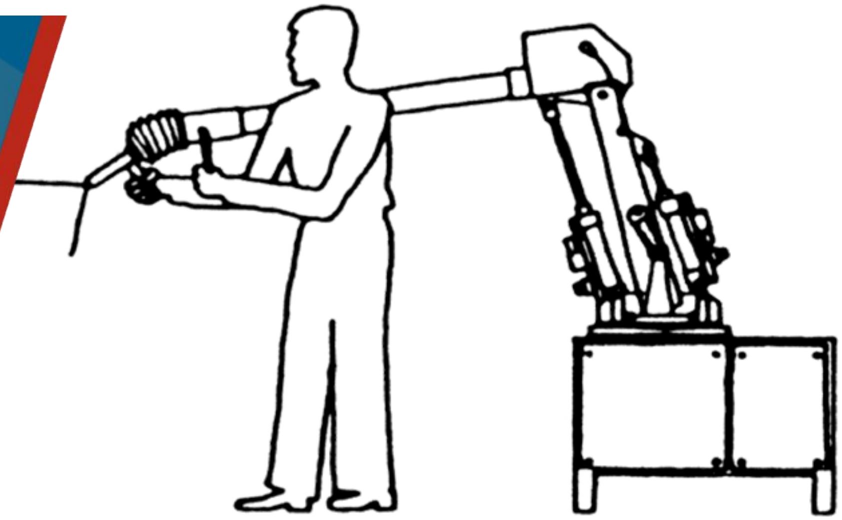
RRLAB
THE ROBOTICS RESEARCH LAB

Robot Programming



Prof. Dr. Karsten Berns
Robotics Research Lab
Department of Computer Science
TU Kaiserslautern, Germany

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

Contents

- Programming of industrial robot
- Online programming
  - Teach pendants
  - Lead through methods
- Offline programming (OLP)
  - Steps of OLP process
  - Robot teaching with visual components
- Teaching by demonstration
- ROS: Robot system operation

- Foliensatz z.T. von
  - Dr. R. Lafrenz, Universität Stuttgart
  - Prof. Zühlke PAK, TU KL

Programming of Industrial Robots

- A key aspect to any robotic system.
- It relays vital information to an industrial robot making it capable of functioning and carrying out specific tasks.
- Once viewed as daunting as it required complex lines of code.
- Advancements in robotics has led to the development of more intuitive programming methods that are user friendly to even the most novice of operators.

Programming of Industrial Robots

- Operating system
  - Real-time capability
  - Interface for robot control
- Programming language
  - Robot-specific language (VAL, …)
- Libraries for standard languages (RCCL for C, …)
- Robot-oriented routines are special data types:
  - matrices,
  - kinematics/dynamics routines,
  - movement commands (Cartesian, joint space)
  - effector commands.

Programming of Industrial Robots

- Task-oriented routines are the knowledge base with:
  - Environmental model,
  - Rule base for the breakdown of tasks,
  - Planning algorithms and the division of complex tasks.

- The most common programming methods for industrial robots:
  - Online programming,
  - Offline programming,
  - Teaching by Demonstration.

Online Programming: Teach Pendants

- Programming the robot when it is online or in an operational mode.
- The two types of online programming:
  - Teach pendant programming
  - Lead through programming.
- Teach pendants
  - most common methods of industrial robot programming.
  - Over 90\% of robots are programmed with teach pendants. Handheld devices connected to the robot via cables.

Online Programming: Teach Pendants

- Positioning and configuration of the robot with special control devices
- Control devices
  - Teachbox
  - Joystick
  - Mouse
  - Teach-ball
  - Applications
  - Point-to-point control
  - Multipoint control (MP)

Online Programming: Teach Pendants

- Many of the newer FANUC teach pendants feature a touch screen.
- Operator input commands through the pendant’s keyboard.
- Operator steps the robot through the program point by point recording the coordinates for each point.
- Technician is able to send the robot to a desire position and record the point.
- Technician may change the speed.
- Once the technician has recorded and tested the program, the teach pendant is disconnected and the robot can operate the stored program at full speed.
- Teach pendant programming is offten associated with control path and point to point control.

