Journal of Intelligent and Robotic Systems 5: 49–62, 1992.

Graphical Simulation for Sensor Based Robot Programming

NICHOLAS TARNOFF, ADAM JACOFF and RONALD LUMIA
Robot Systems Division, National Institute of Standards and Technology (NIST), Gaithersburg, MD 20899, U.S.A.

(Received: 14 November 1990; revised 15 August 1991)

Abstract. The programming of robots is slowly evolving from traditional teach pendant methods to graphical Off-Line Programming (OLP) methods. Graphical simulation tools, such as OLP, are very useful for developing and testing robot programs before they are run on real industrial equipment. OLP systems are also used to develop task level programs. Traditional OLP systems, however, suffer from the limitations of using only position control which does not account for inherent robot inaccuracies and dynamic environments. This paper describes our work on improving and supplementing traditional position control programming methods. A baseline OLP system was implemented at NIST's Automated Manufacturing Research Facility (AMRF). Experience gained in implementing this system showed that an effective OLP system must accurately simulate the real world and must support sensor programming to compensate for real-world changes that cannot be simulated. The developed OLP geometric world model is calibrated using robot mounted ultrasound ranging sensors. This measurement capability produces a baseline geometric model of relatively good static accuracy for off-line programming. The graphical environment must also provide representations of sensor features. For this specific application, force is simulated in order to include force based commands in our robot programs. These sensor based programs are able to run reliably and safely in an unpredictable industrial environment. The last portion of this paper extends OLP and describes the functionality of a complete system for programming complex robot tasks.

Key words. Off-line programming, robot programming, simulation, computer graphics, sensor programming, sensor modeling, robotics, world modeling, calibration.

1. Introduction

A graphical Off-Line Programming (OLP) system provides a simulation of the real world for programming robotic tasks without the use of the real equipment. This approach is potentially very beneficial in improving the safety and cost-effectiveness of robotic workstations. Human safety is significantly improved by limiting physical interaction between the operator and a powered robotic workstation. OLP also protects the equipment, by using simulations to check for robot programming errors that might otherwise have catastrophic results. In addition, programs generated off-line, reduce robot down time which ordinarily occurs when programs are generated using the teach-pendant method.

OLP systems are recognized as very useful for rapid prototyping and debugging of complex robotic workstations and several commercial systems are available for that purpose [1]. However, commercial OLP applications rarely produce the final robot control programs. Despite the significant benefits of using OLP as a production tool, OLP largely remains a prototyping tool for one major reason. Traditional OLP

programs, based on position controls, do not compensate well for inherent robot or world model inaccuracies. We are, therefore, working to improve off-line world model accuracy and, most importantly, support sensor based robot programming.

OLP is also used in advanced robot control to program at a level of control called the task level. Task level commands target the actions of one manipulator on one complete object such as 'deburr part A' [2]. This level of abstraction is very similar to the way in which humans interact with their environment. OLP, therefore, is an intuitive tool for programming at the task level.

While our OLP work is primarily geared toward industrial applications, it is very important to understand how our work relates to accomplishments in the field of advanced robot control at the task level. In general, OLP is a graphical tool for programming complex robotic tasks that are difficult to program only textually. The inherent limitations of the human mind require the assistance of tools to store, recall and interact with large amounts of information simultaneously. OLP is such a tool for manipulating mostly geometric information to arrive at a reliable robot program. The complexity and volume of information involved at the task level, however, is one of the major reasons little progress exists at this level of control. Research of task level functions is very sparse as compared to research of the lower control levels such as hybrid force/position or PID control. Traditional task level research is concerned with automating either top/down (e.g., assembly plans) or bottom/up (e.g., compliant motion) functions. These two approaches are typically used to solve three types of robot independent actions; path planning (gross motion), grasping and mating (contact) [3]. In contrast, it is important to understand our more pragmatic approach which involves iterating between top/down and bottom/up design. We are building a tool that will provide a coherent environment for programming, testing and simulating task level functions for the purpose of automating those functions.

Our current research focuses on solving the inherent inaccuracies of an artificial and remote world model and the limitations of traditional position control programs. These two issues are discussed and related to the OLP implementation at the Cleaning and Deburring Workstation (CDWS) in the AMRF.

2. Application

An OLP system was implemented at the Cleaning and Deburring Workstation (CDWS) in the AMRF [4] in order to study OLP issues. The CDWS, shown in Figure 1, consists of two robots for deburring, buffing and cleaning of machined workpieces [5]. The OLP project involves programming the Unimate 2000 robot for buffing workpieces. Both the Unimate and the Puma 760 robots perform part handling to and from the supply trays, deburring vise and washer/dryer. In addition the Puma 760 performs deburring operations. The workstation control scheme allows for concurrent tasks. Conflicting tasks that could lead to a collision are prioritized and one robot at a time is allowed to operate in the common work area. An Automated

