

Robotica – Robot Industriali e di Servizio

*Lezione 6:
La programmazione e la sensorialità*

6 marzo 2012

Quattro esempi di programmi per robot:

- ⇒ Linguaggi derivati dalle macchine a controllo numerico
- ⇒ Linguaggi derivati dai linguaggi di programmazione “classici” (Basic, Pascal, ecc.)

Lezione 6 La programmazione e la sensorialità

6 marzo 2012 2

Un programma orribile!

```

PROG 00001
N0000 G98 S01,1 S63,8,1 S63.9,3 S71,1 S00
N0001 F8 S00
N0002 F8 S97,1 S00
N0003 G01 F500 S00
N0004 G98 S60,20 S70,1 S00
N0005 G01 F500 S00
N0006 F8 S00
N0007 G01 F500 S00
N0008 G98 S60,20 S71,1 S00
N0009 F8 S60,60 S00
N0010 F8 S00
N0011 G01 F500 S00
N0012 G98 S60,20 S70,1 S00
N0013 G01 F500 S00
N0014 F8 S00
N0015 G01 F500 S00
N0016 G98 S60,20 S71,1 S00
N0017 G01 F500 S10,1 S00
N0018 G98 S17,1,50 S30,1 S32,2 S00
N0019 F8 S97,2 S99
    
```

Lezione 6 La programmazione e la sensorialità

6 marzo 2012 3

Un programma più leggibile

```

PROG 1
10 VELOC 500 mm/sec MAX VELOC 1000 mm/sec
20 TCP 2
30 RECT COORD
40 POS V = 100%
50 SET R5 = 0
60 POS V = 100%
70 POS 50% FINE 1
80 WAIT TIME 2s
90 GRIPPER CLOSE
100 POS V = 50%
110 POS V = 100%
120 POS V = 50% FINE
130 WAIT TIME 2s
140 GRIPPER OPEN
150 POS V = 100%
160 WAIT TIME 6s
170 POS V = 50% FINE
180 WAIT TIME 2s
190 GRIPPER CLOSE
200 POS 50%
210 POS 100%
220 POS 50% FINE
230 WAIT TIME 2s
240 GRIPPER OPEN
250 POS 50%
260 POS 100%
270 ADD 1 TO R5
280 JUMP TO 60 IF R5<50
290 END
    
```

Lezione 6 La programmazione e la sensorialità

6 marzo 2012 4

Un programma leggibile (1)

```

. HERE CASTING
X/JT1   Y/JT2   Z/JT3   O/JT4   A/JT5   T/JT6
54.35   236.78   -29.03  -202.6   78.90   113.76

. HERE BANDSAW
X/JT1   Y/JT2   Z/JT3   O/JT4   A/JT5   T/JT6
305.5   45.78   35.8    -56.8    -34.98  78.21

. HERE CONVEYOR
X/JT1   Y/JT2   Z/JT3   O/JT5   A/JT6   T/JT6
205.45  -389.4  25.97  -178.4   78.63   170.3

. EDIT BANDSAW FEEDING .1
. EXECUTE BANDSAW FEEDING .1,50
1.? SETI COUNT = 0
2.? 10 APPRO CASTING, 25
3.? MOVE 'S' CASTING
4.? WAIT 2
5.? CLOSE I
    
```

Lezione 6 La programmazione e la sensorialità

6 marzo 2012 5

Un programma leggibile (2)

```

6.? DEPART 'S', 50
7.? APPRO BANDSAW, 100
8.? MOVE 'S', BANDSAW
9.? WAIT 2
10.? OPEN I
11.? DEPART 'S', 350
12.? WAIT 6
13.? APPRO BANDSAW, 100
14.? MOVE 'S' BANDSAW
15.? WAIT 2
16.? CLOSE I
17.? DEPART 'S', 250
18.? APPRO CONVEYOR, 25
19.? MOVE 'S', CONVEYOR
20.? WAIT 2
21.? OPEN I
22.? DEPART 'S' 100
23.? SETI COUNT = COUNT +1
24.? TYPEI COUNT
25.? GOTO 10
26.?
    
```