Online Programming: Teach Pendants

- A FANUC R-2000ib can be programmed to weld car frames.
- A FANUC LR Mate 200id is capable of locating parts to be placed on a conveyor due to the instructions relayed to it through an operator’s input using a teach pendant.

FANUC R-2000ib

FANUC LR Mate 200id

Online Programming: Teach Pendants

- Advantages:
  - Most traditional industrial robots come with a teach pendant

    $\longrightarrow$ familiar to technicians.
  - Allow precise positioning, as the robot can be programmed using numerical coordinates, in either world coordinates, robot coordinates or another coordinate system.
  - Great for simple movements, such as painting in a straight line or over a large flat surface.
  - Industrial robots come with teach pendants, making it easy to get robots up and running without needing to purchase or integrate additional programming software.

Online Programming: Teach Pendants

- Disadvantages:
  - Disruptive to the whole system due to robot downtime. The robot must be put into teach mode and all operations using the robot halted until it has been programmed.
  - Requires training to learn and program.
  - Might be difficult for skilled crafts people who are unfamiliar with programming.

Online Programming: Lead Through Methods

- Involve programming robots through demonstration associated with continious path control.
- Operator physically moves a robot through a desired task.
  - Also calls lead through the nose.
- Best for detailed applications as it simplifies programming by eliminating the need to write several lines of complex code.
- Faster than other programming methods but is not suited for simple or straightforward applications.
- Declined in popularity with traditional industrial robots as many have become too large or heavy to physically manipulate their robotic arms.
- Gained popularity for programming collaborative robots.

Online Programming: Lead Through Methods

- The FANUC CR-35iA is one example of a collaborative robot that uses demonstration programming.

Online Programming: Lead Through Methods

- User manually moves the robot arm with the tool using his hands.
- Possible for non-programmers to control and program robots.
- However, it still lacks many of the capabilities of traditional robot programming.
- It is desired to make the robot compliant in one or more directions to enable a user to move the robot by hand to a desired position while recording each step.

Online Programming: Lead Through Methods

- The robot controller usually is passively controlled.

  → the stiffness of the manipulator is reduced in all Cartesian directions and orientations and by that the manipulator is made "fully compliant".
- Fully compliant means: The manipulator is made compliant in all possible directions and orientations, which enables the user to lead the robot arm around freely in any direction and orientation.
- Manipulator will stay in the desired position when the user stops the movement of the robot arm during passive lead-through.
- A passively controlled robot makes it easy and fast for the user to program large and sweeping motions during lead-through programming.

Online Programming: Lead Through Methods

- When the user needs the robot to make large sweeping motions he grasps the robot arm firmly and leads the arm freely to the desired position.
- A disadvantage with the passive control of the robot:
  - Difficult to program more precise movements, such as linear movements.
- This problem can be solved by actively controlling the movements of the robot.
- Example:
  - the movements of the robot can be restricted so that the manipulator only moves in one direction.

Online Programming: Lead Through Methods

- Advantages:
  - Easy to use
  - Little or no programming skill is required.
  - Speed of programming is high
- Disadvantages:
  - Required production line shutdown
  - The technician should be inside of work envelope while the robot is operating which exposes him to great risk of damage

Offline Programming (OLP)

- An increasingly essential tool for manufacturing professionals charged with planning new robot work cells.
- Used for much more than developing the robot program.
- OLP software helps with planning and optimizing the work cell design, virtually commissioning the robot, and accelerating the time to production.
- Use 3D CAD data to create a virtual model of the robot and work cell, and simulate the processes and workflows inside and outside the cell
- Powerful tool for engineers and planners to evaluate trade-offs and make better decisions.

