Robot Programming Methods

- **Robot Programming** is the defining of desired motions so that the robot may perform them without human intervention.
  - identifying and specifying the robot configurations (i.e. the pose of the end-effector, P_e, with respect to the base-frame)

1.MANUAL METHOD
2.WALKTHROUGH METHOD
3.LEADTHROUGH METHOD
4.OFF-LINE PROGRAMMING

Type of Robot Programming

- Joint level programming
  - basic actions are positions (and possibly movements) of the individual joints of the robot arm: joint angles in the case of rotational joints and linear positions in the case of linear or prismatic joints.
- Robot-level programming
  - the basic actions are positions and orientations (and perhaps trajectories) of P_e and the frame of reference attached to it.
- High-level programming
  - Object-level programming
  - Task-level programming

Object Level Programming

- basic actions are operations to be performed on the parts, or relationships that must be established between parts

pick-up part-A **by** side-A1 **and** side-A3

move part-A **to** location-2

pick-up part-B **by** side-B1 **and** side-B3

put part-B **on-top-off** part-A

with side-A5 **in-plane-with** side-B6 **and**

with side-A1 **in-plane-with** side-B1 **and**

with side-A2 **in-plane-with** side-B2

Task Level Programming

- basic actions specified by the program are complete tasks or subtasks

paint-the car-body *red*

assemble the gear-box

ROBOT PROGRAMMING

- Typically performed using one of the following
  - On line
    - teach pendant
    - lead through programming
  - Off line
    - robot programming languages
    - task level programming

Robot Programming Methods

- Offline:
  - write a program using a text-based robot programming language
  - does not need access to the robot until its final testing and implementation
- On-line:
  - Use the robot to generate the program
    - Teaching/guiding the robot through a sequence of motions that can them be executed repeatedly
- Combination Programming:
  - Often programming is a combination of on-line and off-line
    - on-line to teach locations in space
    - off-line to define the task or "sequence of operations"

Use of Teach Pendant

- hand held device with switches used to control the robot motions
- End points are recorded in controller memory
- sequentially played back to execute robot actions
- trajectory determined by robot controller
- suited for point to point control applications

Lead Through Programming

- lead the robot physically through the required sequence of motions
- trajectory and endpoints are recorded, using a sampling routine which records points at 60-80 times a second
- when played back results in a smooth continuous motion
- large memory requirements

On-Line/Lead Through

- Advantage:
  - Easy
  - No special programming skills or training
- Disadvantages:
  - not practical for large or heavy robots
  - High accuracy and straight-line movements are difficult to achieve, as are any other kind of geometrically defined trajectory, such as circular arcs, etc.
  - difficult to *edit out* unwanted operator moves
  - difficult to incorporate external sensor data
  - Synchronization with other machines or equipment in the work cell is difficult
  - A large amount of memory is required

On line Programming

- Requires access to the robot
- Programs exist only in the memory of robot control system – often difficult to transfer, document, maintain, modify

- Easy to use, no special programming skills required
- Useful when programming robots for wide range of repetitive tasks for long production runs
- RAPID

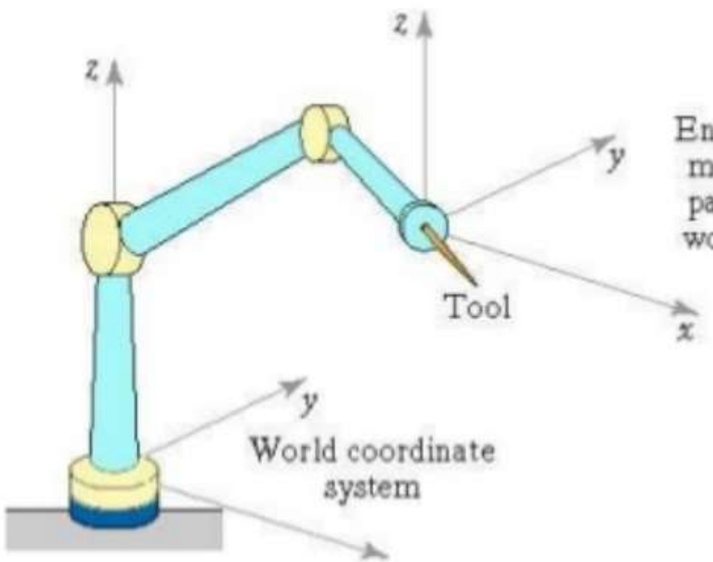
On-Line/Teach Box

- Advantage:
  - Easy
  - No special programming skills or training
  - Can specify other conditions on robot movements (type of trajectory to use – line, arc)
- Disadvantages:
  - Potential dangerous (motors are on)

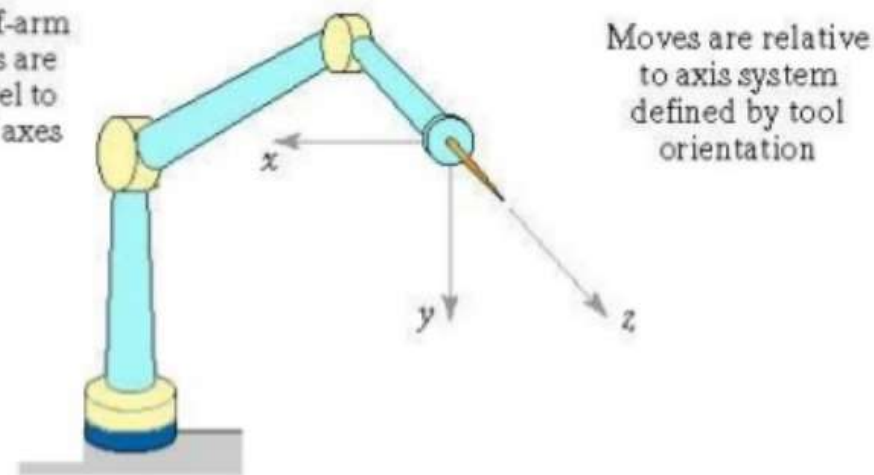
Off-line Programming

- Programs can be developed without needing to use the robot
- The sequence of operations and robot movements can be optimized or easily improved
- Previously developed and tested procedures and subroutines can be used
- External sensor data can be incorporated, though this typically makes the programs more complicated, and so more difficult to modify and maintain
- Existing CAD data can be incorporated-the dimensions of parts and the geometric relationships between them, for example.
- Programs can be tested and evaluated using simulation techniques, though this can never remove the need to do final testing of the program using the real robot
- Programs can more easily be maintained and modified
- Programs can more be easily properly documented and commented.

