Informed peg-in-hole insertion using optical sensors

Eric Paulos John Canny

Department of Electrical Engineering and Computer Science
University of California
Berkeley, CA 94720

Abstract

Peg-in-hole insertion is not only a longstanding problem in robotics but the most common automated mechanical assembly task.[1] *In this paper we present a high precision, self-calibrating peg-in-hole insertion strategy using several very simple, inexpensive, and accurate optical sensors. The self-calibrating feature allows us to achieve successful dead-reckoning insertions with tolerances of 25 microns without any accurate initial position information for the robot, pegs, or holes. The program we implemented works for any cylindrical peg, and the sensing steps do not depend on the peg diameter, which the program does not know. The key to the strategy is the use of a fixed sensor to localize both a mobile sensor and the peg, while the mobile sensor localizes the hole. Our strategy is extremely fast, localizing pegs as they are in route to their insertion location without pausing. The result is that insertion times are dominated by the transport time between pick and place operations.*

1 Introduction

We describe a method for performing accurate insertion operations using simple optical sensors. A key to the method is the use of one sensor to compute the position of the other, which eliminates the need for prior set-up and calibration. The sensors are simple optical beam sensors, which respond to the presence or absence of an object along the beam line. These sensors require relative motion between object and sensor, and we claim that this is a feature. Parts are always being transported between various positions in the workspace during assembly, and beam sensors can perform accurate localization as the part passes by, without the need to pause over the sensor. Other applications of these sensors in the broader setting of RISC robotics are presented in work by Canny and Goldberg.[2] The method has been implemented as we describe later in this paper, and has proved highly reliable.

Typical calibration methods in use today involve special robot actions and costly calculations, resulting in a reduction of the throughput for the entire assembly system. By correcting position errors in the assembly workcell dynamically during normal assembly operations, the various manipulators and other parts attached or grasped by the robots can vary on each assembly execution and still perform successful insertions. This makes our approach well suited for an automated manufacturing system where it is common to find part differences and changing robot position errors over successive assemblies.

1.1 Previous work on insertion

The inherent uncertainty in robots has led several others to propose methods for performing robust assembly operations. Most notable is the 1984 work by Lozano-Pérez, Mason, and Taylor.[3] Their approach, referred to as LMT explicitly models sensor and control uncertainty and defines guaranteed strategies under these models. Erdmann[4] in 1986 expanded LMT and made several of the major steps computational. Later work by Donald[5] added error recovery to LMT. A general algorithm for full LMT was presented by Canny.[6] But planning with this kind of uncertainty is very expensive in the worst case[7], and LMT has not been applied beyond two dimensions.

Much work has dealt with insertion using mechanical compliance, in particular, based on the Remote Center of Compliance (RCC), introduced by Whitney[8] in 1982 and later developed by Peshkin[9] in 1990. RCCs are somewhat specialized though, and active compliance has been studied as a way to deal with a variety of assembly scenarios, in particular by Khatib[10].

The present work complements this earlier work by expanding the set of situations where assembly can be done without the need for compliance. Our dead-reckoning strategy is as fast as RCC insertion, and as flexible as active strategies, while avoiding unnecessary contact between parts and the possibilities of binding or scratching.

1.2 Previous optical sensor calibration work

Optical sensing has been used for calibration since 1988 by Everett and Ives.[11] Their approach uses optical sensors and special precision spheres to perform calibrations. They mount these spheres onto a special calibration block, calibrate the robot's position, and then remove the block. Their off-line calibration scheme is reported to be accurate to 58 microns, while our online insertion scheme works at 25 microns.

Work in 1993 by Prenninger *et al.*[12] uses a laser optical system for calibration. They use a laser/mirror system to triangulate the position of the end-effector of a robot. Their system is quite complex, involving a laser, a mirror mounted on the end-effector, a high precision universal joint, and a CCD vision system. The reported accuracy is about 50 microns.

Beam sensors can be used to do recognition as well as localization, as described by Wallack, Canny, and Manocha.[13] We plan in future to combine that work with the present, to produce a general precision insertion algorithm for arbitrary parts.