Offline Programming (OLP)

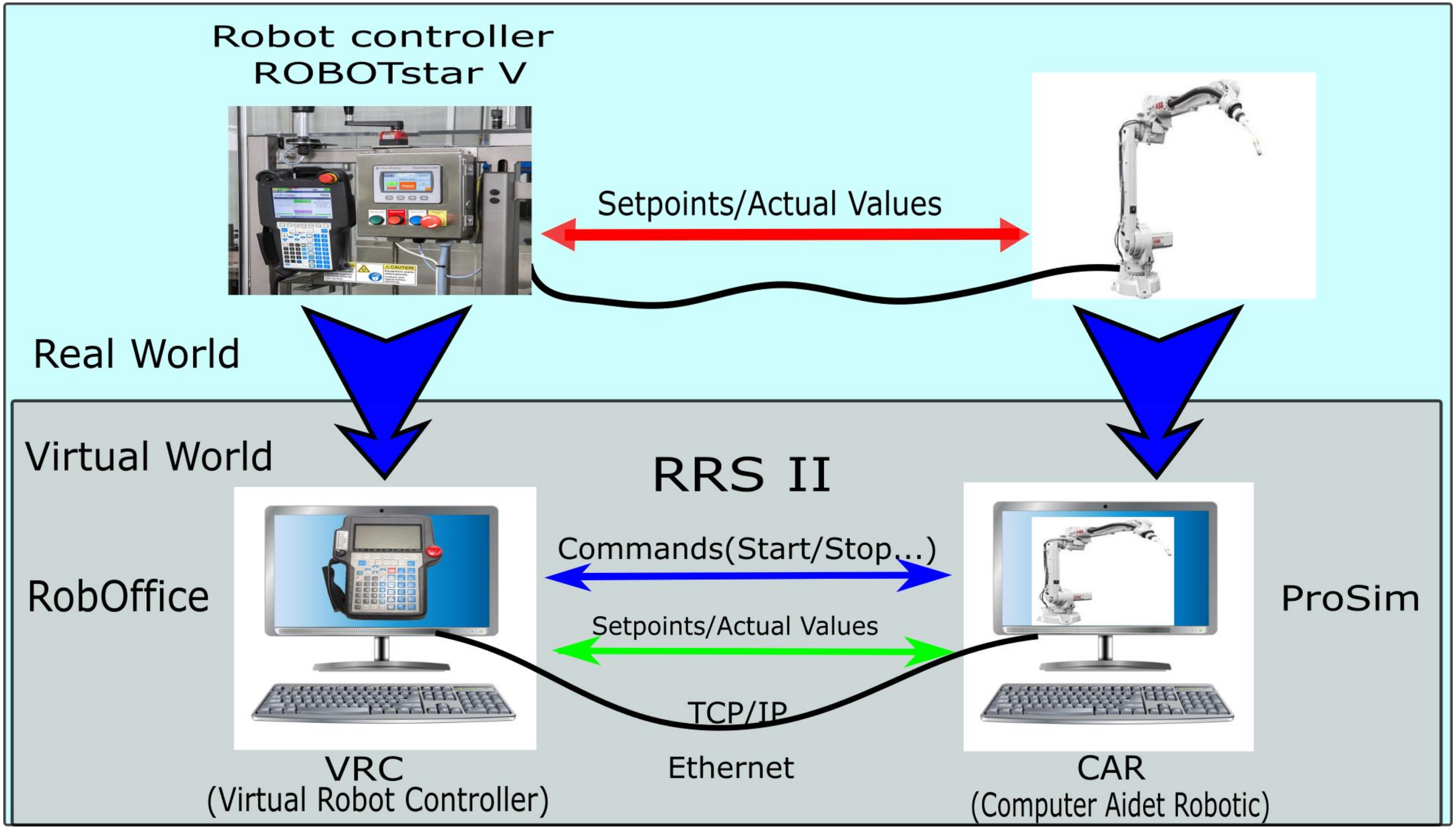
- Provides a meaningful return on investment for many types of automation projects, by:
  - saving time
  - improving productivity
  - and helping manufacturers to identify opportunities for cost savings.

Offline Programming: Steps of the OLP Process

1. **Generation of Models:**
   - creating or obtaining 3D CAD models of the equipment, work pieces, enclosure, tools, and other resources and fixtures that will be in the work cell;
   - Importing them into your OLP software.
   - Accuracy of the models and process related information used is critical to generating a reliable simulation of the process and error-free offline program for the robots.