Coordinate Systems

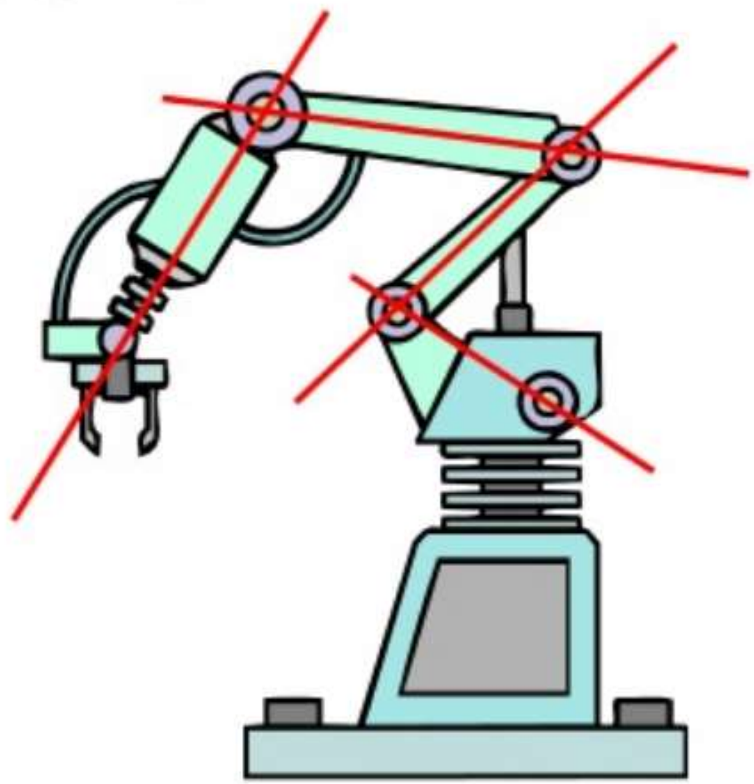


World coordinate system



Tool coordinate system

Configuration: Any particular position and orientation of P_e in space, and so any particular set of joint values, is called a *configuration* of the robot arm.



Motion Commands

MOVE P1

HERE P1 - used during lead through of manipulator

MOVES P1

DMOVE(4, 125)

APPROACH P1, 40 MM

DEPART 40 MM

DEFINE PATH123 = PATH(P1, P2, P3)

MOVE PATH123

SPEED 75

- Robot motion programming commands
-
- MOVE P1
- HERE P1 -used during leadthrough of manipulator
- MOVES P1
- DMOVE(4, 125)
- APPROACH P1, 40 MM
- DEPART 40 MM
- DEFINE PATH123 = PATH(P1, P2, P3)
- MOVE PATH123
- SPEED 75
-
- Input interlock:
- WAIT 20, ON
- Output interlock:
- SIGNAL 10, ON
- SIGNAL 10, 6.0
- Interlock for continuous monitoring:
- REACT 25, SAFESTOP
-
- Gripper
- OPEN
- CLOSE
- Sensor and servo-controlled hands
- CLOSE 25 MM

Interlock and Sensor Commands

Interlock Commands

- WAIT 20, ON
- SIGNAL 10, ON
- SIGNAL 10, 6.0
- REACT 25, SAFESTOP

Gripper Commands

- OPEN
- CLOSE
- CLOSE 25 MM
- CLOSE 2.0 N

Programming Languages

- Motivation
  - need to interface robot control system to external sensors, to provide “real time” changes based on sensory equipment
  - computing based on geometry of environment
  - ability to interface with CAD/CAM systems
  - meaningful task descriptions
  - off-line programming capability

- Large number of robot languages available
  - AML, VAL, AL, RAIL, RobotStudio, etc. (200+)
- Each robot manufacturer has their own robot programming language
- No standards exist
- Portability of programs virtually non-existent

Robot Programming Languages

- The VALTM Language
- The VAL language was developed for PUMA robot
- Monitor command are set of administrative instructions that direct the operation of the
- robot system. Some of the functions of Monitor commands are
- Preparing the system for the user to write programs for PUMA
- Defining points in space
- Commanding the PUMA to execute a program
- Listing program on the CRT
- Examples for monitor commands are: EDIT, EXECUTE, SPEED, HERE etc.

THE MCL LANGUAGE

- MCL stands for Machine Control Language developed by Douglas.
- The language is based on the APT and NC language. Designed control complete
- manufacturing cell.
- MCL is enhancement of APT which possesses additional options and features needed
- to do off-line programming of robotic work cell.
- Additional vocabulary words were developed to provide the supplementary
- capabilities intended to be covered by the MCL. These capability include Vision,
- Inspection and Control of signals
- MCL also permits the user to define MACROS like statement that would be
- convenient to use for specialized applications.
- MCL program is needed to compile to produce CLFILE.
- Some commands of MCL programming languages are DEVICE, SEND, RECEIV,
- WORKPT, ABORT, TASK, REGION, LOCATE etc.