2 RISC Robotics

RISC robotics[2] (Reduced Intricacy in Sensing and Control) is a new philosophy in robotics aimed at merging robotic and hard automation technologies. RISC robotics maintains a small set of simple, inexpensive, and accurate primitive tools that can be easily interchanged and/or combined to perform more complex tasks.

The RISC robotics philosophy can be applied to many areas of manufacturing robotics. For example, RISC grasping combines grasping with simple two and three fingered grippers with traditional fixturing with clamps and vices.

RISC sensing employs simple but precise sensor elements to build up composite sensors that can localize and recognize arbitrary objects from a library.

A side-effect of this design philosophy, which leads to few degrees of freedom and low-dimensional sensor spaces, is that the algorithms for manipulation and sensing in RISC robotics are usually simple, very accurate and very fast.

3 Self-Calibrating Strategy

Our strategy localizes all of the elements necessary for the assembly operation relative to a coordinate frame defined by a fixed optical sensor array in the workcell. The optical sensor array, shown in Figure 1, consists of three oppositely mounted transmitters-receivers sensor pairs. These optical sensors respond to beam breakage, hence object presence or absence along each of the beam lines. This cross beam configuration is mounted on a fixed platform in the workspace and used to determine both sniffer and peg locations relative to its beams.

We also use a reflective sniffer sensor as shown in Figure 2. This reflective sniffer sensor differs from the cross beam sensor by having both its transmitting and receiving sensor elements in the same lens. By mounting this sensor vertically at the end of one of the robot modules we can detect position information for various holes in the workspace by detecting the presence or absence of a reflected optical signal.

Although we discuss our system using two separately mounted elements, a gripper end-effector and a sensor end-effector, we can implement the same system on a single end-effector by mounting an optical sniffer sensor on a gripper end-effector as shown in Figure 3. Using this setup, we simply measure the relative offset of the sniffer sensor from the peg in the gripper using the cross beam sensor arrangement. Using that relative information, we simply remove the offset after we localize the hole using the attached offset sniffer sensor and then perform the insertion.

We will first outline our peg-in-hole insertion strategy followed by detailed descriptions of each localization strategy in Sections 3.2 and 3.3.

3.1 Strategy outline

The first part of our strategy is to localize the sniffer used for hole sniffing with the cross beam sensor configuration. As the tip of the sniffer sensor is passed through the cross beam configuration, beam breakages are sensed and position information recorded. With the now localized sniffer sensor, we move to the assumed hole location and using the sniffer sensor, localize the hole using the pattern from Figure 6. From this sniffing operation we know the relative position of the hole with respect to the cross beam sensor configuration. In the final step we pass the peg through the

