The Virtual Vision Lab:
A Simulated/Real Environment for Interactive Education in Robot Vision *

P. K. Allen T. N. Jones J. P. Crosby P. A. McCoog
Department of Computer Science, Columbia University, NY, NY 10027

Abstract

The Virtual Vision Lab (VVL) is a project aimed at producing instructional lab modules for new and emerging techniques in robotic vision. VVL uses an integrated multi-media presentation format that allows the student to learn about robot vision techniques from textual sources, runtime algorithm codes, live and canned digital imagery, interactive modification of program parameters and insertion of student developed code for certain parts of the tutorial. It aims to translate a research paper in robot vision into a usable and understandable laboratory exercise that highlights the important aspects of the research in a realistic environment that combines both simulated virtual components and real camera imagery. The task the tutorial uses to demonstrate some basic principles of robotics and computer vision is the "pick and place task" which is implemented using a movable robot mounted camera that produces stereo imagery inside a robotic workcell.

1 Introduction

The Virtual Vision Lab (VVL) is a project funded by the National Science Foundation Combined Research-Curriculum Development program (CRCD) aimed at producing instructional lab modules for new and emerging techniques in robotic vision. VVL uses an integrated multi-media presentation format that allows the student to learn about robot vision techniques from textual sources, runtime algorithm codes, live and canned digital imagery, interactive modification of program parameters and insertion of student developed code for certain parts of the tutorial. It aims to translate a research paper in robot vision into a usable and understandable laboratory exercise that highlights the important aspects of the research in a realistic environment that combines both simulated virtual components and real camera imagery. To properly understand the use of robot vision algorithms, one should use a real robotics workcell. Due to the complexity, fragileness, and expense of actual robotic equipment, there are usually no hands-on resources available to undergraduate students that would allow them to test topics they learn in class. However, simulations can create a realistic robotic vision lab that is accessible to students. Our approach was to very accurately model our own robotics lab and translate it into a set of 3-D solid models that could be manipulated and moved as in a real workcell. All movement in the workspace is shown with animation, along with a simulated image from a robot mounted camera. A viewing window is constantly updated as the robot is moved, and the student can interactively modify the 3-D display of the workcell to vary the viewpoint and scaling. The animation and simulated camera output give the user a sense of realism and comfort that greatly enhances the lab modules. The lab also includes textual tutorials and interactive control of algorithms to allow the student to modify results and experiment with new ideas. This is further evidence of the merging of graphics and vision as inverse problems that can be seen to be duals of each other, which has important ramifications in engineering and education. Virtual environments may well be made available to students in the future that replicate costly engineering labs, allowing access to these environments by a much larger audience. Using both virtual environments and internet access, future engineering students may be able to access a variety of labs across the country interactively without leaving their own home institution.