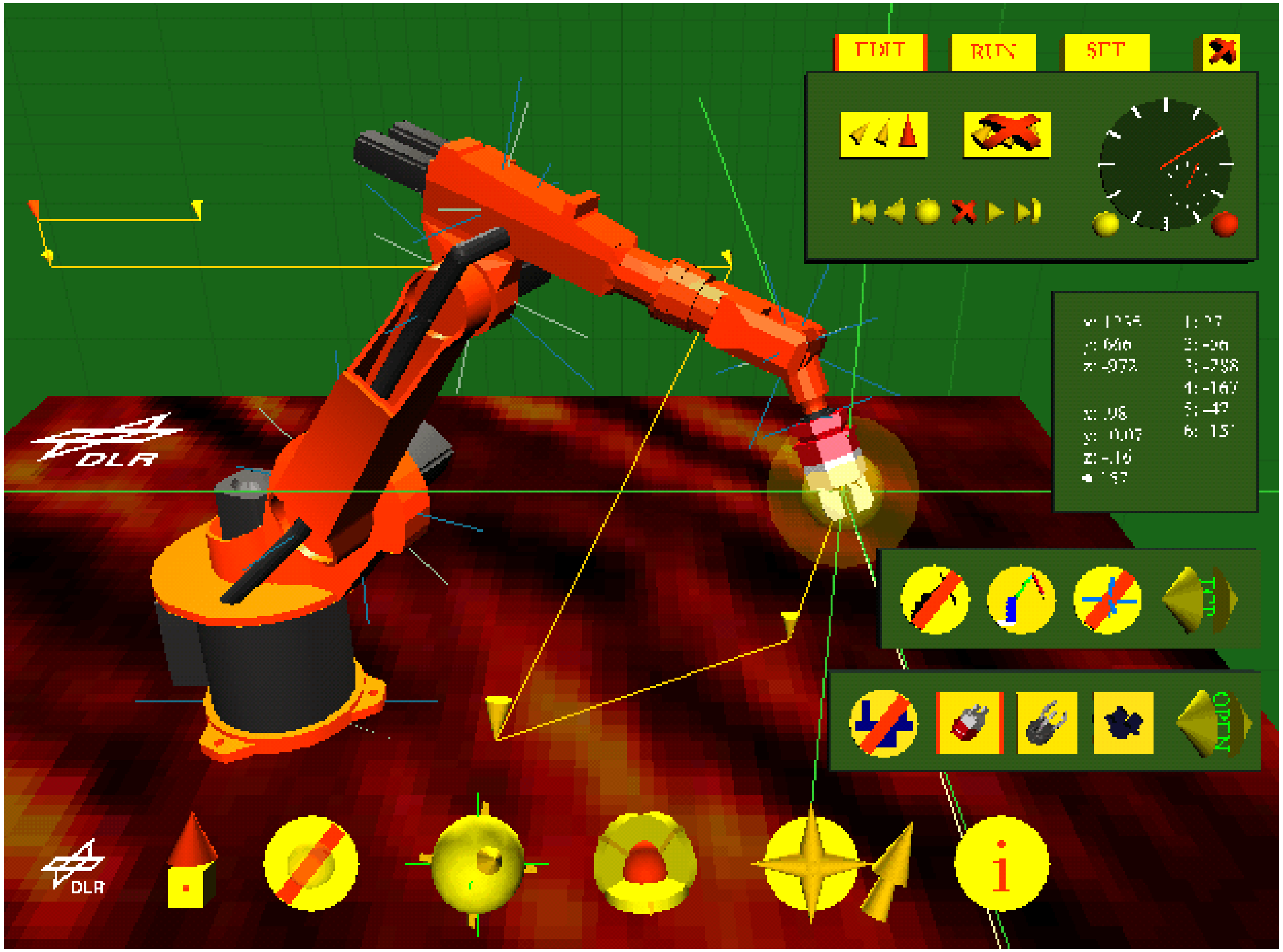
- The creation of the simulation is very complex.
- Compared to this, the optimization of motion sequences is cheaper.
- There is a risk of incorrect or incomplete simulation
- Both kinematic and dynamic parameters must be as exact as possible, otherwise damage to robots is possible

Robot Simulation: Realistic Robot Simulation



RRLAB THE ROBOTICS RESEARCH LAB

Robot Simulation: VRML 2.0 DLR und KUKA



Comparison Between Simulation And Robot

Calibration effort

Comparison between world coordinate system

\+ Less programming effort
\+ Absolute coordinate input possible
\- Calibration is expensive
\- No compensation for inaccuracy of robot system

