# ANRO

## Laboratorium 1

Grupa pierwsza

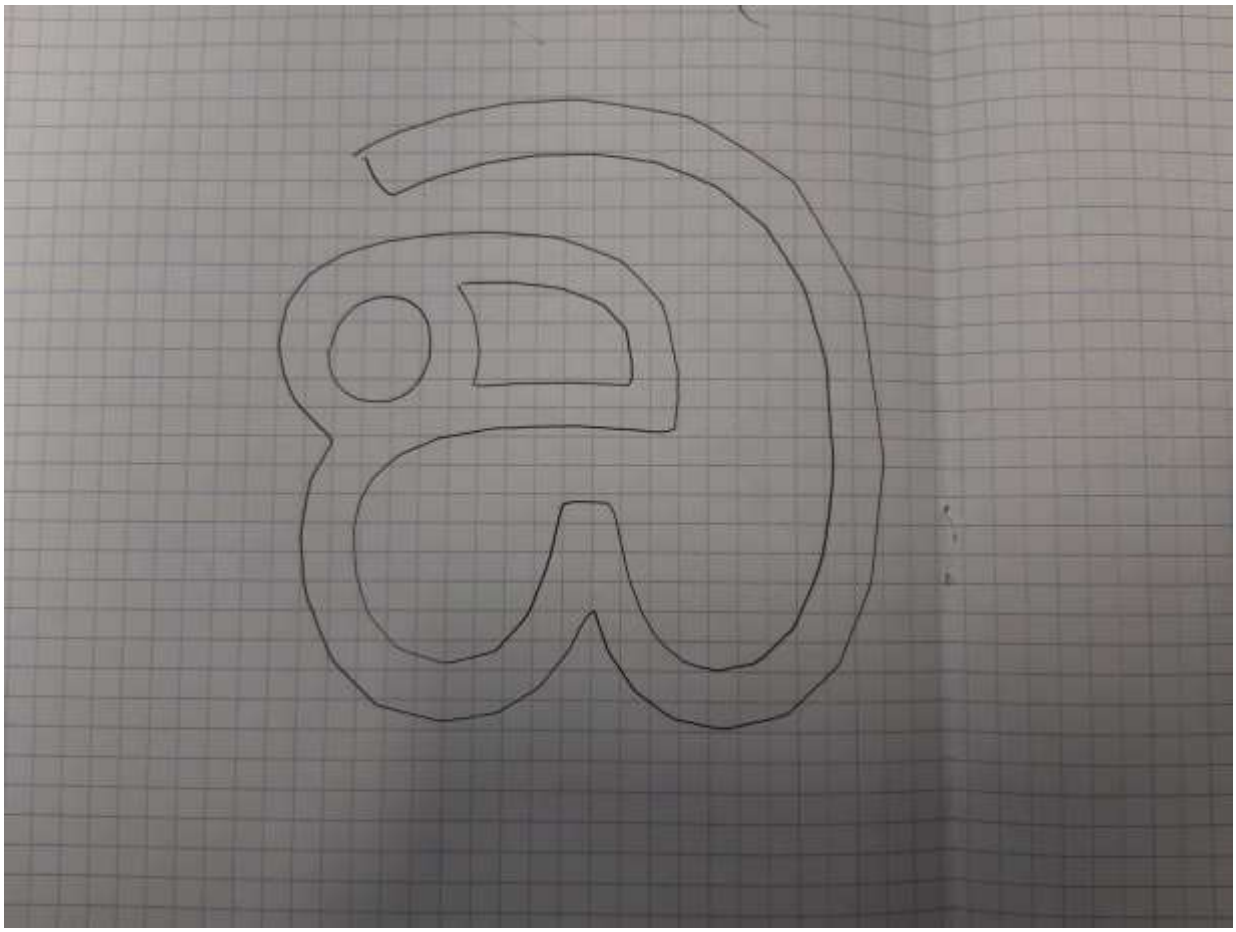
Robot nr 5

Zespół:

