす

【チロルチョコの運搬】の説明（10分）

05_carry_TIROL.py

◆グリッパを使って、指定座標に置いたチロルチョコを別の指定座標に移動させる。

```
dType.SetPTPCmdEx(api, PTPMode.PTPMOVJXYZMode, 200, -100, 0, 0, True)
dType.SetPTPCmdEx(api, PTPMode.PTPMOVJXYZMode, 200, -100, -30, 0, True)

dType.SetEndEffectorGripperEx(api, True, False, True)
dType.dSleep(1000)

dType.SetPTPCmdEx(api, PTPMode.PTPMOVJXYZMode, 200, 0, 0, 0, True)
dType.SetEndEffectorGripperEx(api, False, False, True)
```

目的地の真上に移動
アームを降ろす

エアポンプ起動

グリッパを開く

元の位置に戻る

エアポンプ停止

グリッパを閉じる

【チロルチョコの運搬】の実行（10分）

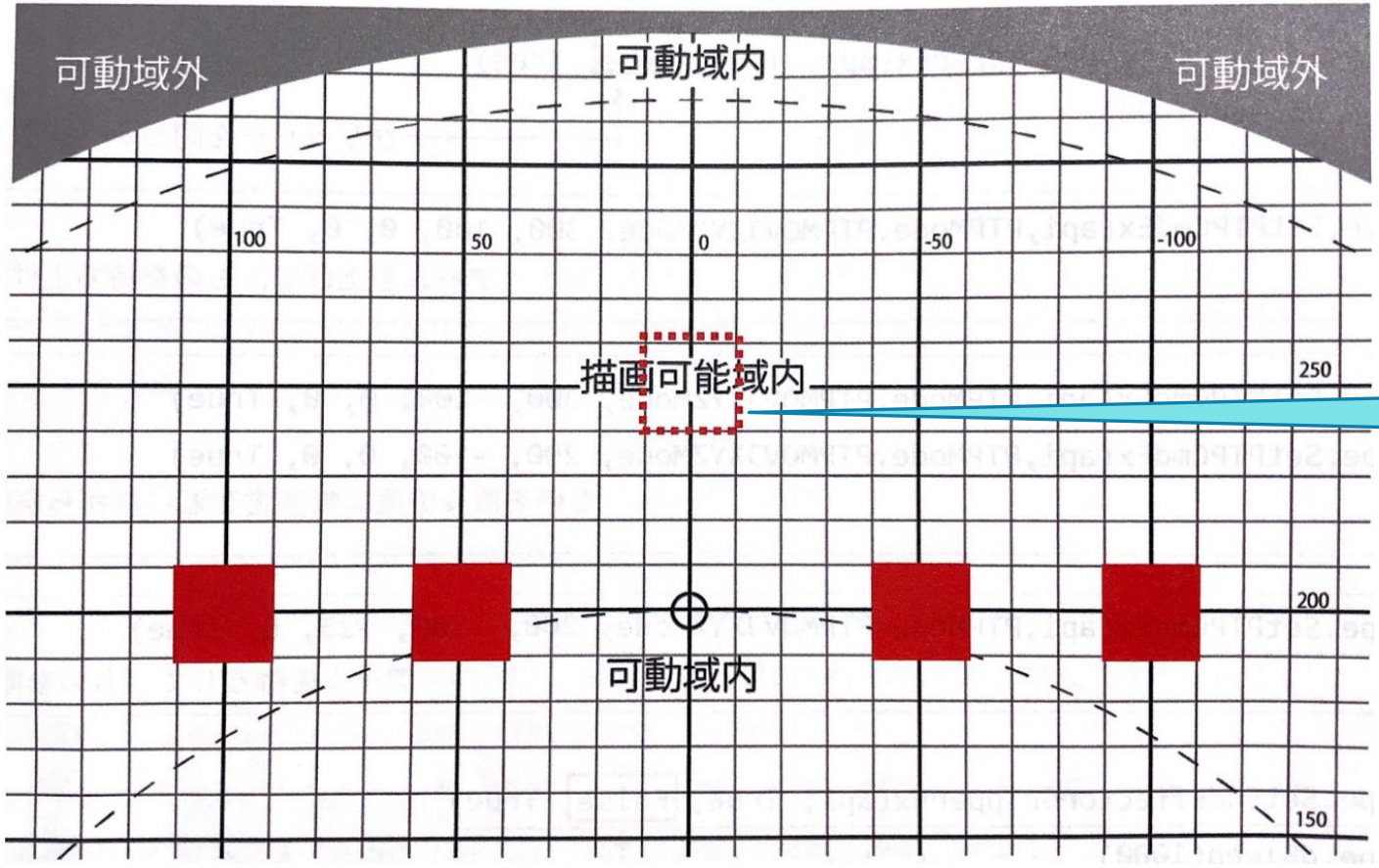
- DOBOTの電源を入れ、「05_carry_TIROL.py」を実行
  > python 05_carry_TIROL.py
- エラー出たら、①DOBOTの電源入れ忘れ ②COM番号の不一致 を確認してください。