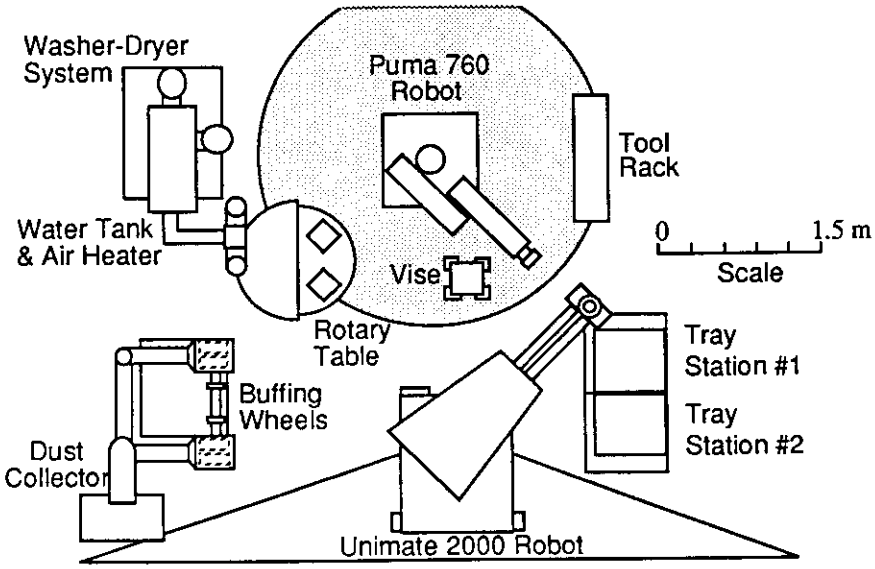


Fig. 1. The cleaning and deburring workstation at the AMRF.

Guided Vehicle (AGV) delivers and retrieves workpieces between workstations and the material buffering system.

The goal of OLP at the CDWS is to program the gross and, more importantly, fine motions of the Unimate 2000 buffering process. While the application area is very specific, the intent for the project is to develop generic capabilities of OLP which are transferable to any robot programming task [6]. In fact, this generic OLP technology is being applied to a new Advanced Deburring and Chamfering System (ADACS) under development in the AMRF.

The initial OLP implementation completed two years ago provided an integrated baseline system. Using this OLP system first involves transferring a CAD wireframe model of the workpiece from a commercial CAD system (CADDS from Computer-Vision) to a commercial OLP system (CimStation from Silma, Inc.) via the Initial Graphics Exchange Specification (IGES) [7]. IGES 3.0 is a standard file format for passing CAD data between systems. The imported IGES workpiece model is added to the OLP world model. The operator then models the remaining workstation components within the CimStation graphical simulation environment. Robot trajectories are created by defining frames on the graphics screen's rendering of the simulated workstation using a mouse and an on-screen pointer. The operator programs robots in simulation by the direct manipulation of the robot model using the computer mouse. The robot can also be movd by entering absolute or relative tool coordinates and orientations. The user then generates programs textually and graphically to operate and synchronize all workstation devices such as robots, material handlers, fixtures and tooling. The resulting programs are tested and revised based on collision detection and other simulated criteria such as sensor input. The final

simulation is saved and the Cartesian or joint space robot trajectories are post-processed for downloading to the Unimate VAL II robot controller. Our experience in implementing and using this baseline OLP system has lead to our current research in trying to improve the accuracy and scope of the world model.

The major impediment to effective OLP is the inconsistency between an OLP system's world model and the real world. The world model acts as a link to the real world for the OLP system and eventually the control system [8]. The first part of the solution lies in improving the world model's geometric fidelity using calibration. The second part of the solution lies in having a world model that facilitates the off-line programming of sensory interactive tasks.

3. OLP Enhancements

3.1. CALIBRATION

Graphical simulation is an intuitive and flexible method of programming robots. The world models on which these simulations are based must be accurate. An effective OLP system must first establish a geometric and kinematic world model of baseline accuracy. This modeling accuracy is not a visually noticeable part of the graphical simulation but may affect the reliability of a program whose instructions depend on a minimum accuracy. For example, a gross motion followed by a guarded motion will have catastrophic results if the simulated distance reserved for the guarded motion is less than the positive error incurred during the real gross motion. The resulting gross motion will cause a collision between the end effector and the workpiece before being able to switch to guarded motion. Another interesting problem arises when the world model error at a robot's work volume boundary results in an invalid robot command to move outside of its work volume. To help in maintaining an accurate world model and avoid problems such as the two just mentioned, the CAD model of the workpiece is imported directly into the simulation environment. This method preserves the model accuracy of the workpiece as designed in CAD. For devices such as the robot or a fixture, the model must be calibrated against the real devices in order to account for significant geometric changes due to wear for example.

