

Integrated Vision and Touch Sensing for CMMs

Billibon Yoshimi, Tsai-Hong Hong, Martin Herman, Marilyn Nashman, and William G. Rippey
National Institute of Standards and Technology
Bldg. 220/B124
Gaithersburg, MD 20899
{yoshimi, hongt, herman, nashman, rippey}@cme.nist.gov
<http://isd.cme.nist.gov/brochure/NGIS.html>

ABSTRACT

This paper describes a calibrated and an uncalibrated method for integrating a vision system with touch sensors and a Coordinate Measuring Machine (CMM). Traditionally, vision systems are used to measure machined parts and to localize manufacturing defects based on appearances. Our system relies on a “coarse-fine” approach to integrating vision with CMMs. Vision guides the CMM to the feature to be measured and the CMM performs the actual measurement. The resulting system is capable of quickly inspecting large regions with high resolution. We will examine what impact the internet may have on CMM technology and the ways in which vision can make CMM technology accessible over it. We will also present results demonstrating the power and flexibility of new, uncalibrated vision techniques.

DISCLAIMER

Certain commercial equipment, instruments, and materials are identified in this paper in order to adequately specify the experimental procedure. Such identification does not imply recommendation or endorsement by NIST, nor does it imply that the equipment, instruments or materials identified are necessarily best for the purpose.

1. Introduction

In the past, Coordinate Measuring Machines (CMMs) required extensive human intervention to take measurements. The human operator controlled all of the CMM's movements and indicated when data points should be logged. When new parts needed to be measured, the operator analyzed the measurements to be taken and then proposed a measurement plan, which consisted of a list describing how and in what order the measurements were to be executed. The chief bottleneck in such CMM systems was the human operator. Since the operator had complete control of the CMM at both the highest and lowest level, the measurement process depended highly on the skills of the CMM operator.

This paper describes how vision-based CMM control algorithms may be used to perform some CMM programming tasks. Some of the paper highlights include the following:

- A proposed decomposition of certain measurement tasks to determine which sub-tasks are suitable for human control and which are suitable for CMM control.

Instead of relying wholly on one or the other, the operator and machine work together in a coordinated manner to perform measurements.

- A description of several improvements made to the general operator-CMM interface.

This includes the development of a system where a human operator plans, requests, and monitors a series of measurement tasks for a CMM system from a remote site.

- A description of how computer vision may be used by both the CMM and human operator to observe and control a measurement task.

We have developed vision algorithms which enable a CMM to sense how it interacts with objects in the 3D world. Using these algorithms, the CMM is capable of "learning" the mapping between image space and the 3D world. We have experimented with two forms of algorithms: calibrated and uncalibrated.

- A description of how vision may be used to perform "virtual" fixturing. Mechanical fixtures are commonly used to constrain the positions of parts before they are measured.

By constraining the part, the measuring system removes much of the positional ambiguity before measuring the part. Vision is capable of performing a similar task. Similar to mechanical constraints, vision can use optical constraints to limit a part's positional ambiguity.

- Finally, a brief discussion of the benefits of hierarchical control.

Our system controller implements coarse-fine control. Vision is used to direct the movement of the CMM with millimeter precision, while touch probes are used to recover sub-micron displacement information. Combined together, vision and touch give the system an extremely fast way to recover dense, high-precision measurement data.

2. Background Information

The Next Generation Inspection System (NGIS)[10] is a multi-year effort to develop a CMM test-bed for testing and developing advanced sensing and control technology. NGIS draws on research from many diverse fields including robot control, image processing, tracking, and metrology.

Our work is a direct consequence of our previous research experience in integrating imaging systems with robot technology. A relatively complete compendium of the different research approaches examined in field of integrating vision and robot systems can be found in [6]. One of the works cited is Allen et al. [2] where we demonstrated that real-time control of tracking and grasping is possible using stereo cameras. This work shows that calibrated stereo vision, coupled with a suitable prediction and noise filtering module, is capable of controlling and coordinating the movements of a robot and gripper system. In Yoshimi and Allen [14], we have shown that the peg-in-hole insertion task may be performed with an uncalibrated camera. This work demonstrated that carefully chosen visual invariants may be used to drive a robotic process efficiently. An uncalibrated camera was used to direct the movements of a Puma 560 robot. The robot's task was to insert a thin peg into a 2 mm hole. The system performed the peg-in-hole insertion repeatably. In Yoshimi and Allen [13], we have shown that robot grasping and manipulation is possible with weakly calibrated cameras. By using image Jacobian-based techniques, a robot system was capable of compensating for image calibration errors. In both systems, vision was used to close an otherwise open loop manipulation problem.

More recently, Nashman et al. [10] described how vision data could be used to control the acquisition of scanning touch data. Their system measured the surface profile of a piston head using visual feedback to control the velocity of the probe tip. Vision served two purposes in their system. First, vision was used to recover the head of a piston. Second, it was used to control the velocity of the probe as it moved across the surface of the piston head. As the tip of the probe approached the lip of the piston head, the probe's velocity decreased. Our current system extends upon this work.

The NGIS controller is the NIST Real-time Control System (RCS) [1]. Several projects at NIST and industry have adopted RCS for building robot controllers. RCS has been used on welding, deburring, and milling machines. NGIS uses the RCS libraries to control the CMM, probe, and vision systems. RCS is a multi-level, hierarchical control system. At the lowest levels, the system models the organization of a single machine. These levels include *servo*, *primitive (prim)*, *elementary-move (e-move)* and *task*. At higher levels, RCS addresses the needs of workstations and factory control. Each level operates on a different time and space scale. At the lowest levels, joints and encoders are controlled at the *servo* level with milli-second accuracy. At the highest levels, factory schedules are controlled with day accuracy. RCS is conducive to rapid prototyping, code-reuse and concurrent software development. The NGIS system implements the 4 lowest levels of RCS control: *servo*, *prim*, *e-move* and *task*.

Before going into depth about our system, we will examine two issues related to the integration of vision and touch. First, we will examine how two paradigms of visual control, calibrated and uncalibrated, apply to CMM control. A calibrated vision system provides a direct mapping between image space and the 3D world. An uncalibrated system does not depend on having such a mapping before operating. An uncalibrated vision system forces us to develop techniques which do not depend on the image space-to-3D world mapping. Although at first glance uncalibrated vision systems look impractical, they are, in fact, cheap to construct and resilient to minor perturbations. We will describe techniques we have developed to perform visual control in both environments.

Second, we will describe some of the challenges faced in developing a distributed interface to all of the individual parts of the system. In particular, we will describe how the user interface was derived to be “internet-friendly” and why developing a CMM measurement system with the internet in mind is a good idea.

3. Vision paradigms - Calibrated vs. uncalibrated vision

Two distinct vision paradigms exist in the visual control domain. The first paradigm, calibrated vision, uses calibrated camera systems and performs all control operations in 3D world coordinates. The second paradigm, uncalibrated vision, uses image Jacobian-based techniques. These two paradigms will be described in more detail in the following subsections.

Calibrated vision

Calibration gives the position and orientation of a set of cameras in the 3D world and the optical characterizations of those cameras. Once the camera calibration is known, the position of any object visible in those cameras may be back projected into the 3D world to determine its 3D position and orientation.

Manipulation under a calibrated camera system is typically performed in the 3D world coordinates. Part locations and the location of the CMM may be specified in 3D. Manipulation is relatively straightforward. Distances to objects are determined using straightforward Cartesian calculations. Grasping and manipulation strategies may also be performed in Cartesian space. Basically, under calibrated vision, all manipulation problems requiring vision are transformed into 3D Cartesian-based manipulation problems.

However, the camera calibration problem is known to be very difficult. Many researchers have spent their careers examining this problem. Karara’s handbook [8] is considered to be one of the most complete handbooks on camera calibration. Still, every year, more and more researchers spend time, money, and effort to develop new calibration techniques and/or to implement and understand old techniques.