- チロルチョコが正しく運搬されることを確認しましょう。
- 移動させる位置を変えてみましょう。測定マットの可動域内の範囲で位置を設定してください。
- 下記コードの赤字の数字(XX)を変え、何が変わるか確認しましょう。（まずは30くらいに）

```
dType.SetPTPCmdEx(api, PTPMode.PTPMOVJXYZMode, 200, 100, 0, XX, True)
dType.SetPTPCmdEx(api, PTPMode.PTPMOVJXYZMode, 200, 100, -35, XX, True)
```

【チロルチョコの積み上げ】の説明（10分）

◆グリッパを使って、複数のチロルチョコを順番に積み上げる。**（レポート：10/100）**



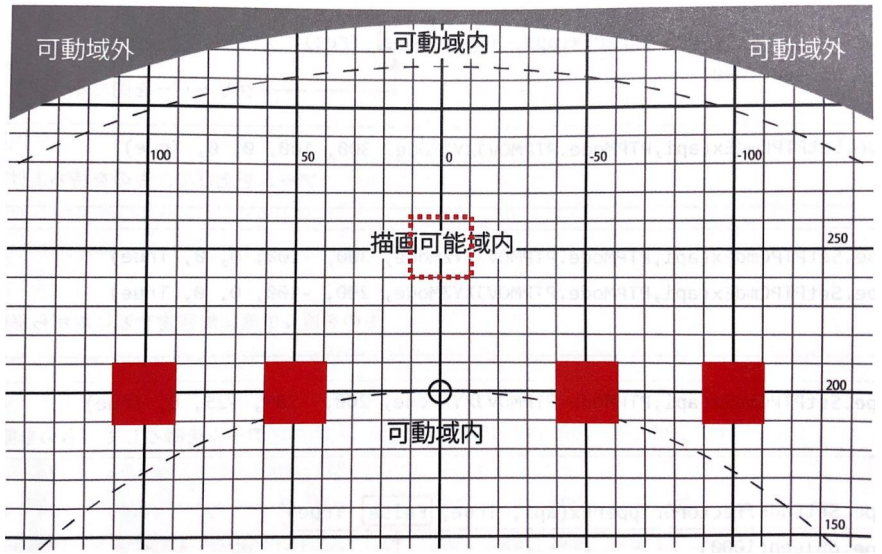
【チロルチョコの積み上げ】の実行（30分）

◆グリッパを使って、複数のチロルチョコを順番に積み上げる。

■仕様

- チロルチョコ１　：（200,　100, -35)
- チロルチョコ２　：（200,　　50, -35)
- チロルチョコ３　：（200,　 -50, -35)
- チロルチョコ４　：（200, -100, -35)
- 積み上げ位置：（250,　　0, -35)

※ 積み上げ位置のZ座標は、
チョコが積みあがるごとに変わります。
チロルチョコの高さは「15」とします。
<u>for文を使わなくてもOK</u>です。



◆pythonのfor文の書き方(例：4回ループ)

```
for i in range(4):
    print(i)
```

出力：
0
1
2
3

【チロルチョコの積み上げ】の実行（30分）

◆グリッパを使って、複数のチロルチョコを順番に積み上げる。

■仕様

- チロルチョコ１　：（200,　100, -35)
- チロルチョコ２　：（200,　　50, -35)
- チロルチョコ３　：（200,　 -50, -35)
- チロルチョコ４　：（200, -100, -35)
- 積み上げ位置：（250,　　0, -35)

※ 積み上げ位置のZ座標は、
チョコが積みあがるごとに変わります。
チロルチョコの高さは「15」とします。

◆pythonのfor文の書き方(例：4回ループ)

```
for i in range(4):
    print(i)
```

出力：
0
1
2
3

◆ヒント

① アームをホームポジションへ移動
② グリッパを開いて起動
③ チョコの真上にアームを移動
④ アームを降ろす
⑤ グリッパで掴む
⑥ アームを真上に上げる
⑦ 積み上げ位置の真上にアームを移動
⑧ 積み上げ位置にアームを降ろす
⑨ グリッパを開く
⑩ アームを真上に上げる
⑪ グリッパを開いて停止

4回繰り返し

【チロルチョコの積み上げ】の実行（30分）

◆グリッパを使って、複数のチロルチョコを順番に積み上げる。

■仕様

- チロルチョコ１　：（200,　100, -35)
- チロルチョコ２　：（200,　　50, -35)
- チロルチョコ３　：（200,　 -50, -35)
- チロルチョコ４　：（200, -100, -35)
- 積み上げ位置：（250,　　0, -35)

※ 積み上げ位置のZ座標は、
チョコが積みあがるごとに変わります。
チロルチョコの高さは「15」とします。

◆pythonのfor文の書き方(例：4回ループ)

```
for i in range(4):
    print(i)
```

出力：
0
1
2
3

◆ヒント

① アームをホームポジションへ移動
② グリッパを開いて起動
③ チョコの真上にアームを移動
④ アームを降ろす
⑤ グリッパで掴む
⑥ アームを真上に上げる
⑦ 積み上げ位置の真上にアームを移動
⑧ **積み上げ位置にアームを降ろす**
⑨ グリッパを開く
⑩ アームを真上に上げる
⑪ グリッパを開いて停止