An array of contact position transducers mounted on three normal surfaces of a fixed jig is often used to perform calibration [9]. Robot mounted non-contact calibration, however, is more flexible. This calibration method is made possible by compact ranging sensors such as ultrasound or recently developed laser triangulation sensors [10]. An instrumented gripper was built that consists of two ultrasound ranging sensors on each gripper finger. The ultrasound ranging sensors are aimed normal to a fixed hard surface and the return sound pulse delay is recorded and converted to a distance. The sensor control electronics are integrated with the CDWS controller and enable automatic data retrieval and manipulation [11]. The calibration gripper is equipped with a quick change adapter enabling the Unimate 2000 robot to attach or detach the calibration gripper when manipulates a target when the object

under calibration cannot be used itself as an ultrasound measurement target. The RPS-300 Migatron ultrasound sensor's reliable range is between 3 and 14 inches and ± 6 degrees off the normal. The accuracy of the sensor is specified to be 0.76 mm. Our implementation of the sensors resulted in a resolution and repeatability of 1.1 and 2.0 mm, respectively. The Unimate 2000 robot, which performs the calibration, is repeatable to within ± 2.5 cm. Compared to the traditional fixed calibration device, our robot mounted ultrasound calibration scheme is non-contact, fast, flexible, and automated. This implementation, however, can measure only along two normal axes and around one axis. Measurement of the complete three-dimensional position and orientation is achieved most effectively by mounting three ranging sensors in a triangle pattern, repeated on three normal surfaces such as that of a cube [12]. This configuration is also easily implemented as a non-contact robot mounted device.

World model calibration using ultrasound sensors consists of two steps; measuring the pose (position and orientation) of objects relative to the robot; and generating a three-dimensional scalar map of robot end effector positioning erros in a volume surrounding a work area of interest. Both of these calibration procedures transform primarily joint space errors into Cartesian space errors and are assumed valid for small distances and angles. Calibration results, therefore, are valid only in the vicinity of the calibrated object or volume. Measurements of object poses are uploaded to the OLP system for off-line calibration of object locations and robot trajectories, as illustrated in Figure 2. An error map, on the other hand, consists of several pose measurements within a volume of interest such as around a fixturing device. VAL II uses the on-line error map to compensate for positioning errors. Positioning errors for the Unimate 2000 in the vicinity of the CDWS vise, for example, vary between 10 and 20%.

Static calibration only reduces the world model's static errors within the limit of the calibration sensor's accuracy. Calibration sensors have a finite accuracy, and dynamic

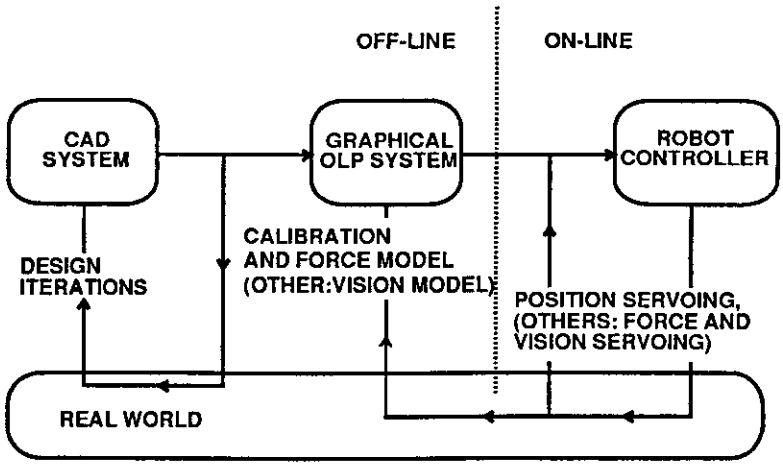


Fig. 2. OLP implementation.

changes in geometry (for example due to temperature fluctuations or wear) are very difficult to model and/or track. On-line, real-time use of sensors is needed to compensate for an unpredictable environment unaccounted for by position control. Reliable OLP programs, therefore, must be sensor based.

3.2. FORCE MODEL

Sensory-interactive programming features, such as force models, are necessary to overcome minor perturbations in a robot work environment. The baseline OLP system built on top of CimStation's simulation capabilities limits the user to robot position control without sensory-interaction. Position control alone, while useful, is not sufficient for robust off-line programming of robots. The reason is that simulations will never attain perfect geometry, kinematic or dynamic replicas of the real world. Due to robot positioning errors and world model inaccuracies, the location of the CDWS's real buffing wheel, relative to the robot, differs from the OLP representation. Therefore, sensors must be used to compensate for errors in the workstation world model and inaccuracies associated with position control of robots [13]. Sensor programming of robot tasks uses sensor feedback from the environment, or environmental models, to account for these errors. An effective OLP sytem must be capable of generating sensor programs based on environmental models.

Environmental models are developed using sensors such as proximity, vision or force/torque sensors. The goal of environmental modeling is to quantify certain features in a subset volume of the workstation world model. Features may include obstacles in robot pathways, textures or shapes of various parts in an assembly, or forces generated on the robot by dynamic (but predictable) events in the workstation. These environmental models are then incorporated into the OLP process to enhance the world model by providing detailed information about regions of interest within the world model. An example of environmental modeling is the world model calibration via ultrasound proximity sensing introduced above. Development of a vision sensing model, to compensate for unknown workpiece placement, is a future goal of this project. The following sections of this paper discuss recently completed work on environmental modeling of forces at the AMRF's Cleaning and Deburring Workstation.