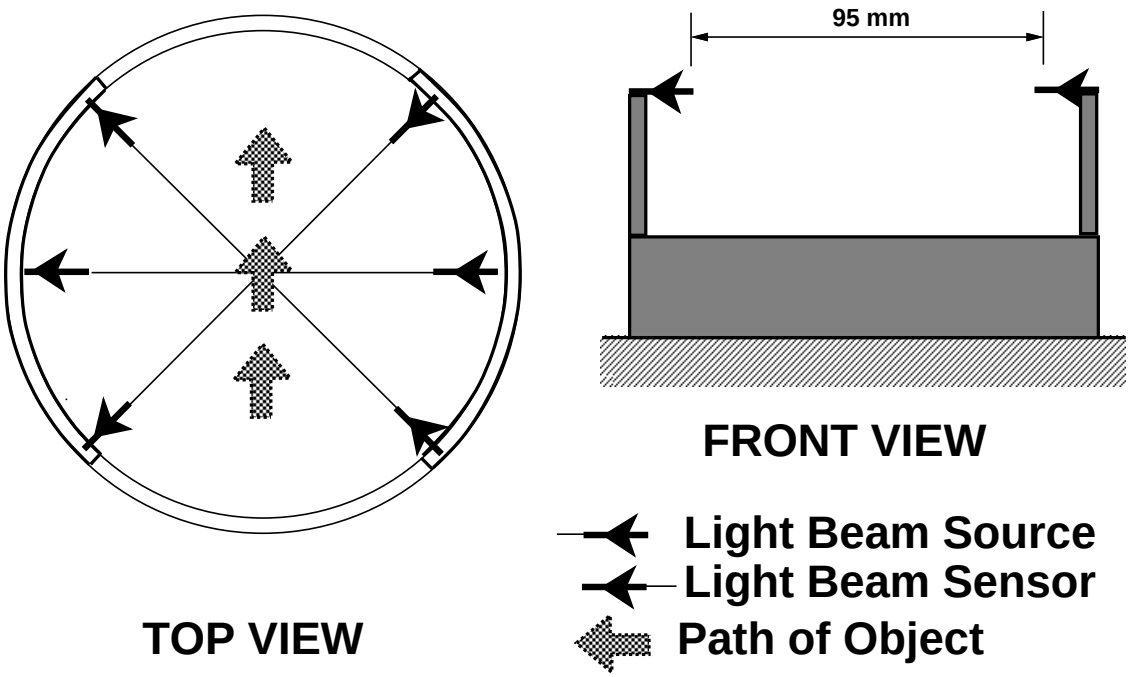

Figure 1: Optical cross beam sensor configuration

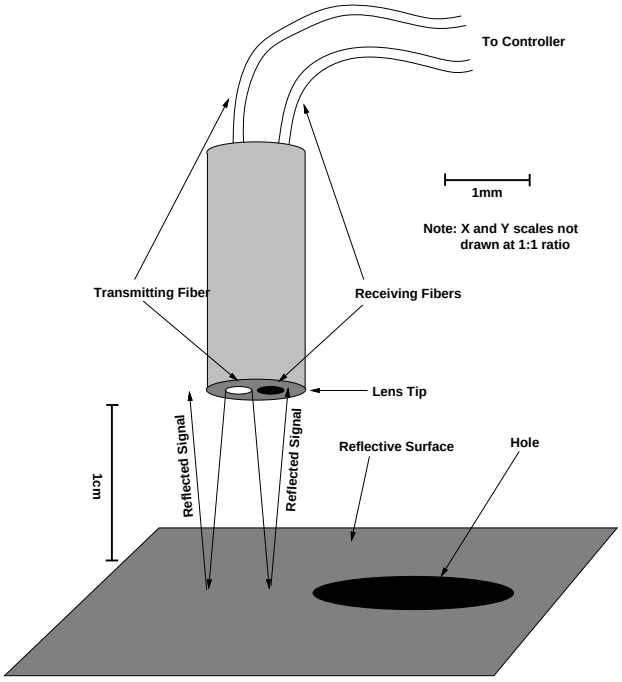

Figure 2: Optical reflective sniffer sensor configuration

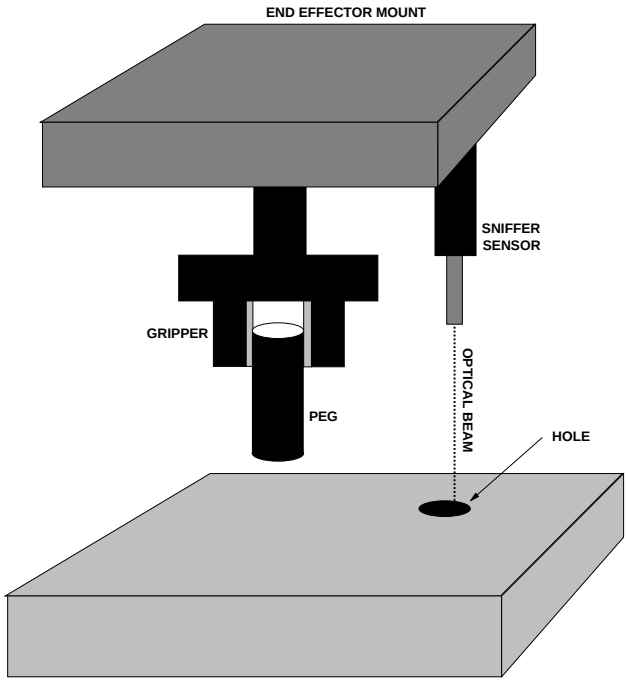

Figure 3: Diagram of hole sniffing sensor mounted on gripper end-effector

cross beam sensor configuration, localizing it. Then, using the relative position information of the hole determined by the sniffer, we perform a dead reckoning peg-in-hole insertion.