4回繰り返し

毎回zに15を足す

次回の説明（5分）

- ロボットアーム（DOBOT）を**Pythonプログラムから制御**する。
- ロボットアームを使って、**指定位置にある対象物を運搬**する。

- AI技術の一つである「ディープラーニング」を使って**AI技術の基本的な考え方**を学ぶ。
- 「ディープラーニング」を使って**画像認識からロボットアームを制御**する。

次回の説明（5分）

◆WEBカメラから取得した画像から、チロルチョコの種類ごとにアームで仕分けする。

◆**最後に自身で課題を設定してもらい、その解決に取り組みます。**（75分予定）

- チロルチョコ以外の対象物の仕分け（例えばペットボトルのキャップなど）
- エンドエフェクタの変更（吸引カップ→グリッパ）
- 認識しづらいチロルチョコ（コーヒー味）の認識
- より多い種類のチロルチョコの仕分け

など、独自の課題を設定してもらいます。（上記から選んでも問題ありません）

次回の説明（5分）

！お願い！
対象物を変更する場合
事前に自身で用意してきてください
※必須ではありません

◆WEBカメラから取得した画像から、チロルチョコの種類ごとにアームで仕分けする。

◆**最後に自身で課題を設定してもらい、その解決に取り組みます。**（75分予定）

- チロルチョコ以外の対象物の仕分け（例えばペットボトルのキャップなど）
- エンドエフェクタの変更（吸引カップ→グリッパ）
- 認識しづらいチロルチョコ（コーヒー味）の認識
- より多い種類のチロルチョコの仕分け

など、独自の課題を設定してもらいます。（上記から選んでも問題ありません）

□望ましい対象物

吸引カップ☞**表面が平なもの**　グリッパ☞**挟む位置の幅が2.8[cm]以下のもの**

(+α)同じ対象物でも絵柄が若干異なるもの（例えばチロルチョコのミルク味は絵柄が10パターン）

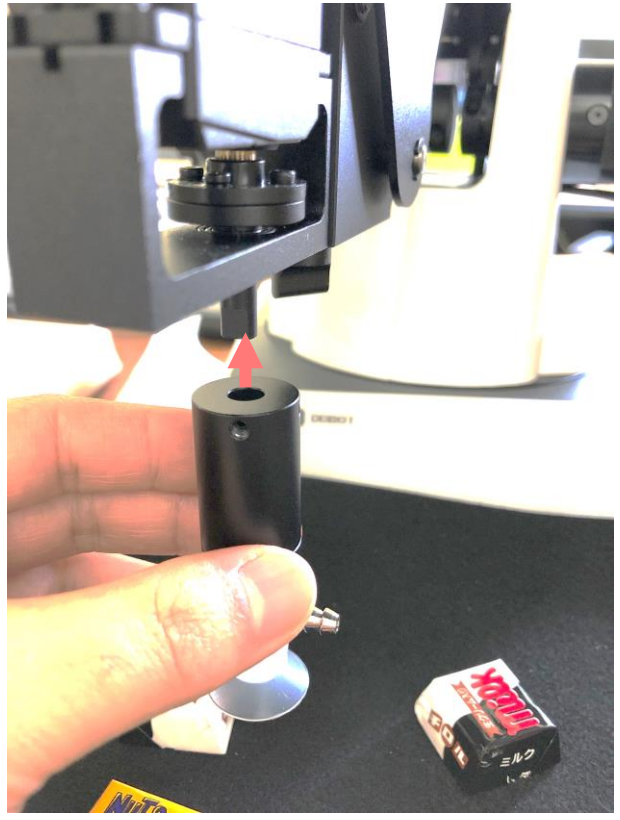
レポートについて

- 【移動中の軌道】の違い（JUMP、MOVJ、MOVL）の長所・短所（6/100）と、それぞれどんな用途に適しているか（4/100）が書かれている。**（計10/100）**
- 【チロルチョコの積み上げ】のプログラムが完成している。**（10/100）**
- AI技術の一つであるニューラルネットワークおよびディープラーニングの基本的な考え方を理解できている。**（10/100）**
- 画像認識によるロボットアーム制御について、独自の課題を設定（10/100）し、プログラムやデータセット等を工夫した（10/100）うえで、課題を解決するプログラムが完成（10/100）しており、結果・考察（10/100）が記されている。
  また、他の学習者（最低1人以上）の、課題設定（10/100）と、工夫した点（10/100）、結果からの考察（10/100）が示されている。**（計70/100）**

片付け（カップへの付け替え）（15分）

1. グリッパのネジを緩める
<u>上のネジだけ</u>

2. 吸引カップを差し込む



3. 吸引カップのネジを緩める
ネジは２つ

参考資料

- Afrel, 【教材】DOBOT AI×画像認識×ロボットアーム制御