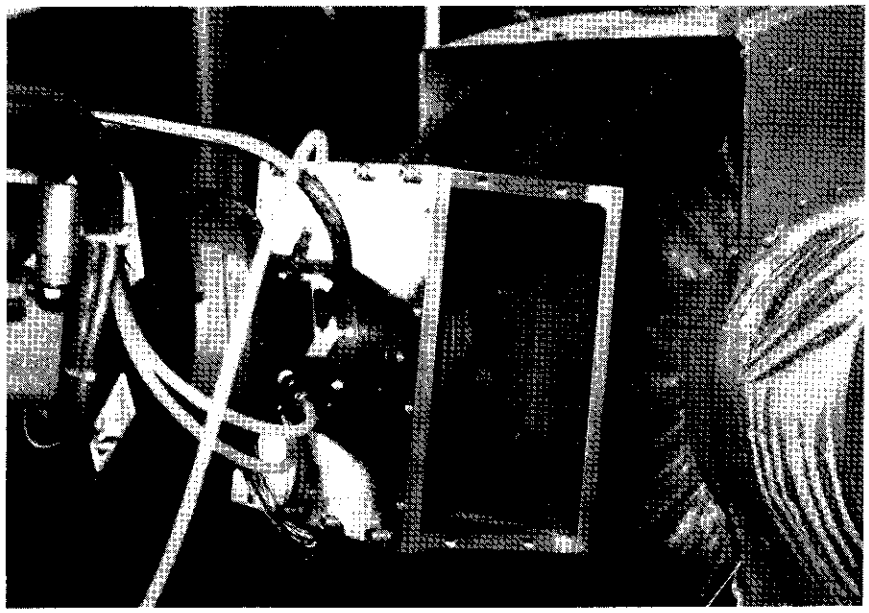
The CDWS application to buffing machined parts was suitable for development of an environmental force model because it involves physical interaction between a robot and spinning cotton wheel. The goal of this automated buffing application is to maintain a desired buffing force on a workpiece held by the robot. The spinning cotton wheel produces a gradual increase in force with depth into the wheel. Position control errors in workpiece placement into the spinning buffing wheel can result in inferior buffing or dangerously high forces on the workpiece and the robot. The environmental model, developed and implemented at the CDWS, was a force/position model of the buffing wheel. The force/position model was then incorporated into our OLP environment so that robot trajectories, programmed off-line, could be tolerant of the actual forces incurred by a robot (with workpiece) during buffing.

The force/position model relates actual buffing forces on a workpiece to position of the workpiece within the buffing wheel. The actual buffing forces were gathered experimentally using the force sampling end-effector shown in Figure 3(a). The force sampling end-effector was designed and built at NIST specifically to obtain the relationship between buffing forces and relative position within the spinning buffing wheel. The force sampling end-effector is a box-like tool which contains a force/torque sensor. One of three different size probes is mounted to the force/torque sensor and coincides with a hole in the faceplate. The faceplate is large enough to appear as an infinite plate to the spinning wheel and the probe tip is flush with the faceplate to avoid vertical (y axis) forces on the probe. The force-sampling end-effector is also equipped with a quick-change adapter enabling the Unimate 2000 to attach/detach the end-effector when a new force/position model becomes necessary. Force readings were taken at 12.7 mm intervals both vertically (y axis) and horizontally (x axis), relative to the buffing wheel, and at 5 mm intervals into the buffing wheel (z axis). Surface plots generated at each (y) level of entry into the buffing wheel are illustrated in Figure 3(b). These surface plots show the wave-like representation of the forces incurred during buffing as the probe penetrates the wheel. The buffing wheel's force/position model consists of a three-dimensional compilation of these surface plots. Linear interpolation was used to determine forces at locations between the experimentally obtained nodes in the force/position model. The buffing wheel's force/position model is applied off-line during simulation of the buffing trajectory to monitor the expected forces on the workpiece. Given a known workpiece position within the buffing wheel, the force/position model outputs the expected force that the workpiece will feel based on the experimental data of that buffing wheel.

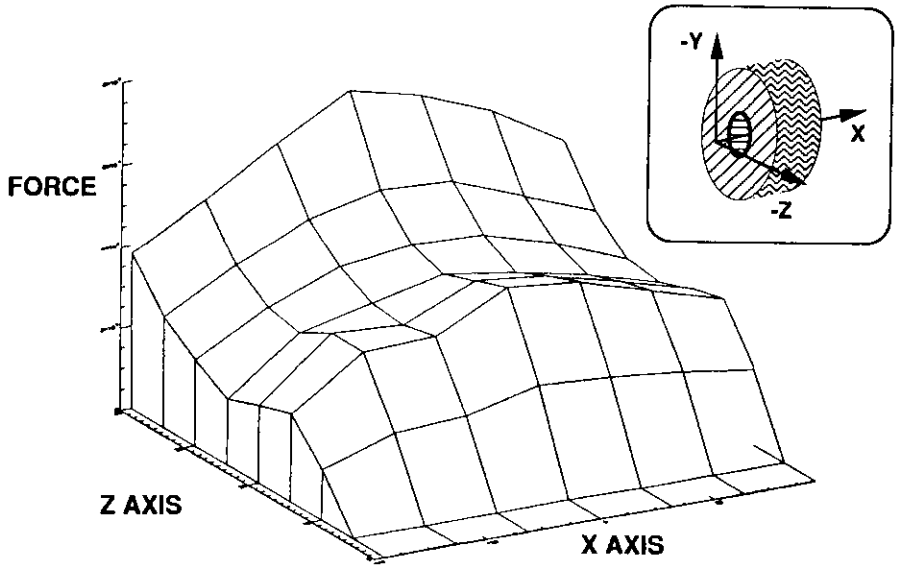
It should be noted that the force/position model is intended as an intermediate step toward on-line, real-time, sensor based robot programming. For on-line compensation of inherent world model inaccuracies and real world changes, real-time force feedback to a sensory interactive controller is necessary. This would allow for robot programming according to the expected force/position model, and then actively serving to the programmed force.