Coordinate input relative to reference body in work space

\+ Position of reference body is known
\+ Easy calibration
\- Calibration is expensive
\- Positioning error only less, not zero

Individual detection of position via image processing

\+ Position needs to be known only roughly. With image processing the position errors are gone
\- High programming effort on robot and simulation side

Programming effort

Offline Programming: Steps of the OLP Process

2. Tool Path Generation

- involves extracting robot positions from 3D CAD data with a specific tool center point.
- Many OLP software packages can do this automatically, and have built-in functions to automatically generate paths from features of the CAD models, such as corners, edges, or other geometry features.

Offline Programming: Steps of the OLP Process

3. Process Optimization

- Iterative design loop with a number of factors and trade offs that need to be considered, so simulation helps significantly with this process.
- Incorporates:
  - Trajectory planning
  - Process planning
  - Tooling design

Process Optimization: Trajectory planning:

- Involves determining the best route for the robot to make, from Point A to Point B.
- It’s not always about planning the fastest or shortest route.
- Robot work cells are typically designed in compact configurations.
- Motions have to be planned carefully to avoid unintended interactions between the robot and other objects in the cell.
- Factors that need to be considered:
  - Motion type and joint configuration
  - Speed / Acceleration,
  - Reachability and collision detection
  - Avoidance.

Process Optimization: Process planning

- Involves planning the processes and workflow in the work cell.
- The major constraints for this step are:
  - Budget,
  - Productivity,
  - Quality,
- Several factors and trade offs that need to be considered. This step includes:
  - Layout design,
  - Resource selection (including robots and other equipment)
  - Maintenance considerations,
  - Sequence optimization.

Process Optimization: Tooling design

- Involves the selection, modification, and placement of tools.
- This includes robotic end of arm tooling and other tools that touch the work piece, such as positioner faceplates and clamps.

Offline Programming: Steps of the OLP Process

4. Post Processing

- Converting the program into the language of the target robot.
- Drivers that convert the robot programs (Post processeors) need to be created to perform this conversion.
- Post processors are specific to the robot brand, application, and other customer specific requirements for safety, usability, performance, etc.
- It’s rare to develop post processors from scratch;

Offline Programming: Steps of the OLP Process

- Requires lots of work and there are many companies that develop/sell them commercially;
- Require some customization for your specific application and setup.
- About 80\% of the commands are the same for each post processor, but what changes is the local customization for each application, customer specific requirements, and special macro commands that have to be called at the beginning or end of a program.

Offline Programming: Initial Calibration

- Involves calibrating errors between the work cell and virtual model, and updating the virtual model to match.
- The goal:
  - to ensure the offline program is running at 100\%, with no unplanned operator intervention.
- Performed using a tool center point on the shop floor.
- Depending on the application, it’s also possible to calibrate without using a tool center point;
- Industrial robot can be used as its own coordinate measuring machine to determine the relative positions of critical components in the work cell.

Offline Programming: Initial Calibration

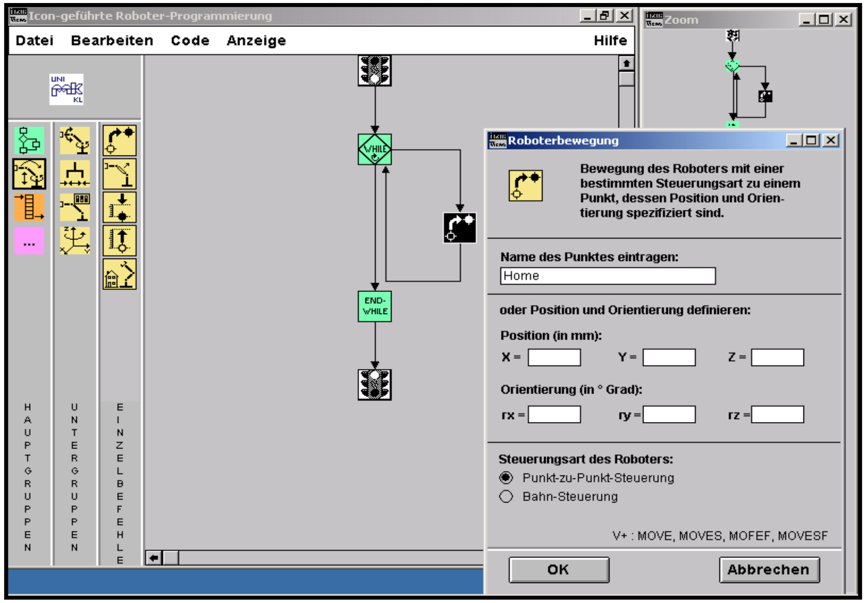
- It’s not always necessary to perform initial calibration offline; it can also be done in the production environment.
- Example:
  - in robotic spot welding, there are usually around 10 to 20 points that need to be programmed in the robot.
- It would be much faster to have the operator calibrate the robot versus doing it offline.
- Similarly, in arc welding, there’s lots of variation between work pieces, and robots use vision sensors to compensate for work piece errors.