In a typical assembly workcell we would place the cross beam sensor configuration between the parts bin and assembly station. As parts were picked from the bin and moved to be assembled, they would pass through the sensor configuration and be localized instantaneously for free.

All of the tools necessary to perform the actions of our algorithm are quick, simple, and highly accurate. In the following sub-sections we present solutions to the above localization problems using these tools.

Since the pegs and sniffer sensor in our environment are cylinders, and the holes circular, we develop efficient localization strategies for these cases. Our approach can be expanded to include non-cylindrical shapes using only a few small additions. Below, we develop two algorithms for solving the localization for both holes and peg-like (cylindrical) objects.

3.2 Localize cylinders

Problem 1: Determine the position and shape of an opaque cylindrical object relative to some reference frame.

3.2.1 Single beam sensor configuration

As we pass an object through a single optical cross beam sensor perpendicular to its motion, we can record the position of the robot when the beam is broken and later reconnected. Figure 4 depicts this process. We record the y-position of the object when the beam is first broken ($T = 0$) and again at when it reconnects ($T = 1$). The diameter of the object can be determined directly from the difference in the positions of the object at the two points where the horizontal beam changed states. Assuming that the function $ypos(t)$ returns the y-position of the robot at time t we get the following.

$$P_{diameter} = \|ypos(1) - ypos(0)\| \tag{1}$$

Using the same data we can also extract the y-positional center of the object relative to the optical beam using Equation 2.

$$P_y = ypos(0) + \frac{P_{diameter}}{2} \tag{2}$$

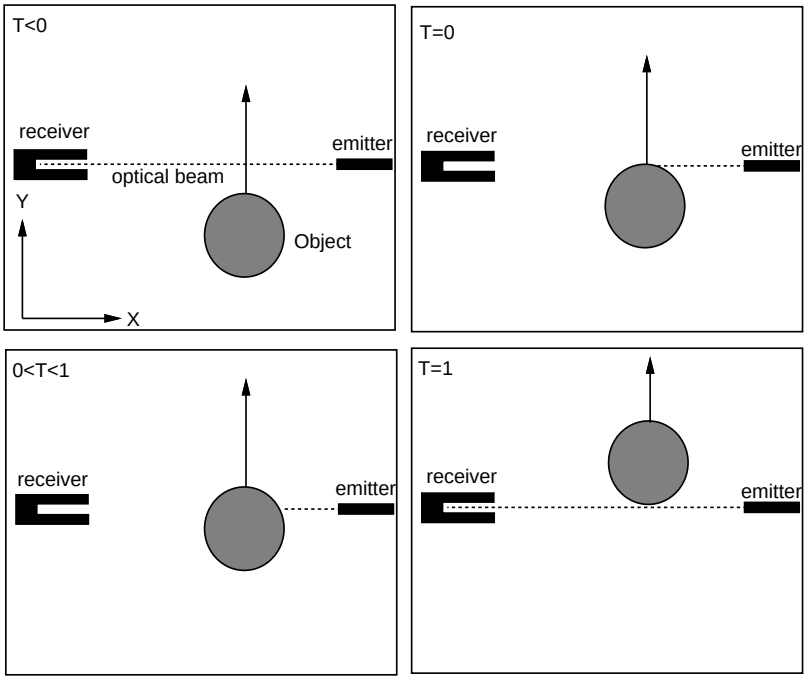

Figure 4: Using an optical sensor to determine an object's y-position and diameter. $T < 0$: the object is approaching the optical beam along the direction shown by the arrow; $T = 0$: the moment when the optical beam is first broken and position information recorded for the first time; $0 < T < 1$: the object is passing through the optical beam; $T = 1$: the optical beam reconnects and position information is recorded again.

We are left to extract the x-positional information for the center of the object. We could use a similar approach using another optical sensor mounted perpendicular the x-axis while moving the object in the x-direction. However, we want to extract all of the position information in a single pass and avoid special robot motions dedicated specifically to sensing and calibration. We solve this by using two non-collinear beams with known angles to obtain complete positional information and shape.