Michał Kwarciński

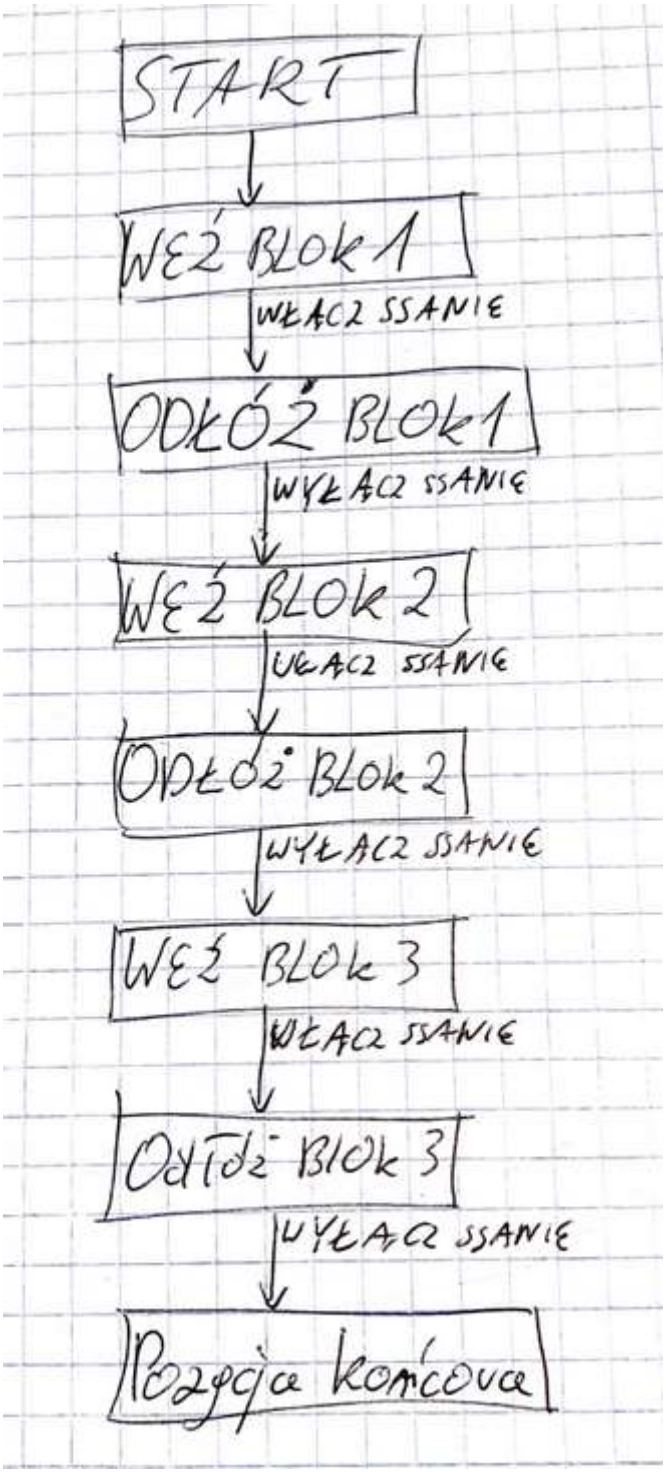
Kacper Marchlewicz

Pracę zaczęliśmy od przetestowania umiejętności rysowania robota. Rysunki wykonywał dość sprawnie, lecz przy łukach widoczne są niedokładności, spowodowane było to odcinkową linearyzacją łuków – powstała krzywa łamana odcinkami prosta.



Następnie postanowiliśmy wykonać wieżę z klocków. Wykorzystaliśmy do tego wszystkie dostępne trzy sposoby programowania. Na papierowej kartce zaznaczyliśmy punkty orientacyjne, które pomagały zobaczyć, gdzie miały znajdywać się klocki i wieża.