3.3. PROGRAMMING INTERFACE

The most visible feature of a graphical OLP system is the operator interface. Our sensory-interactive OLP system was built on top of the simulation capabilities of CimStation. The graphical representation of the CDWS with the buffing wheel's x-y-z axis force display (upper left) is shown in Figure 4. Using this graphical simulation, the user can visually verify that the robot correctly executes its programmed trajectory. In addition, programs using environmental models can report pertinent data from that model to the user graphically, textually or both, as with the force/position model. Environmental models give the user more insight into robot interaction with the environment, aiding in the development of useful and safe robot programs. In this case, the modelled environment is the buffing wheel. The force/position model



(a)



(b)

Fig. 3(a). Force sampling end-effector.

Fig. 3(b). Surface plot of forces ($y = 0$).

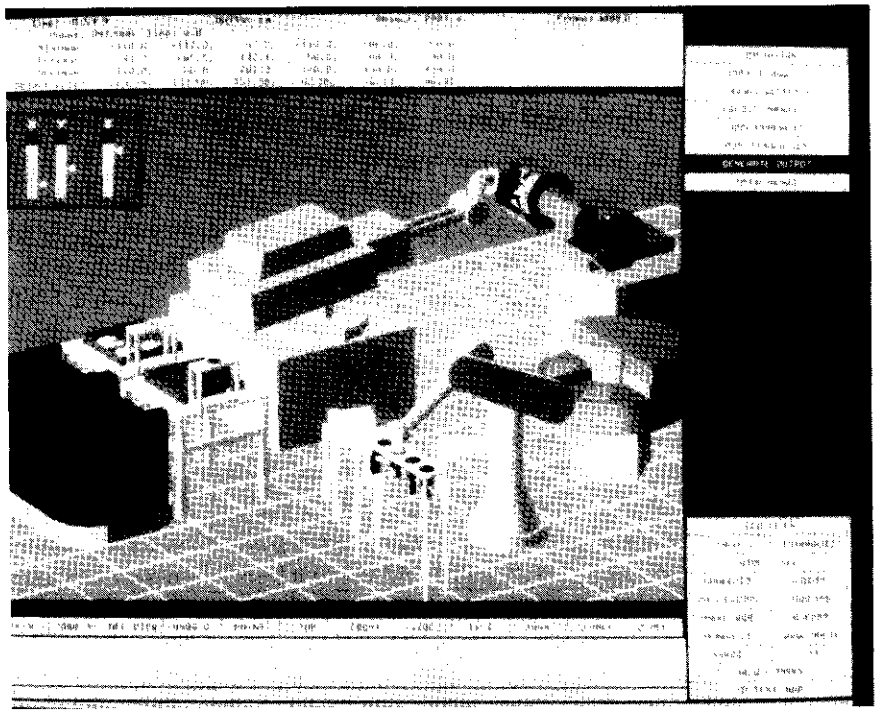


Fig. 4. CimStation simulation of CDWS with force display.

represents the buffing forces acting on the workpiece at various locations within the buffing wheel. The bar graph force display allows the user to adjust the programmed buffing trajectory, off-line, according to the allowable forces of the task, not just by position. The force/position model of the buffing wheel is applied during the robot's simulated buffing trajectory. The position of the workpiece within the simulated buffing wheel is known and is monitored continuously using CimStation's 'object monitor' feature. The workpiece face is divided into smaller facets, in simulation, which are each tracked for collisions with the buffing wheel. Highlighting of the simulated workpiece and buffing wheel signifies a collisions between any facet on the face of the workpiece and the buffing wheel.