Lezione 6 La programmazione e la sensorialità

6 marzo 2012 6

Un programma ottimo (1)

```

1 PROGRAM machine tool loading
2
3 VAR
4 PARK           : Position      : park position
5 CHUCK          : Position      : chuck position
6 CONVEYOR IN   : Position      : conveyor-in position
7 CONVEYOR OUT  : Position      : conveyor-out position
8 PART COUNT 1  : Integer        : counter variable
9
10 CONST
11 CHUCK OPEN = 1 : output signal to open chuck
12 CHUCK CLOSE = 2 : output signal to close chuck
13 STOP CYCLE = 3 : input signal to stop program
14
15 BEGIN
16 $SPEED = 500
17 $MOTYPE = linear
18 $TERM TYPE = coarse
19 PART COUNT 1 = 0
20 REPEAT
21     MOVE TO PARK
22     MOVE TO CHUCK
23     CLOSE HAND 1

```

Lezione 6 La programmazione e la sensorialità

6 marzo 2012 7

Un programma ottimo (2)

```

24     DOUT [1] = ON
25     DELAY 2000
26     MOVE TO PARK
27     MOVE TO CONVEYOR-OUT
28     OPEN HAND 1
29     DELAY 2000
30     MOVE TO CONVEYOR-IN
31     CLOSE HAND 1
32     DELAY 2000
33     MOVE TO PARK
34     MOVE TO CHUCK
35     DOUT [2] = ON
36     DELAY 2000
37     OPEN HAND 1
38     DELAY 2000
39     MOVE TO PARK
40     PART COUNT 1 = PART COUNT 1+1
41
42     UNTIL DIN [stop cycle] = ON
43
44     WRITE ('Total Components Machined', part count)
45
46     END machine tool loading

```

Lezione 6 La programmazione e la sensorialità

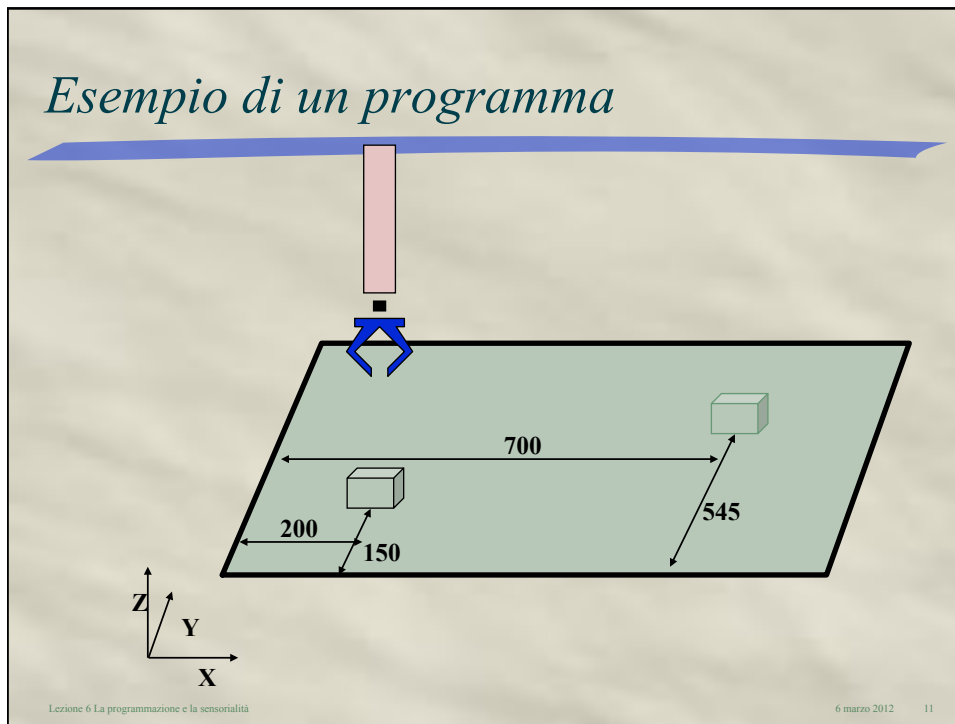
6 marzo 2012 8

Le categorie di linguaggi

- ⇒ Linguaggi per la programmazione esplicita
 - Linguaggi orientati ai giunti
 - Linguaggi orientati al manipolatore
 - Linguaggi orientati agli oggetti
- ⇒ Linguaggi per la programmazione implicita
 - Pianificatori automatici