Calibration techniques usually fall in two general classes: simple, fast calibrations and complex, slow calibrations. Simple, fast calibrations describe the image space to 3D-world mapping with a small number of variables usually using a small number of initial calibration data points. These techniques include many variations of the two plane technique such as the one by Martins et al.[9] and techniques which rely on imaging 8 corners of a viewing volume such as that by Hollinghurst and Cipolla[7]. Generally, these techniques are simple to implement and may be performed relatively quickly. All of these methods produce rough calibration results. Using these calibration techniques usually results in large errors when projecting image space points into the 3D world.

Complex calibration techniques, like many of the algorithms covered in [8], recover the calibration of a camera system with a large number of variables representing many different optical distortions. Unlike the previous set of calibration techniques, these are highly polished numerical algorithms which require the use of nonlinear optimizers to achieve their accuracy and precision. These techniques are often slow, hard to understand, highly susceptible to noise and, at times, do not give better results than the simple, fast techniques. Faster versions of these complex calibration techniques, such as Tsai [12] and Castaño [3], were designed to be used in machine vision applications. These faster versions use complex

models and nonlinear optimization to achieve precise calibrations. These methods also share many of the problems with complex calibration including being difficult to explain, hard to understand, and sensitive to noise.

Calibrating a camera system has many advantages. In an environment where a camera does not move and its optics do not change, a one-time, well-performed calibration can be used for the entire life-time of the system. In this case, recovering a highly accurate calibration is worth the initial start up cost. Calibration also simplifies the visual control problem since all manipulations are performed in 3D world coordinates.

Calibration, however, does not solve all problems in visual control. Because all cameras have a finite number of sensing elements, choosing the right lens for a job is very critical. The camera should see enough of the world with high enough resolution to perform the desired operation. Due to the trade-off between resolution and accuracy, most calibrations are typically accurate in only a small part of the robot's workspace. If a larger area is to be imaged, each pixel must cover a larger percentage of the world. Since the spatial resolution of the image sensor remains constant, the overall 3D spatial resolution decreases. If a smaller area is imaged, parts of the workspace remain unimaged and therefore, uncalibrated. Another problem comes from introducing an artifact into the robot workspace, often one of the first steps in most calibration algorithms. If the calibration process occurs once, the time to install and remove the artifact may be insignificant. But, if daily recalibration is necessary the time spent introducing and removing the artifact may become significant. As mentioned earlier, calibration algorithms are still a subject of intense research. Many of the sophisticated calibration methods require careful estimation of highly sensitive tuning parameters. Since nonlinear optimization is notoriously sensitive, many of these algorithms are difficult to implement and use.

Calibration, a time consuming process, needs to be performed any time the camera's position changes. If small perturbations are made to the camera system or to the calibration matrices, the calibration may fail catastrophically with no possibility of recovery. Calibration methods may make certain assumptions about lens distortions which are violated for extremely high distortion lenses like many wide field-of-view lenses. In many instances, these are the same types of lenses which are most useful in a machine vision system.

Although calibrated vision systems have much to offer in a vision-guided robot system, the problems associated with their implementation make them difficult to use in certain CMM/vision configurations. These problems encouraged us to explore the new uncalibrated techniques described in the next section.

Uncalibrated vision or active vision

Uncalibrated vision and active vision are relatively new areas of study in the robotics and computer vision fields. Generally, these terms imply that the vision system directs the information gathering process. Examples of these techniques include visual servoing, visual object tracking, and gaze or vergence control.

One advantage of these techniques is that they may not require explicit calibration. Since the vision system controls the acquisition of data, it is possible for the system to "learn" the image space to 3D world calibration. One method for learning this calibration is through forming an image Jacobian, a mathematical construct relating differential changes in image space to differential changes in 3D space. An image Jacobian is similar to a calibration matrix except that its value changes at different locations in

space and is usually valid only for a single position in image space and 3D space.

Uncalibrated vision and, in particular, image Jacobian-based control have many advantages. First, the amount of time necessary to recover an image Jacobian for a given position in space is nominal. The initial start-up sequence for the recovery of an image Jacobian is to perturb the robot's position along two axes. Many precision camera calibration procedures require the introduction of artifacts in the robot's workspace. Introducing, imaging, and removing these artifacts requires time. Other calibration methods requiring the robot to move to various positions in the environment also take time. By calculating the image Jacobian only when necessary, we can save a great deal of system time and effort. The system can immediately use Jacobians derived from this process.

Image Jacobians measure the relationship between image space and 3D space directly. Since the vision system observes how the robot moves under different commands, it knows exactly how these commands affect the pose of the robot in the 3D world. If the robot moves to a new position in space, the image Jacobian can be quickly established for that new position. If the camera position changes, this, too, may be learned. The key to developing active vision algorithms which use image Jacobians is to rely on observed relative movements. These techniques require understanding how movements in the 3D world affect the robot configurations in image space.

The process of updating an image Jacobian is similar to that of updating a calibration matrix. Instead of performing a complete and possibly flawed calibration up front, image Jacobian-based techniques are updated as they perform the desired robotic task. These methods are amenable to many machine learning approaches.

As stated earlier, estimates of the image Jacobian appear to approximate the real image Jacobians very well. Both [13] and [14] demonstrate that the discrete form of the image Jacobian is sufficient for performing many robotics tasks. Yoshimi and Allen[14] demonstrate that even a single estimate of an image Jacobian is sufficient if the task is well constrained. Yoshimi and Allen[13] also show that image Jacobian-based techniques can be used for point-to-point robot control. Depending on what control law is chosen and the value of the gains, it is possible to construct linear motion paths using image Jacobian techniques.

Finally, using any differential technique like the image Jacobian allows the system to cancel out many fixed calibration errors. In [13], the system could compensate for a base-line shift of 400mm and still accomplish a peg-in-hole insertion task. Any method which requires the robot to move to a calibrated position would have had difficulty performing this task if the camera calibration were not updated.

Image Jacobians, however, are not without problems. One of the biggest problems is determining when the Jacobians break down. It is possible to check the accuracy of the Jacobian by observing the movements of the robot system in image space. If the signs of any of the terms in the Jacobian change or the Jacobian becomes singular, the system can be ordered to recompute the Jacobian. Yoshimi and Allen[14] have shown that large areas of a robot's workspace have Jacobians which are quite similar. Coupled with error minimizing techniques, an initial estimate of a Jacobian may be used over a large area with little degradation.

In many situations, it may not be possible to calculate an image Jacobian. If the basis set is not chosen carefully, an image Jacobian may be singular. This situation is not uncommon. Any Jacobian computed where one of the axes is a vector which lies along a line passing through the optical center of the camera is an example. Selection of good basis sets for different robot/camera configurations is an open

research topic.

Also, using the initial image Jacobian, the robot may slowly converge to the desired control point. This depends on how accurately the image Jacobian reflects the true image Jacobian at all points along the robot's path.

Still for many users, image Jacobian-like techniques are relatively cheap, straightforward starting points for the implementation of visual control systems.

4. Internet-based delivery of measurement services

Most CMMs require resources beyond the reach of many small companies. The cost of buying, installing, and maintaining a CMM is significant. We are currently working on designing technologies which will enable small companies to operate off-site CMMs safely, efficiently, and effectively. Two major problems with remote CMM operation exist. First, not everyone has CMM expertise. Mixing inexperienced users with expensive machinery is a guaranteed recipe for disaster. Second, the technology for connecting potential users to CMMs is, at best, nominal. Currently, the most pervasive form of connectivity is the internet.

