

Project Vandervoort Jo

29/01/2026

1 Omschrijving

1.1

In mijn project ga ik met de Fanuc robotcell een loopprogramma schrijven, Dit programma zou bijvoorbeeld in productie omgevingen kunnen dienen voor het vullen van vaten.

1.2

Er zijn 2 zijden aan het werkstation. Zijde A is waar de operatoren zich bevinden. De operatoren moeten lege vaten aanleveren voor het vulproces. Aan zijde B is de loading dock waar enkel heftrucks zich mogen verplaatsen. De heftrucks gaan de vaten van zijde B lossen en naar hun volgende locatie brengen.

Vanaf het moment dat de operatoren zijde A voorzien hebben van nieuwe vaten en zijde B is ook leeg gehaald door de heftruckchauffeurs, kan er een startcommando gegeven worden aan de installatie. Eenmaal als het vulproces is beëindigd zal er een startsignaal aan de robot gegeven worden. Hierop gaat de robot elk vat één voor één halen en het gevulde vat in de respectievelijke gespiegelde positie plaatsen aan zijde B. Eenmaal als alle vaten omgezet zijn gaat de robot zich naar zijn homepositie begeven.

1.3

De robot vertrekt steeds vanuit zijn homepositie. in het begin en het einde van het programma wordt de robot steeds naar zijn homepositie gestuurd worden. Dit met het gedacht dat de robot ook al wordt het programma onderbroken nog steeds vanuit zijn homepositie gaat vertrekken. Eenmaal dat de robot een startcommando heeft ontvangen zal hij eerst vanuit zijn register enkele belangrijke posities laden. Deze posities zijn namelijk:

- Wachtpositie zijde A
- Laadpositie zijde A
- Wachtpositie zijde B

- Ontlaadpositie zijde B

Deze posities zijn de enige posities die geteached kunnen worden.

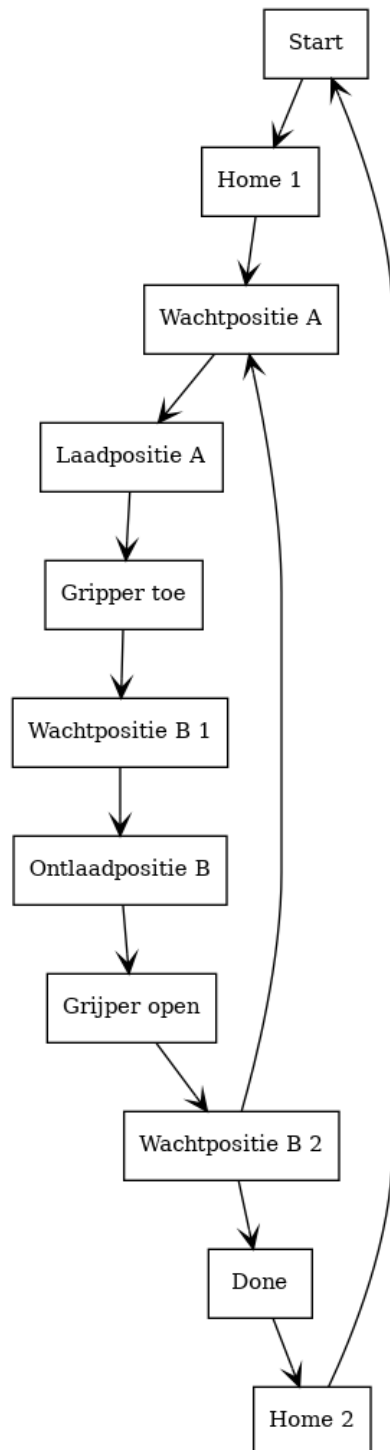
Vanaf hier gaan we 2 loop programma's starten. 1 Loop voor het aantal rijen af te gaan. De andere loop voor elk vat dat er per rij is. De variabele voor het aantal vaten wordt bij elke iteratie gedecrementeerd met 1.

Bij de eerste opdracht in de loop wordt Wachtpositie zijde A aangereden. Deze wachtpositie is net boven het te nemen vat. zodoende we met een rechte lijnige neerwaartse beweging het vat kunnen nemen, en zo de kans op het omstoten te verminderen. Dan beweegt de robot zich naar laadpositie A en sluit zijn gripper bij het aankomen op deze positie. Nu beweegt de robot zich via wachtpositie A naar wacht positie B voor het neerzetten van het vat. Eenmaal wachtpositie B bereikt zal de robot wederom in een rechte lijnige beweging naar beneden het vat neerzetten. De gripper wordt gelost en de robot rijdt terug naar wachtpositie B. Nu zijn we aan het einde van de loop aangekomen. Dit wordt herhaald totdat alle vaten overgezet zijn.