3.2.2 Cross beam sensor configuration

In the general case we have the situation shown in Figure 5. The beam geometry is fixed meaning that initial measurements give us θ_n and θ_p as knowns. The measurement of these beam angles is the only calibration step necessary in our approach and is performed by accurately measuring a few points along each beam with a calibration tool. This calibration is necessary only once after the sensor array has been installed in the workspace.

Using techniques similar to the single beam location strategy, we make a measurement of m as the object passes through the beam configuration. From the geometry of the system we have the following equation.

$$\begin{aligned} m &= a + b \\ &= k\tan(\theta_n) + k\tan(\theta_p) \\ &= k[\tan(\theta_n) + \tan(\theta_p)] \end{aligned}$$

Therefore, the x-positional center of the object with respect to a frame with the $x = 0$ position as shown in Figure 5 is given by the Equation 3. Note that the value for $xpos(t)$ can be taken for any t during the sensor pass since the x-position remains constant throughout the motion.

$$\begin{aligned} P_x &= xpos(t) + k \\ &= xpos(t) + \frac{m}{\tan(\theta_n) + \tan(\theta_p)} \end{aligned} \tag{3}$$

Since the entire denominator of Equation 3 is known and can be computed off-line, we need only to perform a single addition and division operation to extract the x-position information.

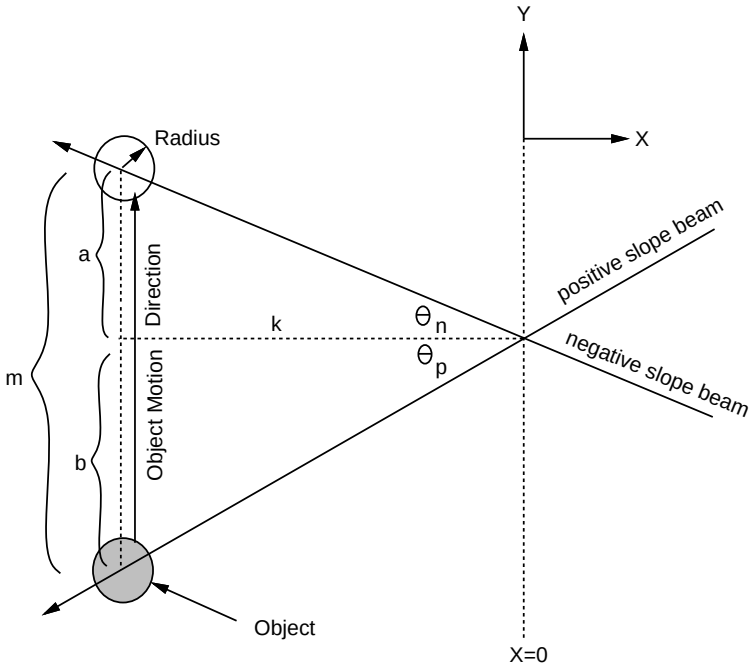

Figure 5: Optical cross beam sensor arrangement

We can also obtain y-positional information from this configuration using Equation 4 where $ypos(\alpha)$ is the y-position of the end-effector when the object is in the shaded position as shown in Figure 5.

$$\begin{aligned} P_y &= ypos(\alpha) + b \\ &= ypos(\alpha) + k\tan(\theta_p) \\ &= ypos(\alpha) + \frac{m\tan(\theta_p)}{\tan(\theta_n) + \tan(\theta_p)} \end{aligned} \tag{4}$$

An important results from this approach is that none of the optical beams are required to be aligned coincident to an axis of the robot. We are now ready to describe the algorithm for localizing a cylinder.

Algorithm 1 (Localize Cylinder) *Assume that we are given a set of optical beams arranged as depicted in Figure 1 and a method to read position information for the robot end-effector as we move the object to be localized through such an arrangement.*

1. *If the object is not mounted on the robot end-effector, grasp the part.*
2. *Pass the portion of the object to be localize through the cross beam sensor arrangement. For insertion tasks the tip of the object is localized.*
3. *Record end-effector position information when any beam changes state along with the current beam state.*
4. *Use Equation 1, 3, and 4 to calculate $P_{diameter}$, P_x, and P_y for the object.*

3.2.3 Three beam sensor

For our implementation we use three cross beam sensors. The middle beam is used to determine the object's diameter and y-position directly and the other diagonal beams for the x-position.