Current internet technology is not capable, and never will be capable of delivering the real-time performance required for normal operator interfaces. The internet is at best non-deterministic and at worst unreliable. We have developed an internet-based solution which overcomes some of these problems. If the only feedback a remote CMM operator gets is digitized video from a camera and simple message packets, then packet delay and loss can cause the system to behave erratically. In the worst case, lost or late control commands may cause extreme robot/probe damage. Whatever interface is implemented should include the most basic of all CMM operations: the emergency stop (e-stop). The e-stop button is probably the most important button on the CMM control panel, since it is often paramount to worker safety. Operating an expensive CMM over the internet requires at least the same level of CMM responsiveness.

The key to the success of internet-based operations on the NGIS system is the development of internet-safe CMM operations. An internet-safe operation is one which was specifically designed with the problems of the internet in mind (non-deterministic operation and no guarantee of delivery). For example, if an operation is moving the probe until it contacts the part, normally a human operator would move the probe using a joystick until the probe came close to the part. The operator would then reduce the speed of the CMM and move the probe closer to the part. An experienced CMM operator can quickly judge which velocity profiles to use when closing to the surface of a part.

Our system achieves this level of operation by separating the real-time component from the user interface. The CMM only executes commands which are finite and have specific terminating conditions independent of human reaction times to succeed. Similarly, high-level image processing is performed in real-time on a dedicated image processor. Updates are sent to the user interface as quickly as possible, but the system never assumes that the users have certain knowledge before the users perform some action.

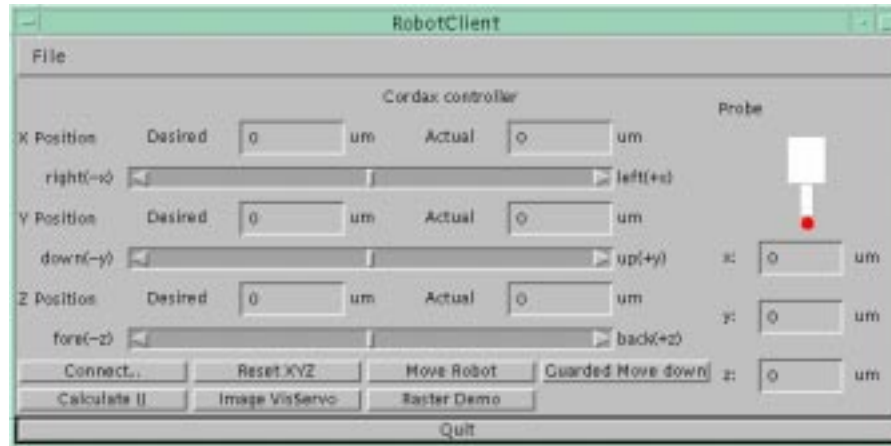


Figure 1. CMM and probe control panel.

Our system performs all probe tracking and robot control in real-time. E-stop commands are automatically generated by the system based on probe and vision data. All other functionality runs in non-real-time mode since the inherent uncertainties of the internet make it impossible to guarantee bandwidth and connectivity. All operations must have some internal mechanism to keep the system operational. We have isolated the user interface (UI) components and have implemented robot manipulation algorithms which do not rely upon “hard” real-time user interaction. Using the touch-based guarded move operation isolates a potentially hazardous real-time condition. The system terminates a movement operation when the probe moves into contact with an object. In this scenario, the real-time aspect of halting the robot does not depend on user intervention. While the user may initiate the task, task monitoring and completion checking are under complete autonomous control.

Other examples of this technique include the visual guarded move - moving the CMM to a desired position in image space, 1D compliance against a surface - moving the CMM to a desired position while maintaining probe contact with a surface to be measured, and 2D compliance against a surface - forcing the CMM to maintain contact with a surface while maintaining a trajectory parallel to the surface of contact. Notice that in all of these instances, the robot is autonomously controlled in real-time at the lowest level of the hierarchy. At the task level, higher level control is less time critical and non-real time. This programming model reinforces the separation of real-time and non-real-time components.

Implementation

The internet interface consists of two parts implemented in Java[4]. These two windows are the probe/CMM control window and the vision control window. Each interface is independent. Exercising the vision system without invoking the CMM and vice versa is possible. Visual servoing is possible, though, only when both interfaces are active.

Figure 1 shows the CMM/probe window (or panel) is comprised of two parts. The CMM control interface is on the left-side of the window. When the CMM/probe is invoked, the user interface connects to four RCS communication channels which we call the Neutral Manufacturing Language (NML). Two command channels exist for sending commands to the CMM and the probe. The other two channels report the status information of the CMM and the probe. Robot positional status information is displayed textually and through the slider bar. The user can modify the position of the robot by either sliding the



Figure 2. Vision control panel.

slider bar or typing in a position. The MOVE ROBOT command is only sent to the robot when the MOVE ROBOT button is pressed. The MOVE ROBOT command actually violates some of the principles discussed earlier. Why? There is no stop action. The CMM will not terminate a MOVE ROBOT command unless the goal position is reached. The GUARDED MOVE DOWN enhances the MOVE ROBOT commands functionality by adding a probe-triggered guard function.

The probe subwindow (on the right-side of the CMM/probe window) is currently a status-only window which displays the probe deflections. The window contents cannot be altered. In the future, we will add probe changing capabilities when a probe changer becomes available.

The vision interface window (Figure 2) allows the user to see what the vision system “sees.” In Figure 2, the top panel contains buttons which allow the user to connect to various vision processing computers to provide visual feedback for control. In our laboratory, we have at least three such computers. The state of the user interface also determines what effect button clicks have on the image display in the bottom panel. For instance, in the normal state, a button click has no effect. Dragging the mouse with the left mouse button depressed causes a box to be drawn on the image, perhaps to highlight some event. If the interface is in `add_tracker` mode, clicking on the interface sends a message to the vision system to allocate a tracker at the clicked-on image position. Currently, the system supports the creation of two different types of image-based trackers: threshold-based and correlation-based. Correlation-based tracking is discussed in [10]. Threshold-based tracking examines pixels in a 10x10 neighborhood surrounding the specific point. It finds the centroid of all pixels whose value satisfies some threshold criterion. The value of the centroid is used as the position of the tracked point. The upper limit on the number of regions tracked depends highly on the vision system and processing platform.

The vision interface also includes support for entities (albeit at a very primitive level currently). Figure 3 shows the Entity Manager control panel used to manage vision scan paths. The probe will scan the part surface while following these paths. The scan paths are point-based entities. The system is capable of recognizing three separate types of scan paths: the random “point-and-click” scan path, the inner surface scan path (the inside of a pocket) and the outer surface scan path (the outside of a boss). All three

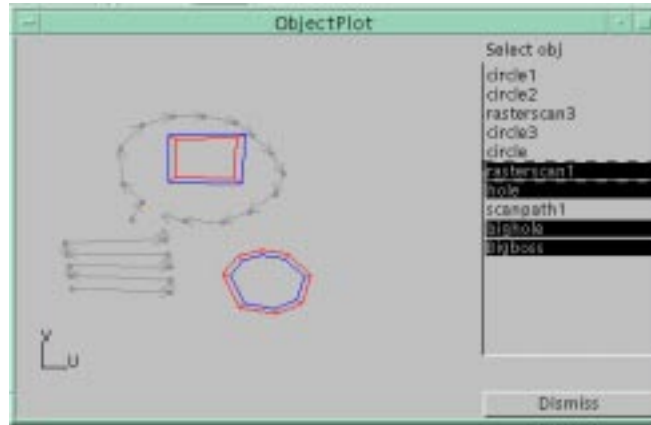


Figure 3. Entity manager control panel

types of paths are indicated in Figure 3. The circle on the lower right is a pocket to be scanned. The rectangle located in the circular path is a boss. The remaining two paths are arbitrary scan paths created by the user with the point and click interface. In the next version of this software, this interface will be supplemented with CAD model constructs.