Prosty opis struktury programów:




START – robot znajduję się w pozycji startowej

WEŹ BLOK 1 – robot włączając ssanie chwyta klocek 1

ODŁÓŻ BLOK 1 – robot przemieszcza klocek na odpowiednie miejsce i odkłada go wyłączając ssanie

WEŹ BLOK 2 – robot włączając ssanie chwyta klocek 2

ODŁÓŻ BLOK 2 – robot przemieszcza klocek na odpowiednie miejsce i odkłada go wyłączając ssanie

WEŹ BLOK 3 – robot włączając ssanie chwyta klocek 3

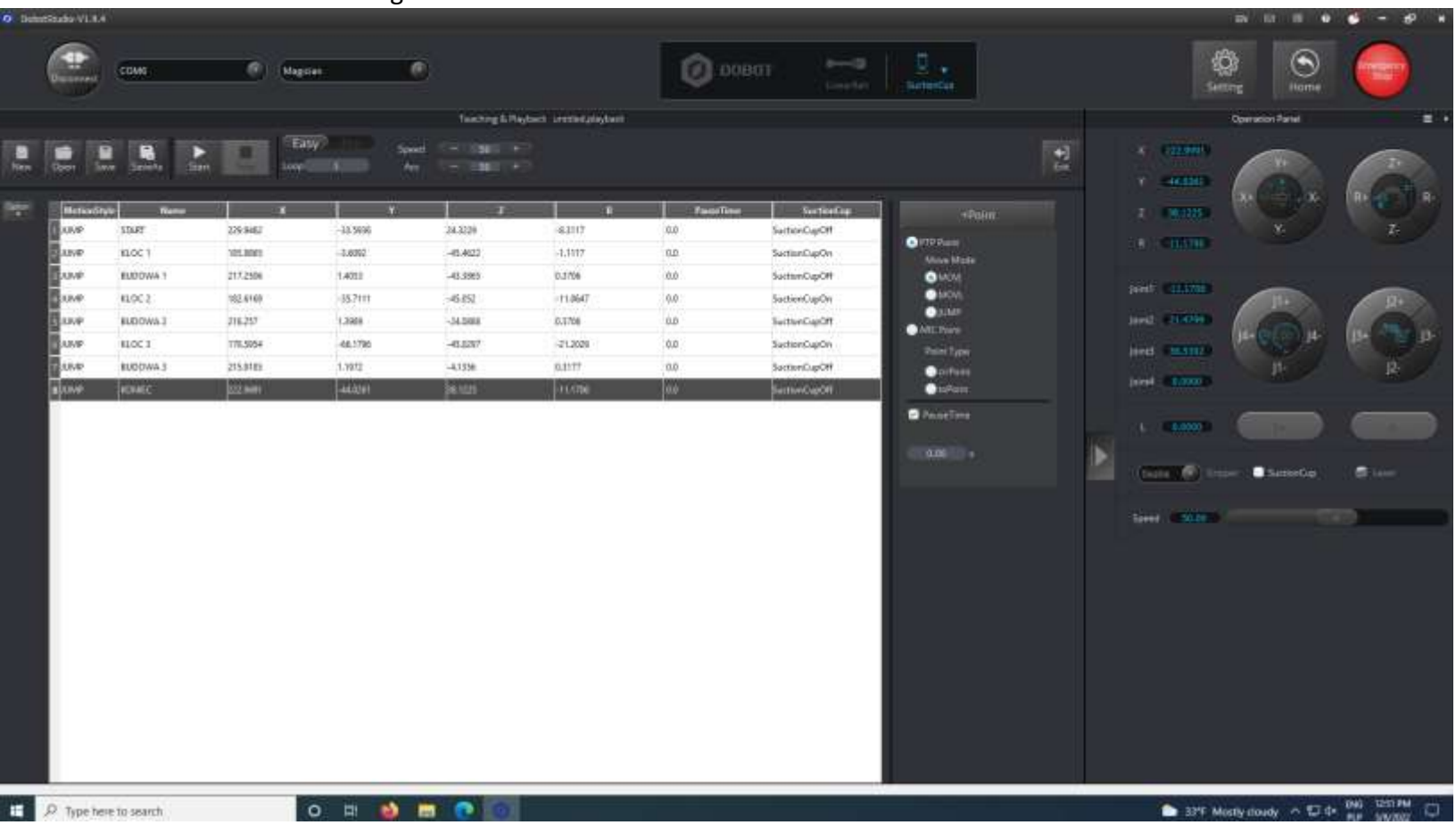
ODŁÓŻ BLOK 3 – robot przemieszcza klocek na odpowiednie miejsce i odkłada go wyłączając ssanie