The position of each facet within the simulated buffing wheel corresponds to a force within the buffing wheel's force/position model. According to the force/position model, the expected forces on each facet colliding with the buffing wheel are summed over the whole workpiece. The total force on the workpiece is represented (x, y, and z axis) in the bar-graph force display (shown in Figure 4) which advises the user of excessive or inadequate forces incurred during a particular simulated buffing trajectory. The expected forces in the x, y, and z directions are monitored graphically using color coded scales. The color code is as follows: a green or yellow force is acceptable, a red force is unacceptable or dangerous. The expected forces on the workpiece can

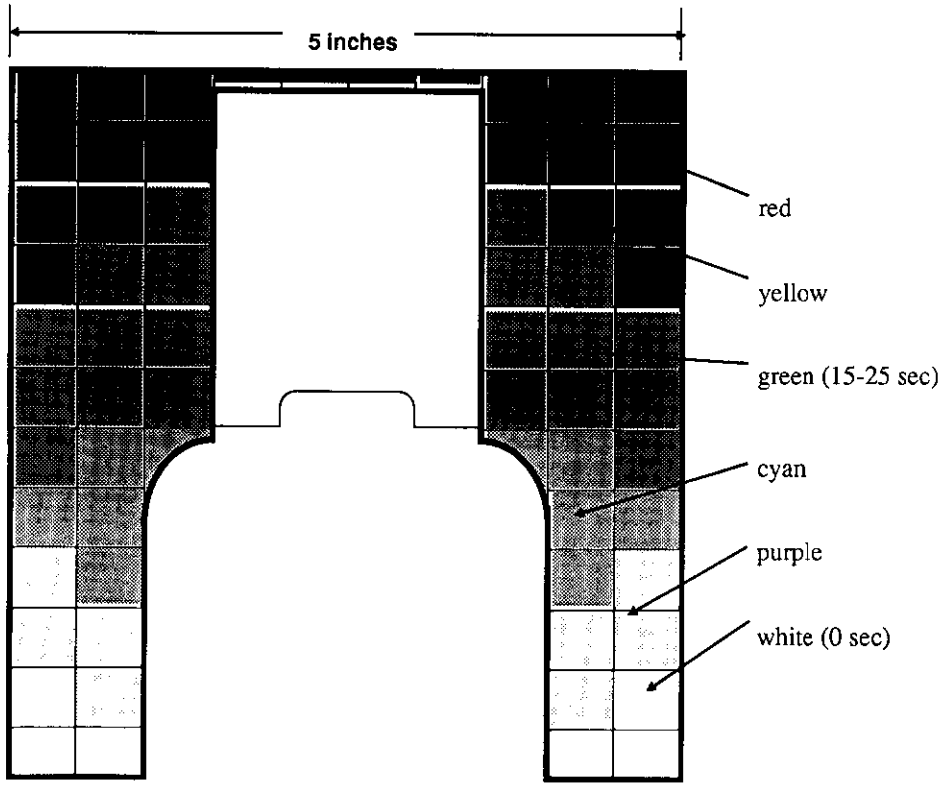


Fig. 5. Mosaic pattern on simulated part face.

also be monitored numerically. Therefore, the force display allows the user to visually examine the expected forces associated with a programmed trajectory and adjust the trajectory as necessary.

Another trajectory evaluation tool, developed for off-line use, is the simulated mosaic pattern shown in Figure 5. The mosaic pattern represents a time-based evaluation of buffing trajectories. Each facet on the face of the simulated workpiece is colored according to the length of time it is exposed to the buffing forces within the wheel. The mosaic pattern represents the time exposure of each facet over the entire buffing trajectory. The mosaic color code is similar to the force display. Facets which are red in color indicate excessive buffing time. Facets which are yellow and green indicate acceptable buffing time. Other colors indicate inferior buffing and white facets are untouched by the buffing wheel. The mosaic pattern distinguishes the areas of most concentrated buffing for a given trajectory, allowing the user to adjust a particular trajectory according to the shape of the workpiece or the desired buffing pattern.

The user can also develop programmed trajectories which isolate sections of the workpiece for safety reasons. For example, if the leading (top) edge of the actual workpiece were to contact the spinning buffing wheel, excessive vertical (y) forces

would immediately slap the workpiece from the gripper. Dangerous situations can be avoided using the mosaic pattern, off-line, to identify and change trajectories which include contact between the simulated workpiece's leading edge and the buffing wheel.

The mosaic tool is used in conjunction with the calibration apparatus and the force/position model to augment the base-line graphical simulation of the CDWS. Together, these environmental models comprise a robust sensory-interactive OLP system which is capable of utilizing sensors to calibrate, model and reflect the environment in which a robot must interact to conduct tasks. This OLP system, with sensor interaction, is used to develop robot programs that are tested to be safe and effective prior to downloading to actual equipment. Therefore, it provides a necessary link in improving the quality of automation while reducing risk to costly machinery in the workstation.