Performance

Since Java is an interpreted language, the resulting software interfaces run approximately one order of magnitude slower than compiled software. This performance cost was deemed acceptable for the greater potential of easily distributable software. For the probe/CMM interface, the relatively low bandwidth necessary for moving the CMM and observing the probe deflections does not seem to tax network bandwidth. On the other hand, the vision interface requires tremendous bandwidth for video images. Using uncompressed 320x240 8-bit grey scale images, we were able to get ~5Hz video updates to a Java user interface. The video server, itself, was capable of processing video and tracking multiple targets at 30Hz. The video server could also relay these tracking results to the probe/CMM system at 30 Hz with minimal delay.

Both interfaces benefited from using RCS Neutral Manufacturing Language (NML)[11] as the communication library. First, having a standard communication interface makes it possible to swap out the user interface and insert a higher level control module. This is one which possibly performs many of the tasks the human operator performs without rewriting the CMM controller code. Moreover, since NML supports multiple writers to a command channel, it is possible for both an expert and a novice to connect to the same CMM and share control of that CMM during some measurement operation. Such operations are normally not part of a standard CMM communications package. Second, a standard set of CMM messages - commands sent from other processes to the CMM process - allows us to concentrate on the task of programming a CMM over learning a new language. The same user interface and/or higher level control process can access our current CMM or a new CMM speaking the same language as long as they both implement the same command interpreter. These and many other aspects of RCS are discussed in [11].

5. System Description

Figure 4 is a schematic illustration of the NGIS system. The CMM controlled by the system is a

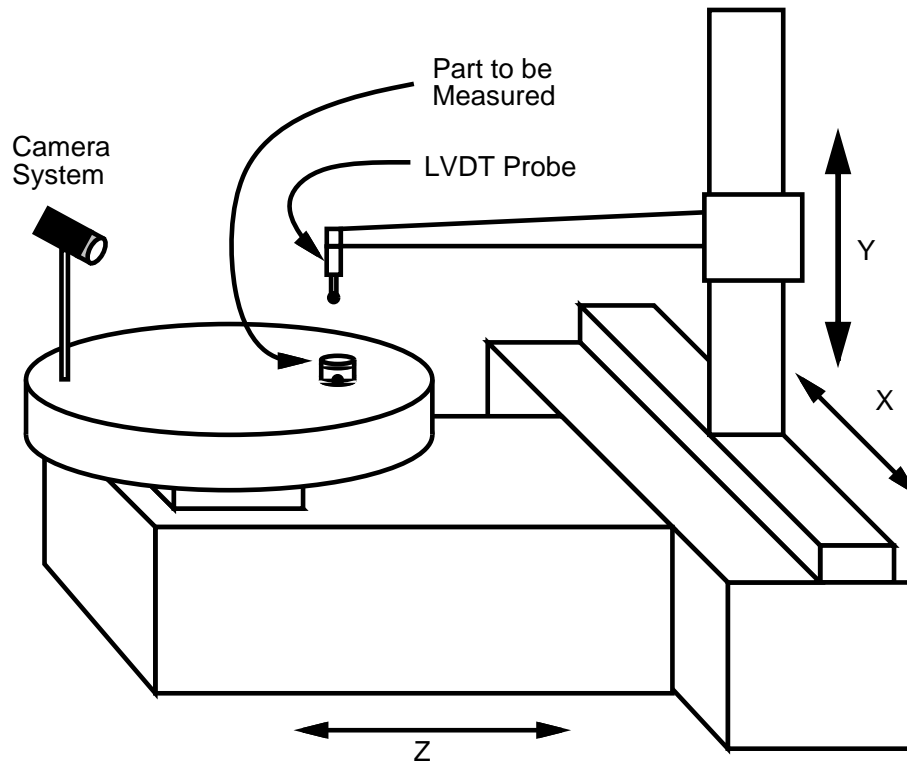


Figure 4. Illustration depicting CMM, probe and camera setup.

four degree-of-freedom Cordax. The rotational degree of freedom associated with the table has been disabled for our experiments. An Automated Precision, Incorporated (API) 2D pan-tilt wrist has been attached to the end effector of the CMM. Future experiments will include automatic normalization of a probe principle axis to the surface scanned. For all of the experiments described in this paper, the wrist was held in a fixed pose.

We have experimented with several different probes in this system. Currently, the complement of probes includes a simple 1D LVDT (linear variable differential transformer), a 3 DOF LVDT from API, a capacitive sensor from ExtrudeHone and a laser triangulation probe from Sensor Adaptive Machines, Inc. (SAMI). A kinematic base has been attached to the base of each probe to facilitate the connection of each probe to the pan-tilt wrist.

The vision system consists of an Elmo "Lipstick" CCD camera with a 12mm lens. The Elmo generates both SVideo and NTSC output. The Elmo SVideo output is fed into a Sun Video digitizer mounted in a Sun Ultrasparc workstation running Solaris 2.5. While the Sun Video digitizer is capable of digitizing 24-bit, 320x240 pixel color images at 30 frames per second (fps), the system only uses 8-bit, grey-scale, 320x240 pixel images. While many specialized vision systems exist which are capable of measuring several million pixels per frame, most commercial off-the-shelf NTSC systems digitize at either a resolution of 640x480 or 320x240 pixels. Even at 320x240 pixels, our system demonstrates the feasibility and utility of vision-based robot control.

The NGIS software is constructed on top of a heterogeneous operating system environment. Currently, the Cordax CMM is controlled under VxWorks, the probes are controlled either under VxWorks or Microsoft Windows NT, the vision system is controlled under Solaris, and the user interfaces are con-

trolled under Java Virtual Machine[4]. In most typical developmental environments, this number of operating systems would make the cost of maintaining communications libraries prohibitively expensive. The NGIS system uses RCS NML to overcome this problem. NML allows many different processes running on different computers to communicate with one another. NML has support for single read/write channels and read/write channels with multiple readers and writers. It has support for queueing messages. It has support for backplane, shared memory, and internet communication. The NML libraries that are available free from NIST are compilable under VxWorks, Solaris, Lynx OS, and the many forms of Microsoft Windows. [11]

As stated earlier, the connection between the various hardware and software components and the user is handled through Java user interfaces. Java is a new language designed with the internet in mind. Java code is not executed on the host machine; it is executed in the user's web browser. The delivery model for Java interfaces is as follows. A pointer to Java code is imbedded in an internet web page. The user loads the web page into his/her web browser via an internet connection. When the pointer corresponding to the browser is accessed, either by key press or by mouse click, the user's web browser opens a connection to the server holding the code, downloads the code and executes it in a secure "sandbox" called the Java Virtual Machine. The Java Virtual Machine was especially designed to securely handle code on the internet. For more information on Java see [4].

Figure 5 illustrates the relationship between the different coordinate systems in the NGIS system. The following parameters are used in the paper to describe various relationships within our system:

- W - the world coordinate system.