Le categorie di linguaggi (espliciti) per manipolatori:

- ⇒ Linguaggi orientati ai giunti
- ⇒ Linguaggi orientati al robot
- ⇒ Linguaggi orientati agli oggetti



Il programma scritto in MAL

```

10 MOVE XL=200, YL= 150, ZL=200
20 OPEN LHAND
30 MOVE ZL=20
40 CLOSE LHAND
50 INCR ZL=100
60 MOVE XL=700, YL=545
70 INCR ZL=-100
80 OPEN LHAND
90 INCR ZL=200
    
```

700

200 150 545

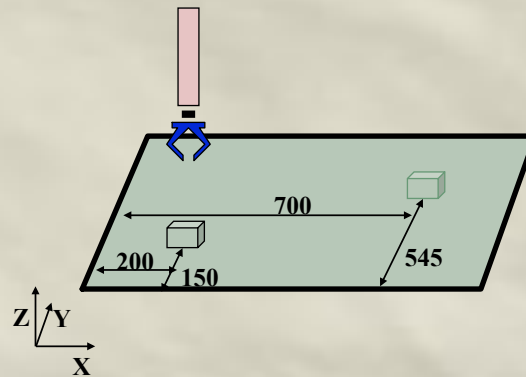
Z
Y
X

Lezione 6 La programmazione e la sensorialità 6 marzo 2012 12

Il programma scritto in VAL

```

APPRO P1, 100
OPENI
MOVES P1
CLOSEI
DEPARTS 100
APPROS P2, 100
MOVES P2
OPENI
DEPARTS 200
  
```



Lezione 6 La programmazione e la sensorialità

6 marzo 2012 13

Confronto fra i due sistemi

- ⇒ Linguaggi orientati ai giunti:
 - Utilizzabili solo su alcuni robot (tipicamente cartesiani)
 - Scarso controllo di traiettoria
 - Semplici da implementare
- ⇒ Linguaggi orientati al robot:
 - Indipendenti dal manipolatore
 - Buon controllo di traiettoria (con possibilità di via points)
 - Possibilità di spostare il sistema di riferimento (anche in tempo reale!)

Lezione 6 La programmazione e la sensorialità

6 marzo 2012 14

Esempio in AML

```
-- ESEMPIO MUOVE, SCENDE, CHIUDE E SALE --
PICKUP:SUBR;
PMOVE(PT(250,300,0,0));
ZMOVE(-250);
GRASP;
DELAY(1);
ZMOVE(0);
END;
```

Lezione 6 La programmazione e la sensorialità

6 marzo 2012 15

Altro esempio in AML:

```
-- PICK AND PLACE CON RITORNO --
MAIN:SUBR;
PICKUP:SUBR;
PMOVE(PT(250,300,0,0));
ZMOVE(-250);
GRASP;
DELAY(1);
ZMOVE(0);
END;
-- FINE PICKUP --

PUTDOWN:SUBR;
PMOVE(PT(250,450,0,0));
ZMOVE(-250);
RELEASE;
DELAY(1);
ZMOVE(0);
END;
-- FINE PUTDOWN --

PICKUP2:SUBR;
PMOVE(PT(250,450,0,0));
ZMOVE(-250);
GRASP;
DELAY(1);
ZMOVE(0);
END;
-- FINE PICKUP2 --

PUTDOWN2:SUBR;
PMOVE(PT(250,300,0,0));
ZMOVE(-250);
RELEASE;
DELAY(1);
ZMOVE(0);
END;
-- FINE PUTDOWN2 --

PICKUP;
PUTDOWN;
PICKUP2;
PUTDOWN2;
END;
-- FINE MAIN --
```