4. Conclusions

4.1. SYSTEM FUNCTIONALITY

Sensor based programming and the corresponding support from off-line graphical simulation is necessary for effective off-line programming of complex robot tasks. Traditional position control of robots is one of the greatest impediments to expanded robot use. Roboticists know that current actuator and sensor technology are capable of supporting complex sensory-interactive tasks, but the complexity of programming such tasks requires the tools and techniques described in this paper. Our future work will investigate tools that provide graphical three-dimensional and two-dimensional interfaces using strong underlying models as shown in Figure 6. The system will support a fast iterative process for the programming of sensory interactive robot tasks

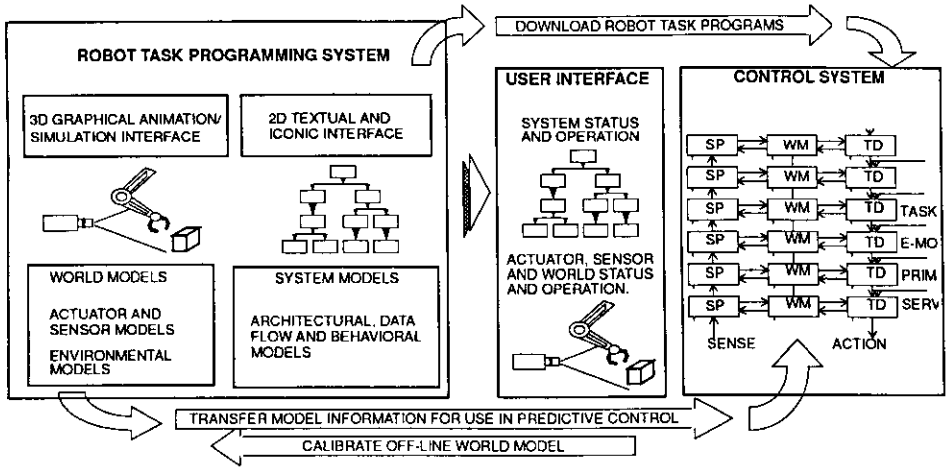


Fig. 6. Robot task programming system and evolution of modules.

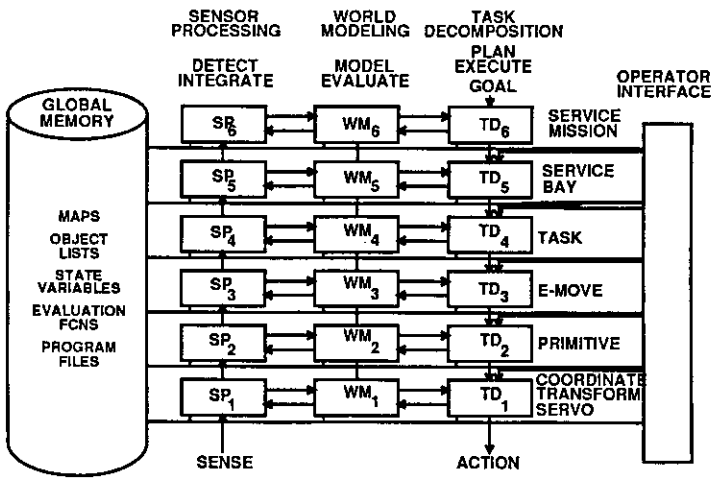


Fig. 7. The NBS hierarchical control system architecture [2].

[14]. Our implementation of model calibration and sensor simulation are clear examples of a systematic progression towards this goal.

4.2. SYSTEM ARCHITECTURE

The NASREM hierarchical control architecture as shown in Figure 7 is very important because it provides a consistent context for our robotic work. The NASREM philosophy has evolved significantly and the current thinking is detailed in [2, 15]. Figure 7 is a simplified illustration of the NASA/NBS Standard Reference Model control structure. The structure has six decomposition levels (Mission, Service Bay, Task, Elementary-Move, Primitive and Servo) and three vertical hierarchies (Sensory Processing, World Modeling and Task Decomposition). Communication is possible between any two modules in the structure by way of the global distributed database. The world model modules facilitate the transfer of data to and from the database thus decoupling Sensory Processing and Task Decomposition. Data communication is primarily horizontal and much more voluminous than the primarily vertical communication of commands and status feedback. Successive decompositions of tasks across both time and space result in a tree structure of task decomposition modules. Each task decomposition module assigns jobs, plans and executes. Planning is more active at the higher task decomposition levels while modules at the lower levels do more process execution.

The role of OLP in the development and operation of a control system is evolutionary. First, the OLP system is a tool for programming the desired control, feedback and data functions of the hierarchical control system. Once the control system is ready for operation, the off-line world models are transferred to the control system for use in predictive control. The OLP graphical interface now provides a user interface for

monitoring as well as operating the automated control system. This evolution is actually an iterative process as the control system continues to develop and improve with operation.

To successfully achieve this process, the OLP system must be modular and include a partial image of the hierarchical control structure [16]. The system must also include simulated actuator, sensor and environmental events. These components form the basis for a graphical OLP system used primarily at the task level for programming and operation.