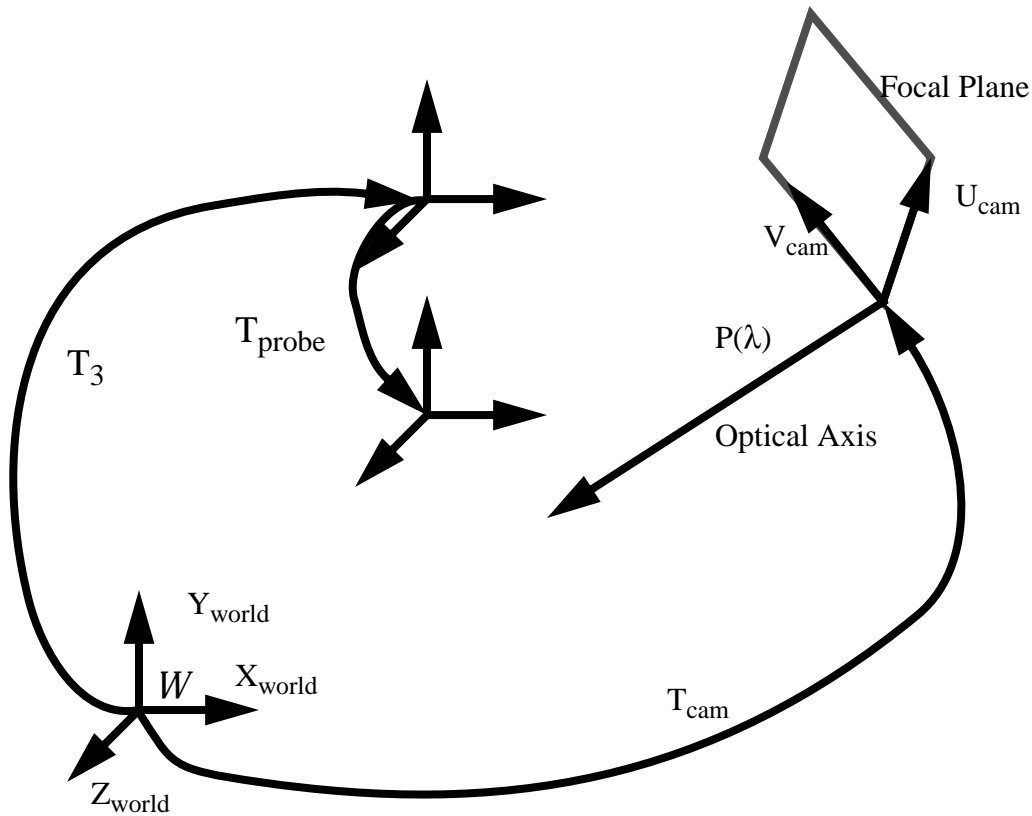


Figure 5. Relationship between the different coordinate systems.

- $(X_{\text{world}}, Y_{\text{world}}, Z_{\text{world}})$ - the axes of W .
- T_{cam} - the camera transform matrix.
- $P(\lambda)$ - the perspective effect introduced by a camera system with focal length λ .
- $(U_{\text{cam}}, V_{\text{cam}})$ - the axes of the image space coordinate system.
- T_3 - the transform from W to the end effector of the CMM minus the probe.
- T_{probe} - the transform from the end-effector of the CMM to the tip of the probe.
- TIP - the actual 3D location of the probe tip in the 3D world. $\text{TIP} = T_3 * T_{\text{probe}}$.
- TIP' - the projection of TIP into image space. $w * \text{TIP}' = P(\lambda) * T_{\text{cam}}^{-1} * \text{TIP}$.
- w - is a camera dependent scaling factor.

6. Experimental Results

We have developed two methods for vision-based CMM control. The first calibrates a camera system using a standard calibration technique and then performs movements and manipulations. The results of this calibration-based approach are presented in the next section. They illustrate the fidelity of a simple camera calibration algorithm. The second section describes an image Jacobian-based technique which requires almost no set-up time. Experimental results are presented demonstrating the efficacy of the image Jacobian in performing a CMM measurement task.

Calibrated vision results

This section describes vision-guided CMM control using a calibrated vision system. The calibration is performed using an algorithm by Castaño [3]. This algorithm is a nonlinear calibration procedure which uses linear calibration results as the initial guess for an iterative search.

We used the following procedure to acquire calibration data for the algorithm. First, the probe is mounted on the CMM, and the camera was mounted in a location where it could observe the workspace of the CMM. The visible workspace of the vision/robot/probe system is initially estimated by hand. The CMM is moved under joystick control to many different positions in the workspace to measure the extent of the camera's field of view. The probe is moved closer to the camera and then moved so the projection of the probe lands as close as possible to the top-left, top-right, bottom-left and bottom-right corners in camera space. It is not possible to reach these corners in most cases due to physical constraints. This defines a "front plane" to the workspace. We then move the CMM away from the camera and perform the same operation. This operation defines a "rear plane." This frustum-like space is then populated with three dimensional points.

Next, a red light-emitting diode (LED) is attached to the probe tip. LEDs are inexpensive and are easily tracked in images. All external lights are dimmed. The LED's position is found by first thresholding the image and then finding the centroids of the resulting pixel clusters. We assume the illumination

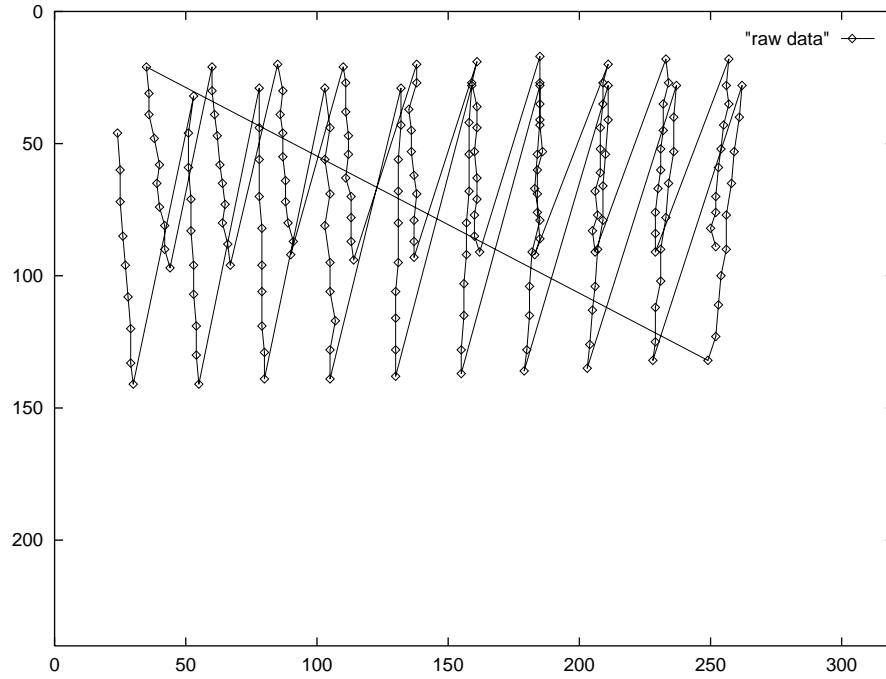


Figure 6. Track of LED in image space for the calibration experiment.
The coordinates are in pixels.

profile for the LED is uniform in all directions. The LED's position in future frames is estimated using a filtering/prediction algorithm described in [10].

The CMM is then commanded to move the probe to each predetermined 3D point within the frustum-like space and then stop. The vision system tracks the LED during the entire movement. When the probe stops, the vision system waits for the image to settle and then it records the (x, y, z) position of the probe tip along with its (u, v) image position. The two sets of values are concatenated together to form $(x_i, y_i, z_i, u_i, v_i)$ quintuplets.

Next we describe the results of one such calibration. For this experiment, the dimensions of the far calibration plane was 292x104 mm. The dimensions of the near calibration plane was 191x133 mm. The two planes were separated by 265mm. The exact corners of the calibrated work space were $(-116.00, -41.00, 22.0)$, $(176.00, -41.00, 22.0)$, $(176.00, 63.00, 22.0)$, $(-116.00, 63.00, 22.0)$, $(-30.00, -41.00, -243.0)$, $(161.00, -41.00, -243.0)$, $(143.00, 92.00, -243.0)$, and $(-30.00, 92.00, -243.0)$. The units are millimeters. The first four points determined the "far" plane. The last four points determined the "near" plane. 100 data points were created in the far plane and 99 data points were created in the near plane. The data points were found by interpolating between the corners of each plane.