Lezione 6 La programmazione e la sensorialità

6 marzo 2012 16

Particolarità di AML

⇒ Movimenti differenziali...

```
DPMOVE (<ΔX, ΔY, ΔZ, ΔR>);
```

Particolarità di AML

⇒ Velocità di movimento

```
PAYLOAD(0);      = Default to switches  
PAYLOAD(1);      = Slow with heavy load  
PAYLOAD(10);     = Fast with light load  
PAYLOAD(12);     = 20 percent of PAYLOAD(1);  
PAYLOAD(19);     = 90 percent of PAYLOAD(1);
```

Particolarità di AML

⇒ Precisione di movimento

```

ZONE(15);    = Not precise but fast
ZONE(1);     = More accurate goal
ZONE(0);     = Default to switches
  
```

Particolarità di AML

```
LINEAR(quality);
```

The "quality" represents the deviation from a straight line. A one (1) for the "quality" gives the straightest achievable move between two points. A fifty (50) for the "quality" gives the greatest deviation from a straight line. A low "quality" value means straighter arm movement, but it also means slower arm movement. The "quality" numbers and their corresponding LINEAR effects are listed in tabular form in the appendices. To deactivate linear motion mode and return to a normal mode, the value 0 is used for "quality." Examples of the command's usage are outlined below.

```

LINEAR(1);    = Straight as possible
LINEAR(0);    = Exit linear
LINEAR(50);   = Fastest with greatest error
  
```

Particolarità di AML

⇒ Test su input digitali

```
TESTI(digital_input_point,value,label);
```

Particolarità di AML

⇒ Acquisizione di coordinate

```
WHERE(point);
```

Particolarità di AML

```
GUARDI(digital_input_point,value);
```

```
MSTATUS(counter_name);
```

This command allows you to determine the completion status of the last executed move. It specifies the counter into which the status is loaded. Counters are run-time variables which can hold an integer or real value. Counters are discussed in "Variables" on page 4-38.

Status codes are listed below.

```
0 = motion completed normally
1 = motion terminated by a guard
2 = motion not started due to a guard
```

After loading the code into a counter, the program may use either TESTC or COMPC to test the value.

Particolarità di AML

```
STAT: STATIC COUNTER;
DI: STATIC COUNTER;
PT1: NEW PT(0,500,0,0);
PT2: NEW PT(0,400,0,0);
PT3: NEW PT(0,0,0,0);
TESTG: SUBR;
      SETC(DI,3);           -- SET THE POINT GUARD TO 3
      GUARDI(DI,1);        -- GUARD POINT 3 FOR A 1 CONDITION
      PMOVE(PT1);          -- MAKE A MOVE
      PMOVE(PT2);
      MSTATUS(STAT);       -- DETERMINE THE COMPLETION STATUS
      TESTC(STAT,0,GOOD);  -- MOTION TERMINATED BY GUARD?
      WHERE(PT3);          -- UPDATE POINT
      WRITEO(ERROR,1);     -- INFORM OPERATOR
      WAITI(OP_OK,1,0);    -- WAIT FOR OK
      PMOVE(PT2);          -- FINISH MOVE
GOOD: ZMOVE(-100);         -- YES, CONTINUE THE PROGRAM
      ZMOVE(0);
      END;
```

Un esempio in AS:

```
OPENI
HOME
JAPPRO #tutorial_centro, 100
LMOVE #tutorial_centro
CLOSEI
JAPPRO #tutorial_centro, 100
DRAW 68,68,0,0,0,0
DRAW 0,0,-100,0,0,0
OPENI
DRAW 0,0,100,0,0,0
HOME
```

Lezione 6 La programmazione e la sensorialità

6 marzo 2012 25

Altri sistemi di programmazione

⇒ Controllo in forza

- Richiede appositi sensori
- Comporta notevoli problemi di controllo
- Non sempre la forza di reazione è normale alla superficie di contatto!
 - Cedevolezza selettivamente controllabile

Lezione 6 La programmazione e la sensorialità

6 marzo 2012 26

La programmazione implicita

- ⇒ Non si specificano le azioni, ma il risultato
- ⇒ Occorre fornire al manipolatore tutte le informazioni:
 - Descrizione dei pezzi
 - Descrizione del risultato
 - Layout della cella
 - Descrizione del manipolatore(i)
- ⇒ La discuteremo in dettaglio più avanti