3.3 Hole sniff

Starting with a rough initial guess of the location of a hole, we will use an optical sniffer sensor to localize the hole to within 25 microns. As in the previous section, we will first look at the geometry of the the hole sniffing problem and describe an efficient algorithm for solving it.

Problem: Determine the location and shape of a cylindrical hole in the workcell.

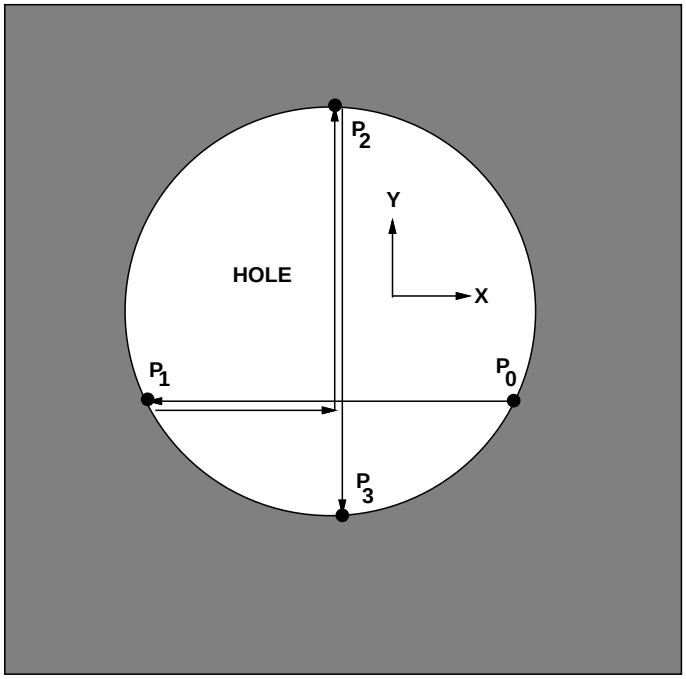

Figure 6: Strategy for collection of hole boundary points.

Our approach centers around recording the location of four points on the edge of the hole using a reflective optical sensor. We must be able to initially locate some point on the edge of the hole. If our initial guess does not place us in the hole, we perform a spiral or grid based search strategy until an edge of the hole is detected. From this first point we use the simple movement strategy depicted in Figure 6 to collect three other edge points which we use to calculate the diameter and center of the hole.

$$H_{diameter} = \|p_2 - p_3\| \tag{5}$$

$$\begin{pmatrix} H_x \\ H_y \end{pmatrix} = p_3 + \begin{pmatrix} 0 \\ \frac{H_{diameter}}{2} \end{pmatrix} \tag{6}$$

Although, three points are sufficient to localize the hole, we collect an additional fourth point to improve the accuracy of the system and simplify the calculations. The cost of collecting this extra point is insignificant since the required motion covers at most a distance equal to the diameter of the hole. As an example for a one centimeter diameter hole, the extra motion can be completed by a typical robot in under 300 ms.

Earlier we claimed that an advantage of our system was that it did not involve costly motions that were specific to sensing and calibration. However, hole localization requires several specialized motions. These motions do not violate our claim since their cost is negligible. For a hole with diameter d, we will move through a distance of at most $3d$. Typical high precision assembly tasks contain holes with diameters of a centimeter or less. This means

that the entire hole sniffing operation can be completed on typical robots in under a second or two.

Algorithm 2 (Localize Hole) *Assume that we are given a reflective optical sensor as depicted in Figure 2 and a method to read end-effector position information for the robot as we move this sensor over a hole.*

1. *Move the robot with the sensor mounted on its end-effector to the assumed hole location.*
2. *If the sensor is located over the hole goto step 4.*
3. *Otherwise the sensor does not point into the hole (i.e. the initial guess for the location of the hole is worse than expected). Therefore, perform a grid or spiral search with the reflective sensor to locate an edge of the hole.*
4. *Perform the robot motions as shown in Figure 6 and record the position of the end-effector at points P_0, P_1, P_2, and P_3.*
5. *Use Equation 5 and Equation 6 to compute $H_{diameter}$, H_x, and H_y for the object.*

4 Experimental Setup

4.1 Sensor hardware

Several optical-fiber photo-electric sensors were used for all of the sensing in this project. The basic sensor consists of a simple 23 gram controller that generates the optical light beams and outputs a digital detection signal with a response time of 50μs. Attached to this controller is a fiber-optic cable with one of several different lens types at its end. These sensors can detect objects with diameters as small as 15–30 microns.