*This work has been funded by NSF grant EEC-93-15517

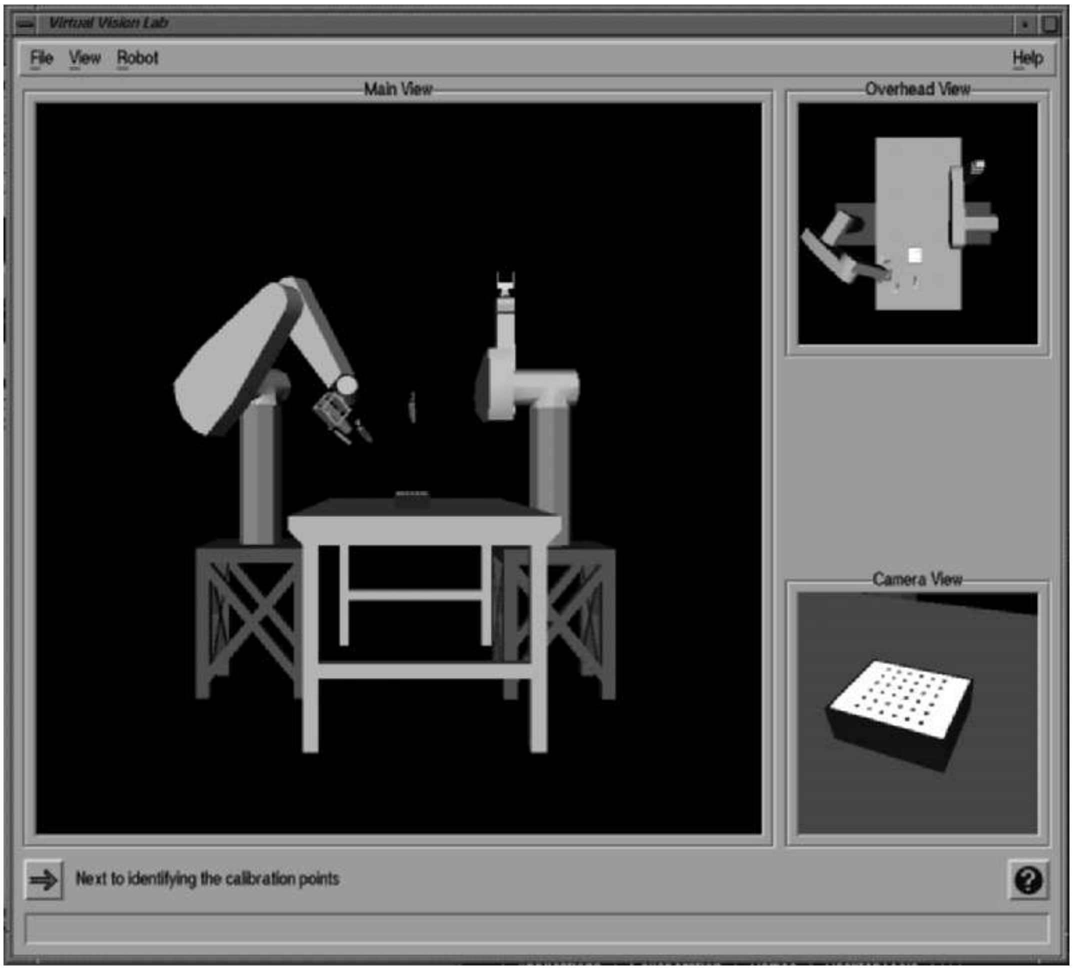

Figure 1: VVL Lab Environment with main, overhead and camera views

2 Simulated Environment

The simulated elements are actual items available in the Columbia University Robotics Vision Lab. There are two Puma 560 robots mounted on either side of a worktable. One Puma has a CCD video camera mounted as its end-effector. The other robot has a pneumatic parallel gripper. This environment can be seen in figure 1. The simulation of the lab environment was done using the 3D graphics toolkit Open Inventor [1] available on Silicon Graphics workstations.

The task the tutorial uses to demonstrate some basic principles of robotics and computer vision is the "pick and place task". It involves having a robot pick up an object from one location and placing it at another location. The pick and place task is one of the most basic and common robotic operations and it is the operation performed most often on production lines. Typically, the object to be picked up has an a priori known location in space, simplifying the task. In VVL, we use a standard video camera as an eye to locate the object within the workspace. Like humans, however, we need two camera locations to triangulate and find the exact position of the object in three dimensional space. In order to triangulate we must know the location of our "eyes". The process of finding the location of a camera is called camera calibration. The task is explained in detail later, but in summary the steps to our task are:

1. Move the robot to location 1 and perform camera calibration
2. Move the robot to location 2 and perform camera calibration
3. Image the object to be picked up from position 1

