

ANRO

Robot nr 5

Laboratorium 1

Zespół:

Grupa pierwsza

Michał Kwarciański

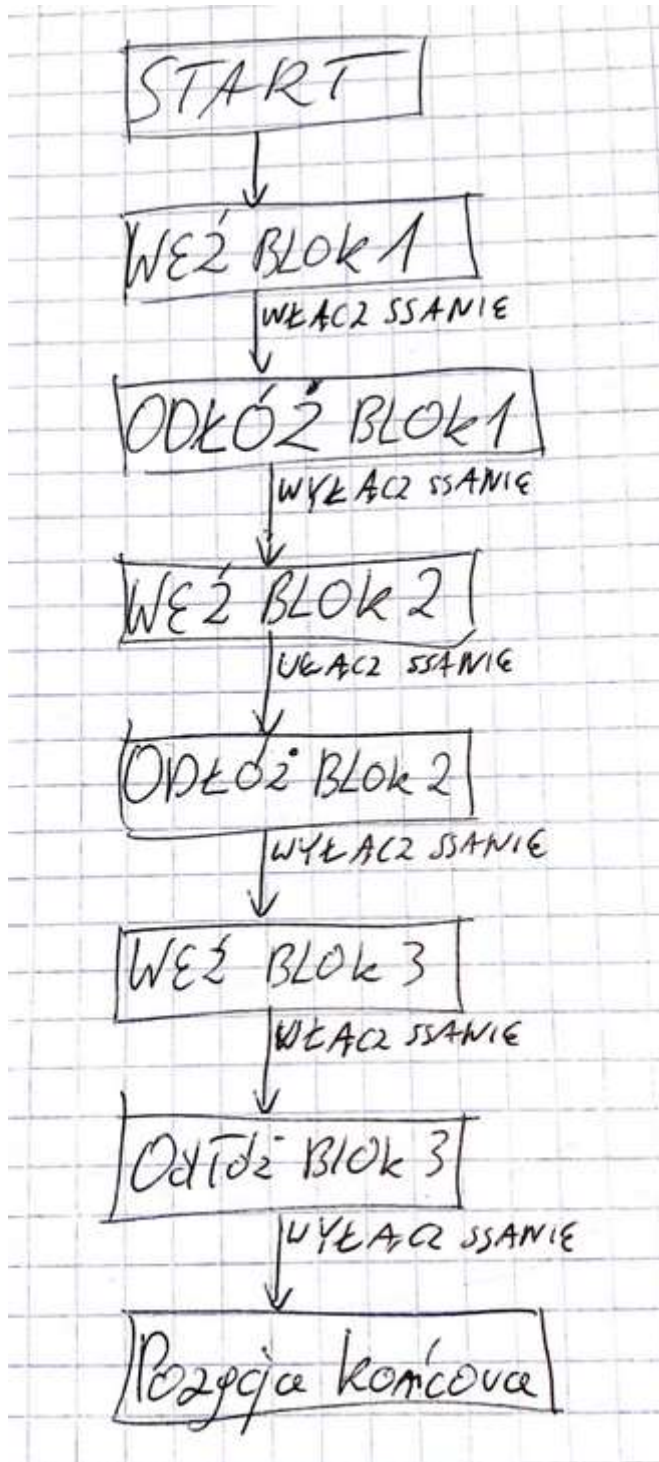
Kacper Marchlewicz

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 znajdować się klocki i wieża.

Prosty opis struktury programów:



START – robot znajduje się w pozycji startowej

WEŹ BLOK 1 – robot włączając ssanie chwytą klocek 1

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

WEŹ BLOK 2 – robot włączając ssanie chwytą klocek 2

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

WEŹ BLOK 3 – robot włączając ssanie chwytą 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:

The screenshot displays the DobotStudio VL 3.4 software interface. The main window shows a table of teaching points for a sequence named 'untitled.playback'. The table has columns for MotionStyle, Name, X, Y, Z, R, PauseTime, and SuctionCup. The sequence consists of 8 steps, alternating between SuctionCupOn and SuctionCupOff states. The right sidebar shows the 'Operation Panel' with various controls like 'X+', 'Y+', 'Z+', 'R+', 'J1+', 'J2+', 'J3+', 'J4+', 'L', 'Speed', and 'SuctionCup'.

MotionStyle	Name	X	Y	Z	R	PauseTime	SuctionCup
LINEAR	START	229.9462	-33.3696	24.3228	-6.3117	0.0	SuctionCupOff
LINEAR	KLOC 1	185.8883	-3.6302	-45.4022	-1.1117	0.0	SuctionCupOn
JUMP	BUDOWA 1	217.2386	1.4253	-45.3983	0.1708	0.0	SuctionCupOff
JUMP	KLOC 2	182.6149	-35.7111	-45.652	-11.8647	0.0	SuctionCupOn
JUMP	BUDOWA 2	216.257	1.3688	-54.5888	0.1708	0.0	SuctionCupOff
JUMP	KLOC 3	176.5054	-46.1796	-45.6267	-21.2023	0.0	SuctionCupOn
JUMP	BUDOWA 3	215.9183	1.1912	-41.138	0.1177	0.0	SuctionCupOff
JUMP	KONIEC	222.9461	-44.241	26.1023	11.1796	0.0	SuctionCupOff

Bloczki Funkcyjne:

The screenshot displays the DobotStudio VL 3.4 software interface in 'Blockly' mode. The main workspace shows a sequence of functional blocks for a teaching sequence. The sequence starts with a 'repeat' block set to 1 time, followed by a 'MoveTo' block, then a 'JumpTo' block, and a 'SuctionCup' block (On/Off). This sequence is repeated for multiple points. The right sidebar shows the 'Operation Panel' with various controls like 'X+', 'Y+', 'Z+', 'R+', 'J1+', 'J2+', 'J3+', 'J4+', 'L', 'Speed', and 'SuctionCup'.

```
repeat 1 times
  MoveTo X: 230 Y: -33 Z: 25
  JumpTo X: 185 Y: -3.6 Z: -45.5
  SuctionCup On
  JumpTo X: 217 Y: 1.4 Z: -43.4
  SuctionCup Off
  JumpTo X: 182.6 Y: -35.7 Z: -45.6
  SuctionCup On
  JumpTo X: 216 Y: 1.4 Z: -54.6
  SuctionCup Off
  JumpTo X: 176.5 Y: -46.2 Z: -45.6
  SuctionCup On
  JumpTo X: 215.9 Y: 1.2 Z: -41.1
  SuctionCup Off
  JumpTo X: 222.9 Y: -44.2 Z: 26.1
```

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.