Our implementation uses both an optical cross beam configuration with three sensors as shown in Figure 1 and an optical reflective sniffer sensor as shown in Figure 2. The digital outputs from the various optical sensors are connected to a VME bus by a digital I/O board. Once on the VME bus, the robot controller can then read the sensor values.

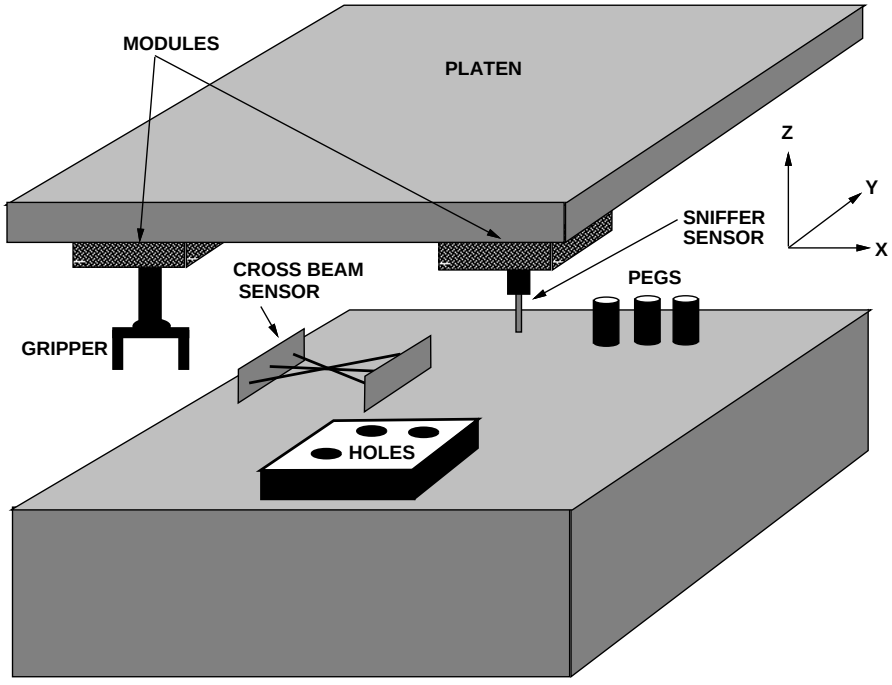

Figure 7: The RobotWorld system

4.2 Robot hardware and control

We implemented our peg-in-hole system at U.C. Berkeley on the RobotWorld robot[14] depicted in Figure 7. This robot consists of several Sawyer motor modules magnetically attached to an 0.8×1.3 meter rectangular steel platen and separated from the platen by an air bearing, thus allowing planar motion with low friction. This Sawyer drive system provides xy-planar positional accuracy on the order of 25–50 microns. Standard DC servo motors attached to the Sawyer drive base of each module provide motion in the z and θ (about z) directions. Although there are four degrees of freedom, we are only concerned with the motion directions using the xy-planar stepper and the z DC motors. These robot systems, originally developed by Yaskawa and well suited for vertical assembly tasks, are in used throughout industry.

The basic RobotWorld configuration for our implementation consists of two independent modules. A three-fingered gripper is attached to one of the modules for the pick and place operations while another module has an optical sniffer sensor mounted on it for hole sniffing. In this system, compliance, a highly desired property of RCC peg-in-hole systems, is traded for high positional accuracy. All robot control and sensing are performed by Talisker[15], a multi-threaded, real-time robot control system developed by Ed Nicolson *et al.* at U.C. Berkeley running on a 68040 microprocessor on a VME backplane.

5 Results

We performed repeated insertion tests of our peg-in-hole strategy using pegs and holes of various diameters both with and without chamfers. It is also important to note that both the peg and hole locations were displaced by arbitrary distances up to approximately one centimeter over successive executions to show the dynamic self-calibration feature. The results from these tests are shown in Table 1. We conclude that our system is highly robust to varying position errors over multiple executions.