2 Technische documentatie

2.1 Software

2.1.1 Flowchart



2.1.2 Code

2.1.2.1 Totale code

Zie bijlage “Totale Code”

2.1.2.2 Code snippets

inladen van werkposities

Ik heb enkele vaste posities die tijdens programma verloop niet gewijzigd mogen worden. Deze posities mag men wel teachen aangezien dit de basispositie zijn voor het verdere verloop van het programma.

```
9:  !pos inladen ;
10: PR[61:jo_pickWork]=PR[60:jo_pickMem]    ;
11: PR[64:jo_dropWork]=PR[63:jo_dropMem]    ;
12: PR[62:jo_pickWaitWork]=PR[66:jo_pickWaitMem]    ;
13: PR[65:jo_dropWaitWor]=PR[67:jo_dropWaitMem]    ;
```

Offset's op 0 zetten

Hier ga ik mijn R offset registers op 0 zetten aangezien dit geheugen remanent is.

```
15: R[76:offsetX]=0    ;
16: R[77:offsetY]=0    ;
17: R[78]=0            ;
```

For loops

Hier initiëer ik 2 geneste for loops. Eentje voor het aantal rijen en eentje voor het aantal cilinders.

```
22: !for rows ;
23: FOR R[74:CNTR_ROWS]=1 TO R[75:ROWS] ;
24: ;
25: !for cil ;
26: FOR R[72:CNTR]=1 TO R[73:AANTAL] ;
```

Offsets implementeren

Hier ga ik steeds een offset voor de volgende cilinder ingeven.

```
57: PR[61:jo_pickWork]=PR[60:jo_pickMem]    ;
58: PR[61,1:jo_pickWork]=R[76:offsetX]      ;
59: PR[61,2:jo_pickWork]=PR[61,2:jo_pickWork]+R[77:offsetY]    ;
60: ;
61: PR[62:jo_pickWaitWork]=PR[66:jo_pickWaitMem]    ;
62: PR[62,1:jo_pickWaitWork]=R[76:offsetX]      ;
63: PR[62,2:jo_pickWaitWork]=R[77:offsetY]      ;
64: ;
65: PR[64:jo_dropWork]=PR[63:jo_dropMem]    ;
66: PR[64,1:jo_dropWork]=R[76:offsetX]      ;
```

```

67: PR[64,2:jo_dropWork]=PR[64,2:jo_dropWork]+R[78]      ;
68:      ;
69: PR[65:jo_dropWaitWor]=PR[67:jo_dropWaitMem]          ;
70: PR[65,1:jo_dropWaitWor]=R[76:offsetX]                ;
71: PR[65,2:jo_dropWaitWor]=PR[65,2:jo_dropWaitWor]+R[78]      ;
72:      ;

```

2.1.3 Variabelen en IO-lijst

R-register

- 72 = CNTR
- 73 = AANTAL
- 74 = CNTR_ROWS
- 75 = ROWS
- 76 = OFFSET_X
- 77 = OFFSET_Y_PICK
- 78 = OFFSET_Y_DROP

PR-register

- 59 = Home positie = vast
- 60 = Pick positie punt 1 = vast
- 61 = Pick positie werkgeheugen
- 62 = Pick wacht Positie werkgeheugen
- 63 = Drop positie punt 1 = vast
- 64 = Drop positie werkgeheugen
- 65 = Drop wacht positie werkgeheugen
- 66 = Pick wachtpositie punt 1 = vast
- 67 = Drop wachtpositie punt 1 = vast

3 Media

Ik heb een github repository aangemaakt met daarin de vooruitgang van mijn project. <https://github.com/JoVdv/Robotica>

5 Checklist skills

6 Checklist requirements

bijlage

Totale code

```

/PROG  CHESS_J02
/ATTR

```

```

OWNER      = MNEDITOR;
COMMENT    = "";
PROG_SIZE  = 1728;
CREATE     = DATE 25-12-11  TIME 21:33:44;
MODIFIED   = DATE 26-01-28  TIME 01:29:32;
FILE_NAME  = ;
VERSION    = 0;
LINE_COUNT = 108;
MEMORY_SIZE = 2052;
PROTECT    = READ_WRITE;
TCD:  STACK_SIZE    = 0,
      TASK_PRIORITY = 50,
      TIME_SLICE    = 0,
      BUSY_LAMP_OFF = 0,
      ABORT_REQUEST = 0,
      PAUSE_REQUEST = 0;
DEFAULT_GROUP = 1,*,*,*,*;
CONTROL_CODE  = 00000000 00000000;
/APPL
/MN
1:  UFRAME_NUM=5 ;
2:  UTOOL_NUM=5 ;
3:  ;
4: J PR[59:jo_home] 100% FINE ;
5:  ;
6:  R0[8:OFF:Close Gripper]=OFF ;
7:  R0[7:OFF:Open Gripper]=ON ;
8:  ;
9:  !pos inladen ;
10: PR[61:jo_pickWork]=PR[60:jo_pickMem] ;
11: PR[64:jo_dropWork]=PR[63:jo_dropMem] ;
12: PR[62:jo_pickWaitWork]=PR[66:jo_pickWaitMem] ;
13: PR[65:jo_dropWaitWor]=PR[67:jo_dropWaitMem] ;
14:  ;
15: R[76:offsetX]=0 ;
16: R[77:offsetY]=0 ;
17: R[78]=0 ;
18:  ;
19: !init aantal cil ;
20: R[73:AANTAL]=4 ;
21:  ;
22: !for rows ;
23: FOR R[74:CNTR_ROWS]=1 TO R[75:ROWS] ;
24:  ;
25: !for cil ;
26: FOR R[72:CNTR]=1 TO R[73:AANTAL] ;