Figure 6 is the path taken in image space by the LED as it moved through all of the calibration points. The location of the LED was found by thresholding the entire image at a predetermined intensity threshold. The LED's position was calculated to be the centroid of the resulting pixel cluster. The starting position of the LED is image location $(252, 89)$. The final position is $(24, 46)$. Castaño's calibration algorithm returns a calibration matrix. Figure 7 shows the result when the original 3D points are projected into image space using the resulting calibration matrix. The results appear to track the position of the original points very well. The entire effort took approximately 30 minutes of machine and operator time.

Uncalibrated vision results

In this section, we describe our experiments and experiences in developing new image Jacobian-based techniques for CMM control. Our method assumes that a mechanism exists for picking out the tip of the probe in the image and for indicating the desired scan path. In this system, the probe tip and scan path are indicated by the user on the graphic user interface.

We model the relationship between the camera system and image space using the following equation:

$${}^w \begin{bmatrix} U_i \\ V_i \\ 1 \end{bmatrix} = P(\lambda) \cdot T_{cam}^{-1} \cdot T_3 \cdot T_{probe}, \text{ where } T_{probe} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \text{ in end effector coordinates.} \quad (1)$$

Differentiating the above equation results in a relationship which relates image space with the 3D world. However, full knowledge of all the parameters is necessary to derive the complete Jacobian matrix. Since we assume that the calibration between image space and the world does not exist, we do not recover the image Jacobian directly.

Instead, the system estimates the control movements for a 2D Z-X plane in W . If y is kept constant, the differential relationship between image space and W can be written as follows:

$$\begin{bmatrix} \delta z \\ \delta x \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} \delta u \\ \delta v \end{bmatrix}, \text{ for } \delta y = 0. \quad (2)$$

where J , the estimate of the image Jacobian, is

$$J = \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} \frac{\delta z}{\delta u} & \frac{\delta z}{\delta v} \\ \frac{\delta x}{\delta u} & \frac{\delta x}{\delta v} \end{bmatrix} \quad (3)$$

Eq. (2) links small linear moves in image space to small linear moves in W .

The algorithm the system follows to derive the initial image Jacobian is:

1. Move the CMM to some position in W where TIP is observable in image space.
2. TIP projects to the point TIP' in image space. (Projections from 3D to 2D image space are denoted by the prime notation.)
3. Move the probe to a new position $TIP + (0, 0, \delta z)$. The change in the pose of the probe tip in image space $(\delta u_{\delta z}, \delta v_{\delta z})$ is recorded.
4. Move the probe to position $TIP + (\delta x, 0, 0)$. The difference between TIP' and the new position of the probe tip in image space $(\delta u_{\delta x}, \delta v_{\delta x})$ is recorded.

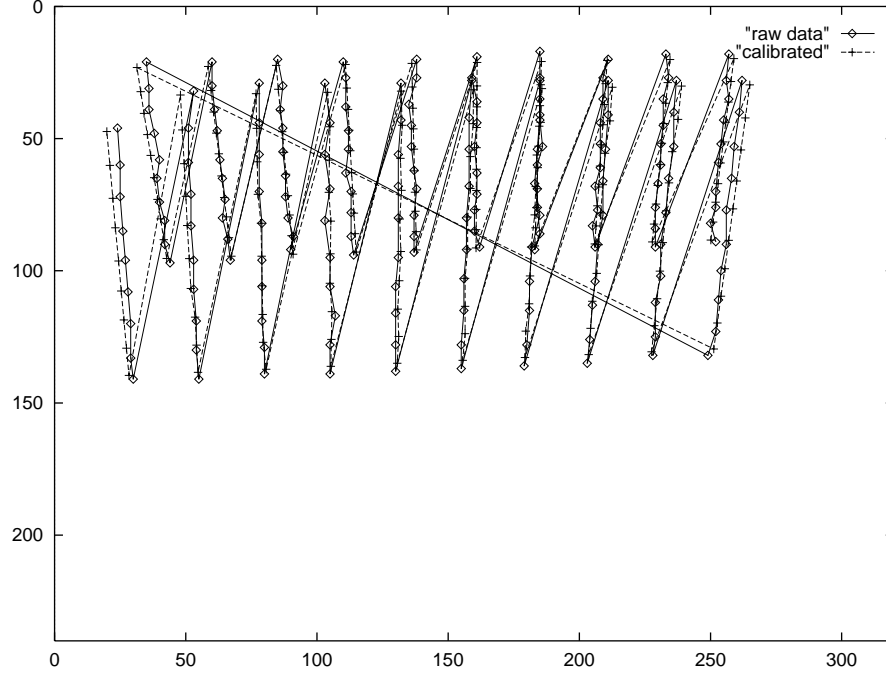


Figure 7. Projection of original 3D points onto the image plane using the calculated calibration matrix. Results are overlaid on original LED data points. The coordinates are in pixels.

5. These perturbations $(\delta z, \delta x)$, $(\delta u_{\delta z}, \delta v_{\delta z})$, and $(\delta u_{\delta x}, \delta v_{\delta x})$ are substituted back into equation (2) to find the image Jacobian.

The resulting Jacobian approximation, J , may be applied in the following manner. If $TIP'_{desired}$ is the desired location for the probe tip in image space and TIP' is the observed position of the probe tip in image space, we can compute:

$$\vec{e}' = TIP'_{desired} - TIP' \quad (4)$$

Where \vec{e}' is an image space vector which represents the difference between the desired tip position and the observed tip position. \vec{e}' can be converted into a 3D vector in W by:

$$\vec{e} = J \cdot \vec{e}' \quad (5)$$

\vec{e} can be used to approximate a position in W where the probe tip might project to $TIP'_{desired}$. This position, $\vec{e} + TIP$, can be fed to the CMM controller as the next desired position. Since the image Jacobian is not uniform over the entire imaged region, $\vec{e} + TIP$ is only a very rough approximation to the desired location. If the system only makes these error correcting moves, it is unlikely to converge to the desired solution. This is especially true when the initial error is large. Instead, the control algorithm updates its error estimate as soon as new image information becomes available. After δt , the time interval necessary to process a new image, the probe tip travels to some new position, TIP_i . Since the probe