Robot Teaching with Visual Components

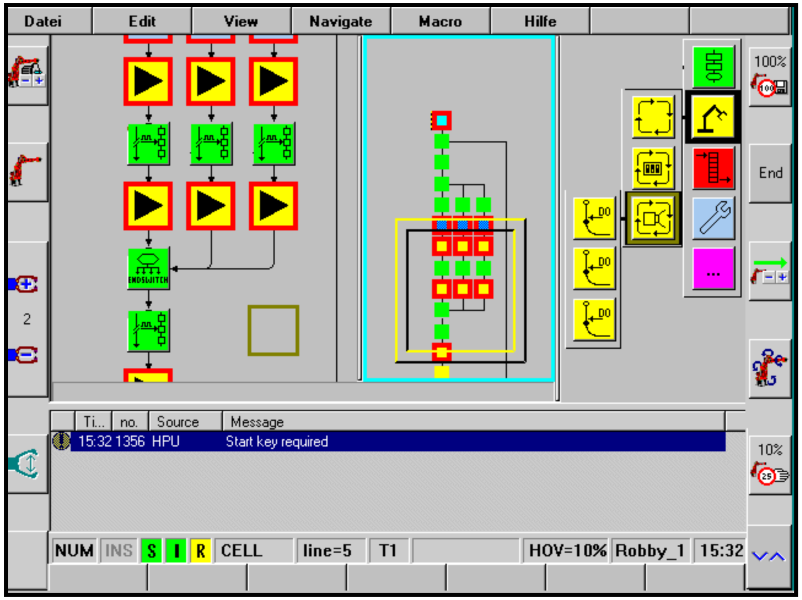
- Visual Components Premium, can create programs that define the actions and routines to be performed by an industrial robot.
- This can be done by virtually teaching the robot positions, or by using the curve-teaching tool to generate paths.
- Use a combination of these methods in developing robot program.
- “curve-teaching tool”, greatly simplifies robot path teaching.

Robot Teaching with Visual Components

- Programing with two or higher-dimensioned structures
- Elements: Graphics, diagrams, icons, animations



Graphical Robot programming, prototype of PAK



Implementation of the PAK prototype for use with the KUKA handheld terminal

Robot Teaching with Visual Components:

- Path Satement:
  - In Visual Components describes a sequence of positions for a robot to follow to complete a given path.
  - Depending on the application, a robot program may have tens or even hundreds of positions that need to be defined.
  - Creating a robot program can take a lot of time, especially if you have to teach all of the positions.

Robot Teaching with Visual Components:

- Curve-teaching Tool
  - Automating the robot path teaching process.
  - Analyze object geometries, makes path predictions, and suggests robot paths.
  - Generating the statements in robot program code.
- OLP is not just developing a robot program; it's a process for designing and planning an entire robot work cell.
- OLP software helps manufacturers to design better production solutions by allowing them to evaluate and visualize the many requirements, constraints, and trade offs in the OLP process.