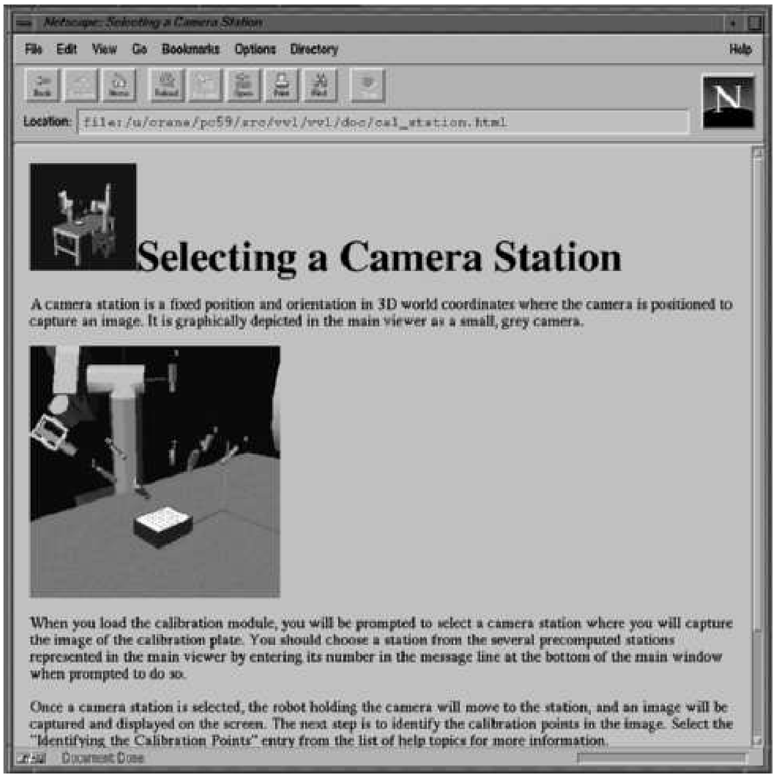

Figure 2: Netscape Help Viewer for Camera Station Selection

4. Image the object to be picked up from position 2
5. Match feature points from images 1 and 2 and triangulate to find the object location
6. Move robot to the object and pick it up
7. Image the final object position from position 1
8. Image the final object position from position 2
9. Match feature points from the images and triangulate to find the new object location
10. Move robot with the grasped object to the new object location, and place object

During the camera calibration stage, a calibration plate and light table are added to the scene. The calibration plate is glass plate with 36 precisely placed calibration disks. This plate and light table can be seen in figures 2 and 3. Once calibration is complete, the plate and light table are removed and several LEGO objects are added for the pick and place task; there is a rocket, a landscape with launch pad, and a rocket transport vehicle. These items can be seen in figure 5.

All movement in the workspace is shown with animation. The "Camera Window", seen in the lower right of figure 1, shows what is being viewed by the video camera mounted on the Puma 560 robot. The window is constantly updated as the robot is moved. The animation and simulated camera output give the user a sense of realism and comfort that is not existent in standard two dimensional still image simulations.

2.1 Hypertext Help

When the student begins the tutorial, two windows activate. One window is the main tutorial window and the other contains the help viewer. The main tutorial window (see Figure 1) contains three views of the environment: a "main view" that is similar to how a person would see the setup in an actual lab, an overhead view, and a view from the camera attached to one of the robots. A question mark button in the tutorial window links it to the