4.3. FUTURE DEVELOPMENTS

A three-dimensional graphical interface is necessary for manipulating geometric and sensor feedback information. However, it is not well suited as a two-dimensional interface for architectural, data flow and behavioral analysis of task level robot programs. These functions require a different user interface to support the programming of complex robot tasks including sensor based and exception handling functions. Such analysis tools can provide a centralized interface for iterating between bottom/up and top/down program development. For example, constraints on the kinds of actuators and sensors that are available will determine the contents of the world model (bottom/up). On the other hand, the logic and data flow behind a sequence of task level actions must be analyzed from the task level down to the simulated actuator and sensor results (top/down). The task decomposition, data flow and behavioral analysis functions are currently done using manually intensive methods such as a text editor. This programming method significantly limits the user to relatively simple task level programs and hence impedes research. This very same problem is recognized in the computer science field and Computer Aided Software Engineering (CASE) tools are being developed to analyses complex software and hardware systems. CASE concepts are also beginning to appear in robot control work [17, 18]. More importantly, certain CASE prototyping tools are beginning to support a variety of control system paradigms that enable the control systems engineer to implement a better system rather than being constrained by rigid analysis methods. Those same tools correctly place a heavy emphasis on graphical interaction as a means of programming and analyzing a system.

Our graphical OLP system will be the three-dimensional component of a robot programming 'CASE tool'. This software engineering tool will enable the application/programming expert to quickly program and test new domain specific (e.g., deburring) tasks such as one related to a new sensor capability. In addition, the expert will be responsible for grouping tested robot tasks into simplified application specific templates. The expert will further simplify the programming process for the operator by automating appropriate programming functions. The resulting application template provides the factory operator/non-programmer with a simple set of commands and rules as an interface to an otherwise complex robot controller. These tools enable the factory operator to quickly and intuitively reprogram a workcell.

This approach to programming robots for complex tasks provides a consistent, modular and evolutionary programming method from research in task level programming and sensor fusion through factory operations. The secondary but very important result of this process is the structuring and capturing of expert knowledge. The formerly daunting process of having to develop task level programs is now partitioned into a manageable and traceable subset of capabilities.

References

Important Note: The National Bureau of Standards (NBS) became the National Institute of Standards and Technology (NIST) on August 23, 1988.

1. Gini Maria, 1987, The future of robot programming, *Robotica* **5**, 235–246.
2. Albus, J.S., McCain, H.G., and Lumia, R., 1989, NASA/NBS standard reference model for telerobot control system architecture (NASREM), NBS Technical Note 1235, National Institute of Standards and Technology, Gaithersburg, MD. Also available as NASA document SS-GSFC-0027.
3. Latombe, J.C., 1988, Toward automatic programming, *Proc. Control and Programming in Advance Manufacturing Systems Conference*, IFS Pub, 85–102.
4. Simpson, J.A., Hocken, R.J., and Albus, J.S., 1982, The automated manufacturing research facility of the National Bureau of Standards, *J. Manufacturing Systems* **1**(1), 17.
5. Murphy, K.M., Norcross, R.J., and Proctor, F.M., 1988, CAD directed robotic deburring, *Second International Symposium on Robotics and Manufacturing Research, Education and Applications (ISRAM)*, Albuquerque, NM, 959–965.
6. Lumia, R., 1988, CAD-based off-line programming applied to a cleaning and deburring workstation, *NATO Advanced Research Workshop on CAD Based Programming for Sensor Based Robots*, Il Ciocco.
7. Craig, J.J., 1987, Silma CimStation technical overview – Version 3.0", Silma Inc., Los Altos, California.
8. Kent, E.W. and Albus, J.S., 1984, Servoed world models as interfaces between robot control systems and sensory data, *Robotics* **1**, 17–25.
9. Nowrouzi, A., Kavina, Y.B., Kochekali, H., and Whitaker, R.A., 1988, An overview of robot calibration techniques, *The Industrial Robot* **12**, 229–232.
10. Dietrich, J., Hirzinger, G., Heindl, J., and Schott, J., 1989, Multisensory telerobotic techniques, *NATO Advanced Research Workshop 'Traditional and Non-Traditional Robot Sensors'*, Maratea, Italy.
11. Bostelman, R., 1989, Electronic design of the infrared/ultrasound sensing for a robot gripper, National Institute of Standards and Technology, NISTIR 89-4223.
12. Riley, D.L., 1987, Robot calibration and performance specification determination, *Proc. 17th Symposium on Industrial Robots* **10**, 1–15.
13. Crane, C. and Duffy, J., 1981, An interactive animated display of man-controlled and autonomous robots, *Proc. ASME Computers in Engineering Conference*, Chicago, IL.
14. Volz, Richard A., 1988, Report of the robot programming language working group: NATO Workshop on Robot Programming Languages, *IEEE J. Robotics Automat.* **4**, 86–90.
15. Albus, J.S., 1990, A theory of intelligent systems, *Proc. 5th IEEE International Symposium on Intelligent Control*: Philadelphia, PA.
16. Tarnoff, N. and Lumia, R., 1988, The role of off-line robot programming in hierarchical control, *2nd Internat. Symposium on Robotics and Manufacturing Research, Education and Applications*, ISRAM, Albuquerque, NM, 967–974.
17. Angermuller, G., Niedermanayr, E., and Roth, N., 1989, Off-line programming and simulation of flexible assembly, *Assembly Automation* **9**.
18. Ghani, N., 1988, Sensor integration in ESPRIT, *2nd IFAC Symposium on Robot Control SYROCO '88*, 323–328.