Robot Teaching with Visual Components:

- Advantages of OLP Combined with Simulation:
  - Robot can be programmed even before it is set up.
  - Design and engineering flaws can be identified early on. Changes can be made on the computer if necessary, avoiding the need for costly on-site modifications.
  - Extensive changes to robot applications are often much easier to implement through offline programming than through direct modification of the robot.
  - In the 3D computer environment, every part of the robot environment can viewed from all sides. In real life, certain perspectives are often concealed or hard to access.

Robot Teaching with Visual Components:

- Disadvantages of OLP Combined with Simulation:
  - Virtual models will (probably) never be able to represent the real world with 100% accuracy. Programs may still need to be altered after they are applied to the real robot.
  - Might take longer overall. Although offline programming reduces downtime of the robot, it means that someone has to spend extra time developing the simulation, as well as testing it on the robot.
  - Can sometimes end up wasting time sorting out simulator issues instead of solving production challenges.

Teaching by Demonstration

- Offers an intuitive addition to the classic teach pendant.
- Involve moving the robot around, either by manipulation a force sensor or a joystick attached to the robot wrist just above the end effector.
- As with the teach pendant, the operator stores each position in the robot computer.
- Many collaborative robots have incorporated this programming method into their robots, as it is easy for operators to get started immediately using the robot with their applications.

Teaching by Demonstration

- Advantages:
  - Quicker than traditional teach pendants.
  - Removes the need for multiple button pressing, allowing the operator to simply move the robot to the desired position.
  - More intuitive than both traditional teach pendants and simulation programs. This makes it simple for operators to learn.
  - No knowledge of programming concepts or being familiar with 3D CAD environments is needed.
  - Very good for detailed tasks which would require many lines of code to achieve the same effect, such as welding or painting of intricate shapes.

Teaching by Demonstration

- Disdvantages:
  - Uses the physical robot for programming.

  → Does not reduce downtime, as much as offline programming.
  - Moving the robot to precise coordinates is not as straightforward as with the other methods.
  - Not so good for tasks which are algorithmic in nature.

ROS: Robot System Operation

- An open-source robotics middleware suite.
- ROS is not an operating system but a collection of software frameworks for robot software development.
- Provide services designed for heterogeneous computer cluster such as:
  - Hardware abstraction
  - Low-level device control
  - Implementation of commonly used functionality
  - Message-passing between processes
  - Package management

ROS: Robot System Operation

- Running sets of ROS-based processes are represented in a graph architecture where processing takes place in nodes that may receive, post and multiplex sensor data, control, state, planning, actuator, and other messages.
- Despite the importance of reactivity and low latency in robot control, ROS itself is not a real-time OS (RTOS).
- It is possible to integrate ROS with real-time code.
- The lack of support for real-time systems has been addressed in the creation of ROS 2, a major revision of the ROS API which will take advantage of modern libraries and technologies for core ROS functionality and add support for real-time code and embedded hardware.

ROS: Robot System Operation

- Software in the ROS Ecosystem can be separated into three groups:
  - language-and platform-independent tools used for building and distributing ROS-based software.
  - ROS client library implementations such as roscpp, rospy, and roslisp;
  - packages containing application-related code which uses one or more ROS client libraries.

ROS: Robot System Operation

- Language-independent tools and the main client libraries (C++, Python, and Lisp) are released under the terms of the BSD license, and as such are open-source software and free for both commercial and research use.
- The majority of other packages are licensed under a variety of open-source licenses.
- These other packages implement used functionality and applications such as hardware drivers, robot models, datatypes, planning, perception, simultaneous localization and mapping, simulation tools, and other algorithms.

ROS: Robot System Operation

- Main ROS client libraries are geared toward a Unix-like system, primarily because of their dependence on large collections of open-source software dependencies.
- For these client libraries, Ubuntu Linux is listed as Supported, while other variants such as Fedora Linux, macOS, and Microsoft Windows are designated experimental and are supported by the community.
- The native Java ROS client library, rosjava, however, does not share these limitations and has enabled ROS-based software to be written for the Android OS. rosjava has also enabled ROS to be integrated into an officially supported MATLAB toolbox which can be used on Linux, macOS, and Microsoft Windows. A JavaScript client library, roslibjs has also been developed which enables integration of software into a ROS system via any standards-compliant web browser.

Coming up Next...

Summary and Examples in Application