POZYCJA KOŃCOWA – robot przemieszcza ramię na pozycję końcową

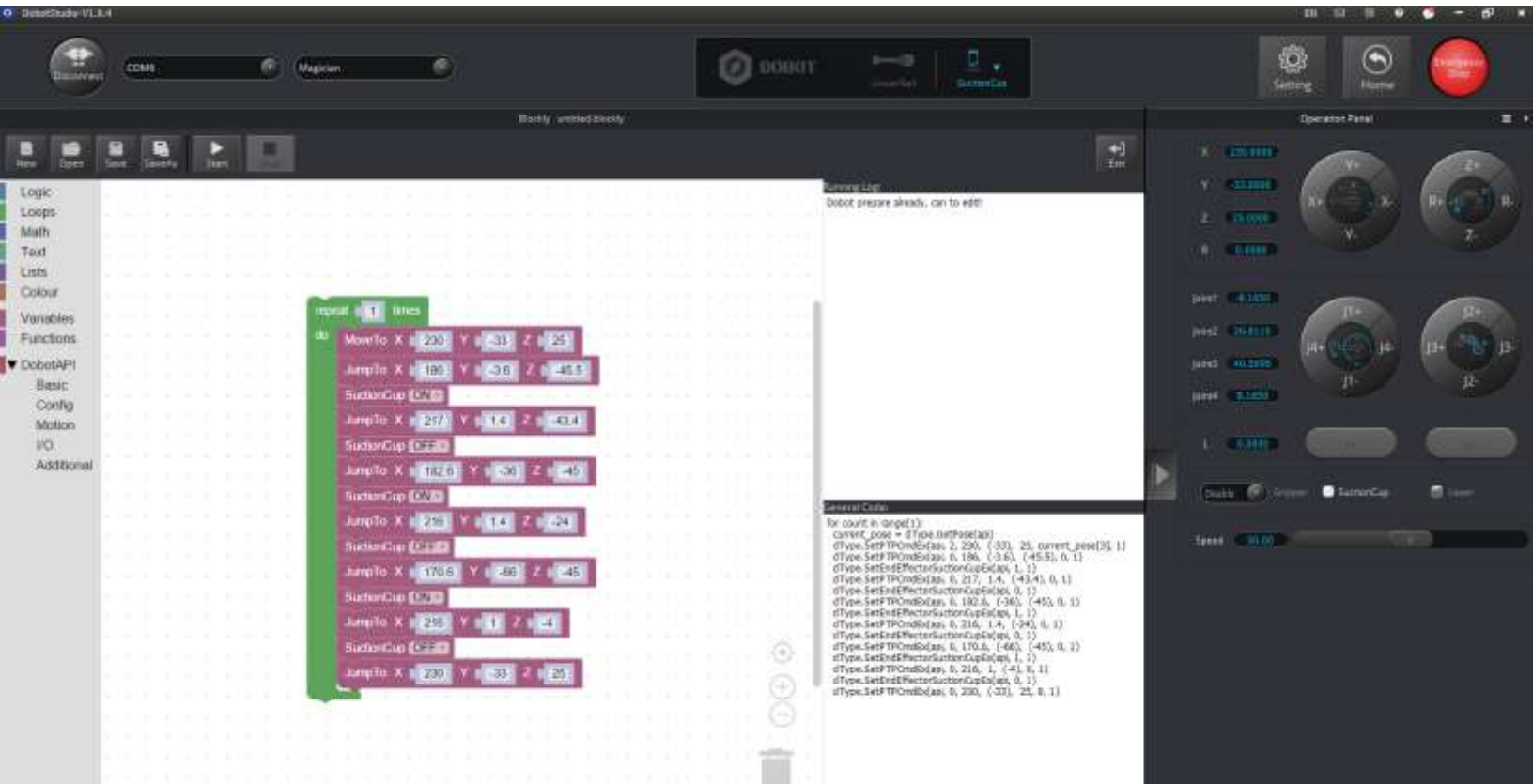
Robot startuje z bezpiecznej pozycji, kolejno sięga po klocek znajdujący się w konkretnym miejscu (załącza ssanie) i odkłada go w wyznaczonym miejscu wieży (wyłącza ssanie). Czynność tą powtarza dla wszystkich klocków. Na koniec przemieszcza się do pozycji końcowej.