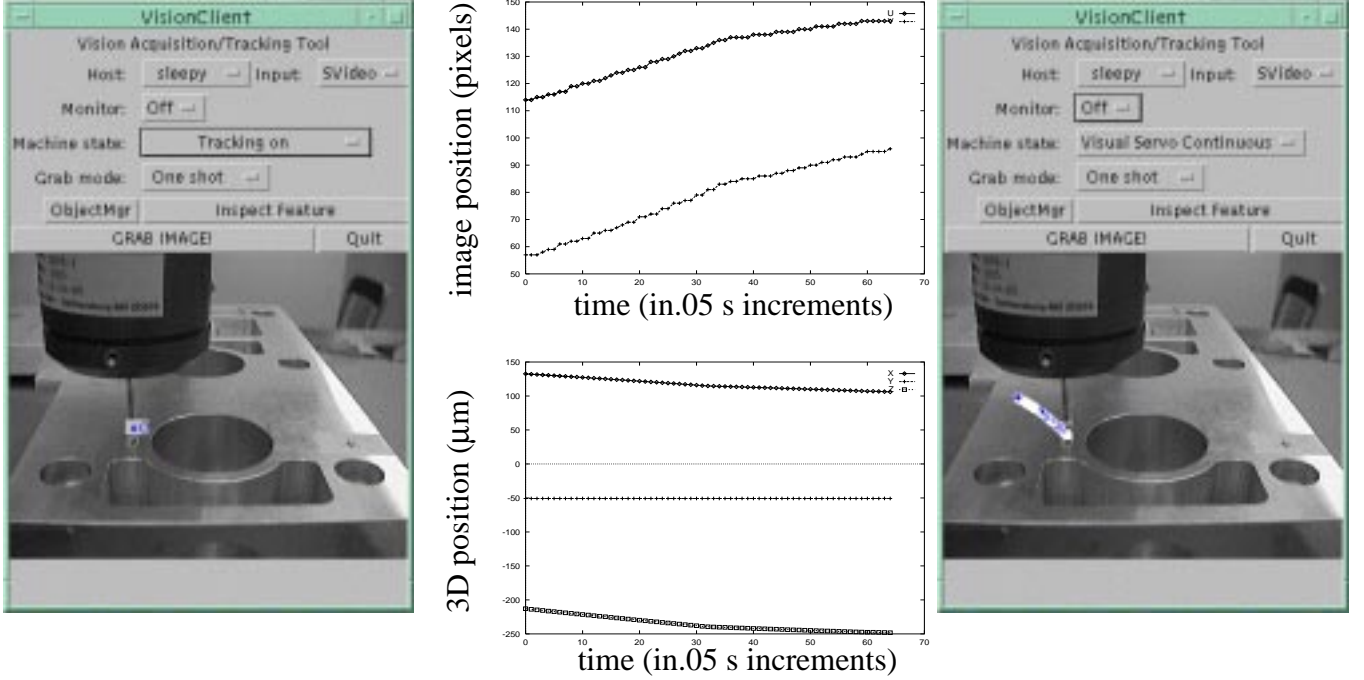


Figure 8. **LEFT:** Initial starting position of probe tip for experiment 1. **TOP:** U and V 2D position of probe in image space vs. time **BOTTOM:** X, Y, Z 3D position of probe vs. time **RIGHT:** Complete trajectory of probe tip as observed in image space.

started moving from TIP to $\hat{e} + TIP$, TIP_i must lie between TIP and $\hat{e} + TIP$. Instead of proceeding to $\hat{e} + TIP$, the system can use the knowledge that TIP travels to TIP_i and that it observed TIP' move to TIP'_i . The system calculates a new \hat{e}' based on these results and applies the associated \hat{e} to TIP_i . The low gain associated with the image Jacobian-based correction causes the system to be overdamped. It converges to the desired location without oscillation. Convergence is subject to many restrictions. First, the region where servoing is to take place should be fairly planar. Perspective projection of a plane preserves order making our initial estimate of the Jacobian sufficient. Second, the sign of the Jacobian must not change in the region. Third, the Jacobian must be well conditioned in the region. As long as the general sign of the image Jacobian is correct, the desired position will be achieved. Espiau et al. [5] have shown that convergence is guaranteed if the signs of the image Jacobian are correct and that the performance of the system increases as the approximation approaches the ideal value.

In our experiments, the time required to acquire the image Jacobian is about three seconds. The system makes three assumptions regarding the vision/probe/CMM system for these experiments. First, the perturbations must be observable by the imaging system. Second, the vectors which result from these perturbations must not be collinear. The cross product of the two initial position changes made to recover Eq. (3) must be non-zero. Third, the surface or object on which the servoing takes places must be relatively planar.

To test the usability of the image Jacobian, we devised a few experiments to determine how it performs under different conditions. In the first experiment, visually servoing to a point, the user indicates a position in image space where he would like the CMM to place the probe tip. Before the experiment is run, the NGIS system makes a guarded move to the part surface. The guarded move is in the -Y direction. Once contact has been established, the system is put into surface compliance mode. We have imple-

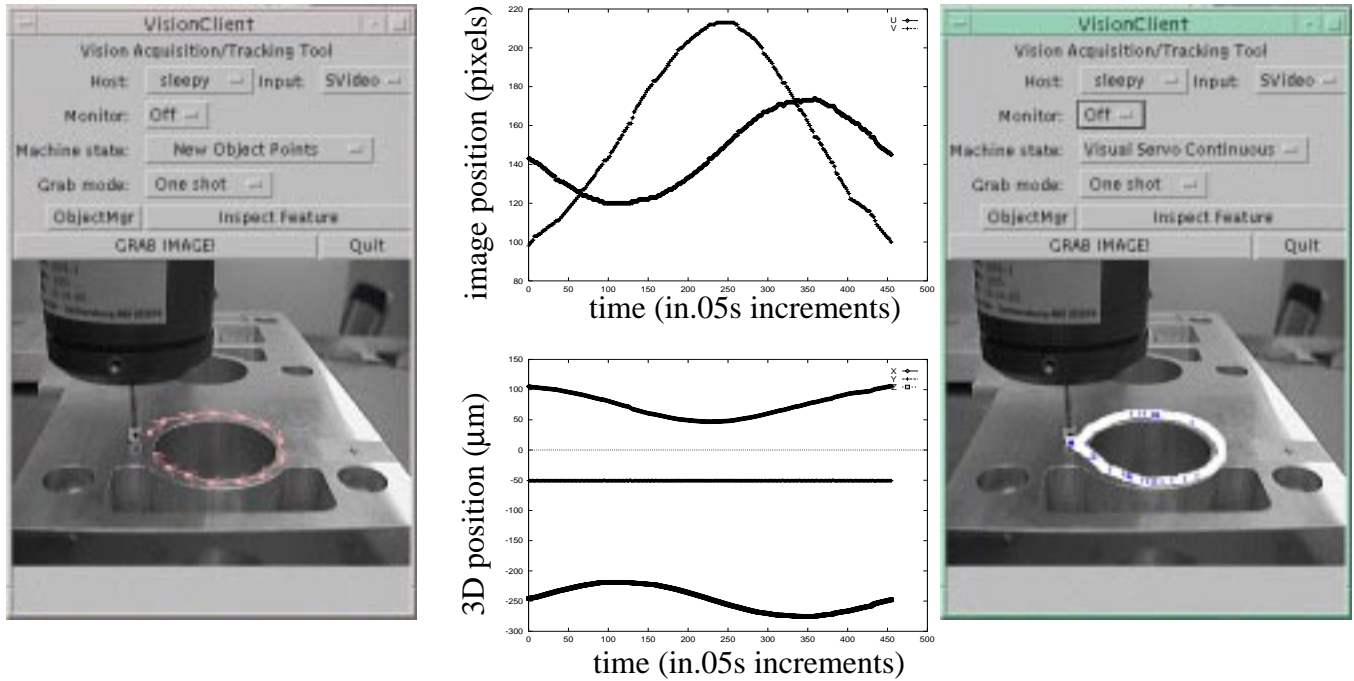


Figure 9. **LEFT:** User input path for CMM to scan. **TOP:** U and V 2D position of probe in image space vs. time. **BOTTOM:** X, Y, Z 3DD position of probe vs. time. **RIGHT:** Complete trajectory of probe tip as observed in image space.

mented a surface compliance scheme which allows the system move in the X and Z directions, while maintaining a constant Y probe displacement. The CMM uses proportional error control to maintain this displacement. The user interface sends the command VISUAL SERVO TO PT giving the (u, v) position in image space as a parameter.

Figure 8 (**LEFT**) shows the initial starting position of the probe as seen from the user interface. The small square on the probe tip indicates the position of the probe tracker in image space. As the probe moves in 3D, the square tracks the position of the probe in image space. The probe starts at image location (57, 114). The system estimates the image Jacobian at that spot. The user then clicks on position (96, 143). Figure 8 (**RIGHT**) is a trace of the probe tracker in image space. Figure 8 (**TOP**) shows (u, v), the probe's position in image space, plotted against time. Figure 8 (**BOTTOM**) shows the actual x, y, z position of the probe tip center in 3D plotted against time. In both graphs, the trajectory taken by the probe was fairly smooth and free of errors. At time 35, the probe velocity decreases. We implemented a probe velocity constraint which reduced the robot velocity when the probe approached within 10 pixels of the desired position. This was originally used with a much slower system to keep the probe from oscillating when it reached the goal position. In the future, we will examine more sophisticated control. During the entire operation, the system maintained probe contact with the surface of the part as seen by the flat line for Y in figure 8 (**BOTTOM**).