Insertion Tolerance	Chamfered Hole or Peg?	Attempted Insertions	Successful Insertions	Success Rate
25μm	No	100	99	99%
26μm	Yes	100	99	99%
26μm	No	100	100	100%

Table 1: Insertion test results

In addition, our peg-in-hole system is fast. A typical insertion from start to finish took 13 seconds of which 2 seconds were devoted to sniffing sensing, 0 seconds for object localization, and the remaining 11 seconds for pick, place, and transport operations. Subsequent insertions are performed even quicker since we do not require re-localization of the sniffer sensor before those holes are localized. We also exploited the parallelism of the system by localizing holes in parallel with localizing pegs.

6 Conclusion

In this paper we have shown the use of RISC sensors for performing high accuracy peg-in-hole insertions. More importantly our strategy is dynamically self-calibrating, making it robust to position errors in the pegs, holes, and robot end-effector over successive assembly operations. We have eliminated the need for any costly off-line calibrations, instead localizing objects to within 25 microns during normal assembly operations such as part transport. Finally, our system is cost-effective, using inexpensive sensors already found in manufacturing applications.

7 Acknowledgments

The authors would like to thank Ed Nicolson for designing the controller for RobotWorld and for all of his assistance with the real-time interface to the RobotWorld environment, Aaron Wallack for calibrating and upgrading the cross-beam, optical sensor, and Dr. Richard Murray for constructing the original prototype interface to RobotWorld. In addition we would like to thank all of the people mentioned above for their numerous thoughtful criticisms, suggestions, and comments.

Financial support was proved in part by National Science Foundation Presidential Young Investigator Award #IRI-8958577 and National Science Foundation Grand #IRI-9114446.

References

[1] James L. Nevins and Daniel E. Whitney. Computer-controlled assembly. *Scientific America*, 238(2):62–74, February 1978.

[2] John F. Canny and Kenneth Y. Goldberg. A "RISC" paradigm for industrial robotics. Technical report, University of California, Berkeley, February 1993. ESRC 93–4/RAMP 93–2.

[3] T. Lozano-Pérez, M.T. Mason, and R.H. Taylor. Automatic synthesis of fine-motion strategies for robots. *International Journal of Robotics Research*, pages 3–24, 1984.

[4] Michael Erdmann. Using backprojections for fine motion planning with uncertainty. *International Journal of Robotics Research*, pages 19–45, 1986.

[5] B. Donald. A geometric approach to error detection and recovery for robot motion planning with uncertainty. *Artificial Intelligence*, 37:223–271, 1988.

[6] J. Canny. On computability of fine motion plans. In *IEEE Conference on Robotics and Automation*, pages 177–183, 1989.

[7] J. Canny and J. Reif. New lower bound techniques for robot motion planning problems. In *IEEE Conference on Foundations of Computer Science*, pages 39–48, 1987.

[8] D.E. Whitney. Quasi-static assembly of compliantly supported rigid parts. *ASME Journal of Dynamic Systems Measurement and Control*, pages 65–77, 1982.

[9] Michael A. Peshkin. Programmed compliance for error corrective assembly. In *IEEE Transations on Robotics and Automation*, pages 473–482, 1990.

[10] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, RA-3(1):43–53, February 1987.

[11] Louis J. Everett and Thomas W. Ives. A sensor for measurments in the calibration of production robots. In *IEEE International Conference on Robotics and Automation*, pages 174–179, 1993.

[12] J.P. Prenninger, M. Vincze, and H. Gander. Contactless position and orientation measurments of robot end-effectors. In *IEEE International Conference on Robotics and Automation*, pages 180–185, 1993.

[13] Aaron Wallack, John Canny, and Dinesh Manocha. Object localization using crossbeam sensing. In *IEEE International Conference on Robotics and Automation*, pages 692–699, 1993.

[14] "Description of a Robot Workspace Based on a Linear Stepper Motor". *AT&T Technical Journal*, 67(2):6–11, 1967.

[15] Ed Nicolson. Real-time robotic workcell server specification. Technical report, University of California, Berkeley, May 1992.