```

```

27:  ;
28:  !go pre pick ;
29: J PR[62:jo_pickWaitWork] 100% FINE      ;
30:  ;
31:  !go pick ;
32: J PR[61:jo_pickWork] 50% FINE      ;
33:  ;
34:  R0[7:OFF:Open Gripper]=OFF ;
35:  R0[8:OFF:Close Gripper]=ON ;
36:  WAIT 1.00(sec) ;
37:  ;
38:  !go wait pick ;
39: J PR[62:jo_pickWaitWork] 100% FINE      ;
40:  ;
41:  !go wait drop ;
42: L PR[65:jo_dropWaitWor] 100mm/sec FINE      ;
43:  ;
44:  !go drop ;
45: L PR[64:jo_dropWork] 100mm/sec FINE      ;
46:  ;
47:  R0[8:OFF:Close Gripper]=OFF ;
48:  R0[7:OFF:Open Gripper]=ON ;
49:  WAIT 1.00(sec) ;
50:  ;
51:  !go up ;
52: L PR[65:jo_dropWaitWor] 100mm/sec FINE      ;
53:  ;
54:  !add offset ;
55:  R[76:offsetX]=R[76:offsetX]+52      ;
56:  ;
57:  PR[61:jo_pickWork]=PR[60:jo_pickMem]      ;
58:  PR[61,1:jo_pickWork]=R[76:offsetX]      ;
59:  PR[61,2:jo_pickWork]=PR[61,2:jo_pickWork]+R[77:offsetY]      ;
60:  ;
61:  PR[62:jo_pickWaitWork]=PR[66:jo_pickWaitMem]      ;
62:  PR[62,1:jo_pickWaitWork]=R[76:offsetX]      ;
63:  PR[62,2:jo_pickWaitWork]=R[77:offsetY]      ;
64:  ;
65:  PR[64:jo_dropWork]=PR[63:jo_dropMem]      ;
66:  PR[64,1:jo_dropWork]=R[76:offsetX]      ;
67:  PR[64,2:jo_dropWork]=PR[64,2:jo_dropWork]+R[78]      ;
68:  ;
69:  PR[65:jo_dropWaitWor]=PR[67:jo_dropWaitMem]      ;
70:  PR[65,1:jo_dropWaitWor]=R[76:offsetX]      ;
71:  PR[65,2:jo_dropWaitWor]=PR[65,2:jo_dropWaitWor]+R[78]      ;
72:  ;

```

```

73:  ;
74:  ;
75:  !end for cil ;
76:  ENDFOR ;
77:  ;
78:  R[76:offsetX]=25*R[74:CNTR_ROWS]      ;
79:  R[77:offsetY]=R[77:offsetY]-50      ;
80:  R[78]=R[78]+50      ;
81:  R[73:AANTAL]=R[73:AANTAL]-1      ;
82:  ;
83:  PR[61:jo_pickWork]=PR[60:jo_pickMem]      ;
84:  PR[61,1:jo_pickWork]=R[76:offsetX]      ;
85:  PR[61,2:jo_pickWork]=R[77:offsetY]      ;
86:  ;
87:  PR[62:jo_pickWaitWork]=PR[66:jo_pickWaitMem]      ;
88:  PR[62,1:jo_pickWaitWork]=R[76:offsetX]      ;
89:  PR[62,2:jo_pickWaitWork]=R[77:offsetY]      ;
90:  ;
91:  PR[64:jo_dropWork]=PR[63:jo_dropMem]      ;
92:  PR[64,1:jo_dropWork]=R[76:offsetX]      ;
93:  PR[64,2:jo_dropWork]=PR[64,2:jo_dropWork]+R[78]      ;
94:  ;
95:  PR[65:jo_dropWaitWor]=PR[67:jo_dropWaitMem]      ;
96:  PR[65,1:jo_dropWaitWor]=R[76:offsetX]      ;
97:  PR[64,2:jo_dropWork]=PR[65,2:jo_dropWaitWor]+R[78]      ;
98:  ;
99:  ;
100: ;
101: ;
102: ;
103: ;
104: !end for rows ;
105: ENDFOR ;
106: ;
107: ;
108: ;
/POS
/END

```