In experiment 2, the system started, once again, with the probe making a guarded move to the surface of the part. The user then clicked in several locations in the vision user interface indicating a path for the probe to follow. Figure 9 (**LEFT**) shows the starting image of the CMM with small arrows indicating the scan path for the CMM. Figure 9 (**RIGHT**) is trace of the path taken by the probe over the entire path. Figure 9 (**TOP**) shows (u, v), the probe's position in image space, plotted against time. Figure 9 (**BOTTOM**) shows the actual x, y, z position of the probe tip center in 3D over time. In both graphs, the trajectory taken by the probe once again was smooth and free of errors. During the entire scanning operation,

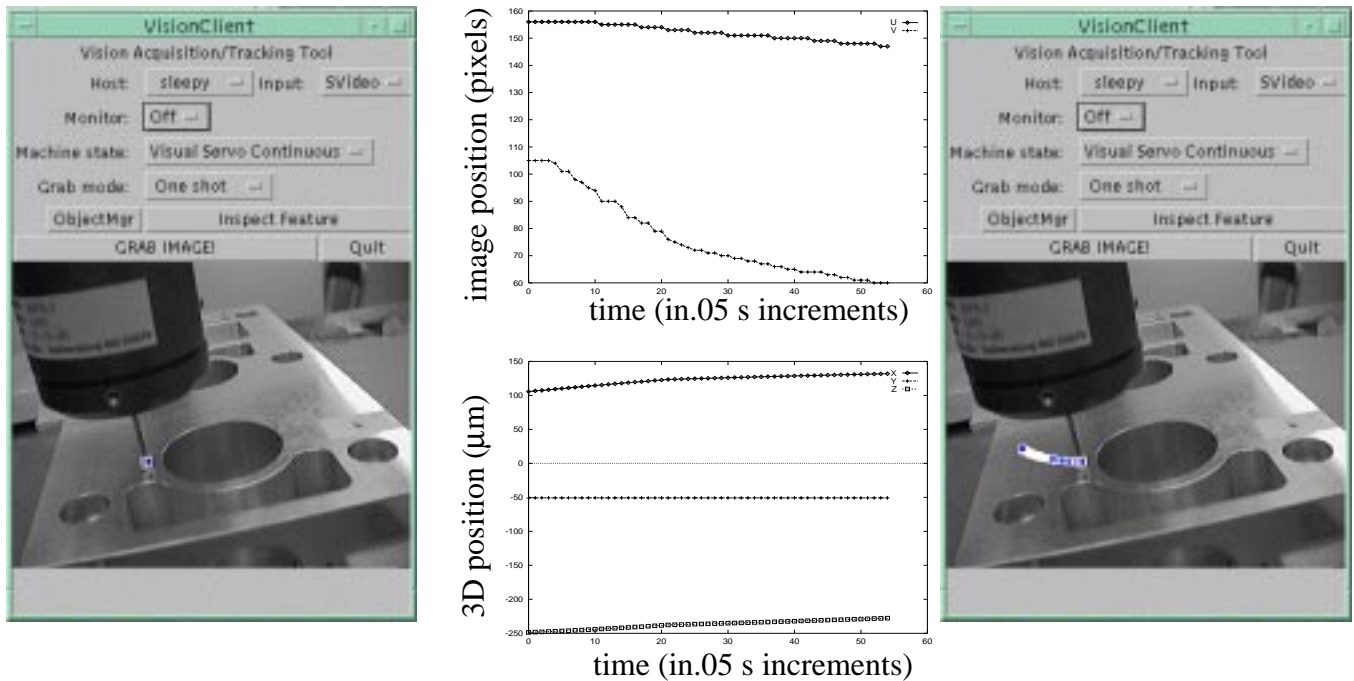


Figure 10. **LEFT:** After calculating image Jacobian, camera was rotated ~ 15 degrees counter-clockwise. **TOP:** U and V 2D position of probe in image space vs. time. **BOTTOM:** X, Y, Z 3D position of probe vs. time. **RIGHT:** Complete trajectory of probe tip as observed in image space.

the system maintained probe contact with the surface of the part.

Finally, in experiment 3 we repeated experiment 1 except in this case the camera system was rotated approximately 15 degrees from the position where the image Jacobian was calculated. Figure 10 (**LEFT**) shows the initial configuration of the probe system as viewed from the rotated camera. Figure 10 (**TOP**), (**BOTTOM**) and (**RIGHT**) are similar to their counterparts in experiment 1. Even though the camera position was disturbed, the system compensated for the disturbance and performed the desired VISUAL SERVO TO PT operation.

6. Conclusion and Future Research

Our system demonstrates that vision may be used to guide the acquisition of precise CMM measurements. We have shown that vision does not need to be calibrated with 3D space to be useful. We have demonstrated that an uncalibrated vision system adds flexibility to a measurement task. When coupled with an appropriate control strategy, uncalibrated vision may be used to direct CMM movement with almost no set-up time. We have begun to investigate how the internet may be used to advance CMM technology. In particular, we have addressed the issue of internet-based CMM control and the use of internet-safe CMM control strategies. We will continue to develop new sensor-servoed scanning techniques with an ultimate goal of transferring these new technologies to manufacturing plants.

7. References

- [1] Albus, J.S., Meystel, A.M., "A Reference Model Architecture for Design and Implementation of Intelligent Control in Large and Complex Systems," International Journal of Intelligent Control and Systems, Vol. 1, No. 1, pp. 15-30, 1996.

- [2] Allen, P., Timcenko, A., Yoshimi, B., Michelman, P., "Automated Tracking and Grasping of a Moving Object with a Robotic Hand-Eye System," IEEE Transactions on Robotics and Automation, Vol. 9, No. 2, pp. 152-165, 1993.
- [3] Castaño, A., Hutchinson, S., "Visual Compliance: Task-Directed Visual Servo," IEEE Transactions on Robotics and Automation, Vol. 10, No. 3, pp. 334-342, June 1994.
- [4] Cornell, G., Horstmann, C., Core Java, SunSoft Press, Mountain View, CA, 1996.
- [5] Espiau, B., Chaumette, F., Rives, P., "A New Approach to Visual Servoing in Robotics," IEEE Transactions on Robotics and Automation, Vol. 8, No. 3, pp.313-326, 1992.
- [6] Hashimoto, K., Visual Servoing, World Scientific, 1993.
- [7] Hollinghurst, N., Cipolla, R., "Uncalibrated Stereo Hand-Eye Coordination," Department of Engineering, University of Cambridge, No. CUED/F-INFENG/TR126, 1993.
- [8] Karara, H., Non-Topographic Photogrammetry, American Society of Photogrammetry and Remote Sensing, Falls Church, VA, 1989.
- [9] Martins, H, Birk, J., Kelly, R., "Camera models based on data from two calibration planes," Computer Graphics and Image Processing, Vol. 7, pp. 173-180, 1981.
- [10] Nashman, M., Hong, T., Rippey, W., Herman, M., "An Integrated Vision Touch-Probe System for Dimensional Inspection Tasks," Proceedings of the SME Applied Machine Vision '96 Conference, Cincinnati, OH, June 3-6, 1996.
- [11] Proctor, F., "Intelligent Controller Systems Development," <http://isd.cme.nist.gov/brochure/SoftwareSystems.html>, ISD Electronic Brochure, NIST, 1997.
- [12] Tsai, R., "A Versatile Camera Calibration Technique for High-accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV cameras and Lenses," IEEE Journal of Robotics and Automation, Vol. RA-3, No. 4, Aug. 1987.
- [13] Yoshimi, B., Allen, P., "Visual Control of Grasping and Manipulation Tasks," IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, pp.575-582, 1994.
- [14] Yoshimi, B., Allen, P., "Alignment Using an Uncalibrated Camera System," IEEE Transactions on Robotics and Automation, Vol. 11, No. 4, pp. 16-521, August, 1995.