Kody źródłowe:

Hand-Hold Teaching:



Bloczki Funkcyjne:



Python:

```
import math

waitTime = 10

current_pose = dType.GetPose(api)

dType.SetPTPCmd(api, 2, 230,    (-33),    25, current_pose[3], 1)

dType.SetPTPCmd(api, 0, 186,    (-3.6),    (-45.5), 0, 1)

dType.SetWAITCmd(api, waitTime, isQueued=1)

dType.SetEndEffectorSuctionCup(api, 1, 1, 1)

dType.SetPTPCmd(api, 0, 217,    1.4,    (-43.4), 0, 1)

dType.SetWAITCmd(api, waitTime, isQueued=1)

dType.SetEndEffectorSuctionCup(api, 0, 1, 1)

dType.SetPTPCmd(api, 0, 182.6,    (-36),    (-45), 0, 1)

dType.SetWAITCmd(api, waitTime, isQueued=1)

dType.SetEndEffectorSuctionCup(api, 1, 1, 1)

dType.SetPTPCmd(api, 0, 216,    1.4,    (-24), 0, 1)

dType.SetWAITCmd(api, waitTime, isQueued=1)

dType.SetEndEffectorSuctionCup(api, 0, 1, 1)

dType.SetPTPCmd(api, 0, 170.6,    (-66),    (-45), 0, 1)

dType.SetWAITCmd(api, waitTime, isQueued=1)

dType.SetEndEffectorSuctionCup(api, 1, 1, 1)

dType.SetPTPCmd(api, 0, 216,    1,    (-4), 0, 1)

dType.SetWAITCmd(api, waitTime, isQueued=1)

dType.SetEndEffectorSuctionCup(api, 0, 1, 1)

dType.SetPTPCmd(api, 0, 230,    (-33),    25, 0, 1)
```

## Problemy:

Znacznych problemów nie napotkaliśmy.

Jedyną rzeczą wartą uwagi, przy pisaniu kodu dla wieży w pythonie, była potrzeba ustawienia kolejki przy włączaniu ssania (isQueued = 1), ponieważ bez tego robot wykonywał jednocześnie 2 komendy, co prowadziło do niepoprawnego działania funkcji ssących.

## Link do filmów:

https://1drv.ms/u/s!AkMYzt8ywVnohkLF8bgrqfNWXjrH?e=gzBCPC

## Pytania kontrolne:

***1.Czy wartości zmiennych złączowych w sposób jednoznaczny determinują pozycję końcówki manipulatora Dobot Magician?***

Tak.

***2.Ile stopni swobody ma Dobot Magician bez założonego narzędzia/efektora?***

Trzy.

***3.Dlaczego na trzeciej osi robota nie umieszczono silnika krokowego, a na drugiej osi (przy podstawie) umieszczono dwa silniki? Na jaki parametr manipulatora wywarło to bezpośredni wpływ?***

Ciężkie silniki krokowe są umieszczone bliskie podstawy co daje duży udźwig i większą stabilność robota. Zapewnia to również sprzężenie dwóch członów robota dzięki czemu uzyskujemy stałą orientację końcówki od podłoża.

***4.Czy okrąg narysowany przez robota będzie idealny? Jeśli nie, to czym jest to spowodowane?***

Okrąg narysowany przez robota nie będzie idealny, ponieważ robot linearyzuje dane odcinki krzywej – interpoluje daną krzywiznę.

***5.Wyłączyliśmy manipulator i zmieniliśmy jego pozycję (przy wyłączonym zasilaniu). Jaką procedurę trzeba wykonać po ponownym włączeniu i dlaczego?***

Należy wykonać auto-home, aby robot wrócił do pozycji startowej, ustalił swoje współrzędne.