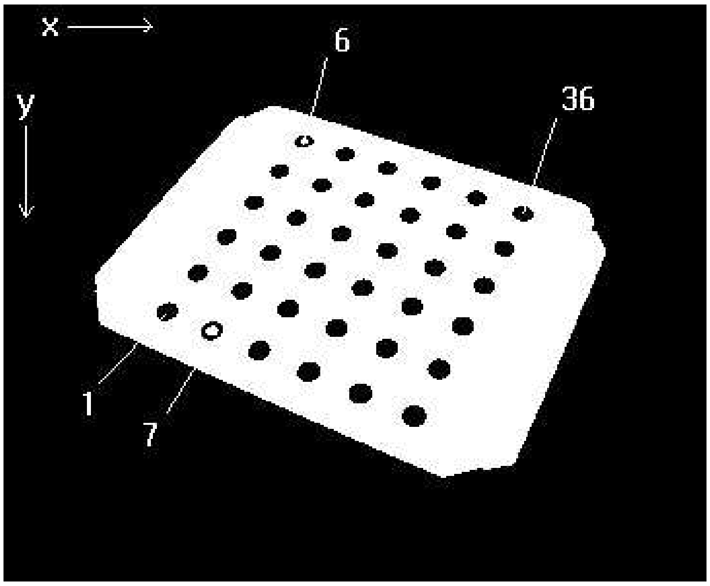

Figure 3: Real Image of the Calibration Plate taken from 1 of 9 precomputed views

help viewer (see Figure 2). Certain dialog windows and image windows that pop up throughout the course of the tutorial also contain help buttons. Whenever a help button or the question mark button is pressed, the help viewer window raises above all other windows on the screen and displays help that is pertinent to the current stage of the tutorial.

The help documentation was written in HTML [2] format and uses the Netscape [3] web browser as a help viewer. More and more users are becoming familiar with HTML files and Netscape-like browsers due to the increasing popularity of the World-Wide Web. Hopefully, a large portion of the users will be familiar with the format before they use the tutorial thus making it easier to use. HTML proved to be an excellent documentation method by allowing us to include images, links to other help files, and links to other sites that provide documentation and programming libraries that the student might need. Figure 2 shows the help viewer at one particular instance in the tutorial.

2.2 Calibration of the Camera

The student learns quickly that they cannot perform the pick and place task without calibrating the camera on the robot. Thus, the tutorial guides them to camera calibration first. A problem that other simulated environments frequently encounter is one of realism. They provide the user with perfect images, which is something that never happens in a real lab setting. To avoid this, we used images taken in our own lab. These images have noise, distortion, and other imperfections that the student would encounter given the same problem in a real lab. When the student specifies moving the robot in the workcell to an imaging position, VVL responds by moving the simulated robot to 1 of 9 prespecified locations. Previously, we have imaged the calibration plate and objects in the environment form each of these positions, and provide real imagery from the actual camera view. Figure 3 is a real image from the lab and the noise and imperfections are visible.

2.2.1 Calibration Algorithm

We use a two-stage algorithm for camera calibration using off-the-shelf TV cameras and lenses based on [4, 5] as well as a new class of separable restoration/reconstruction algorithms for geometric distortion correction based on [6]. We also have a hypertext link in the tutorial to the technical report describing this method. While the two-stage camera calibration algorithm is aimed at efficient computation of the internal and external parameters of a camera, the new class of separable restoration/reconstruction algorithms is intended to correct geometric distortion in terms of the parameters that characterize the camera. This method is also useful in image warping applications. In addition to the calibration and restoration/reconstruction algorithms, we also address in unconstrained nonlinear optimization that is part of the two-stage camera calibration algorithm and is useful in many other applications.

The overall transformation for calibration is in Fig. 4.

3D world coordinate (x_w, y_w, z_w)

$\downarrow$

Step 1
Rigid body transformation from (x_w, y_w, z_w) to (x, y, z)
Parameters to be calibrated: Rotation and Translation

$\downarrow$

3D camera coordinate (x, y, z)

$\downarrow$

Step 2
Perspective Projection with pin-hole lens geometry
Parameters to be calibrated: Focal Length

$\downarrow$

Undistorted image coordinate (X_u, Y_u)

$\downarrow$

Step 3
Radial lens distortion
Parameters to be calibrated: Radial Distortion Coefficient

$\downarrow$

Distorted image coordinate (X_d, Y_d)

$\downarrow$

Step 4
TV camera scanning and acquisition timing error
Parameters to be calibrated: Scan Factor and Image Center Coordinates

$\downarrow$

Image coordinate in computer frame memory (X_f, Y_f)

Figure 4: The Transformation from 3D World Coordinate to 2D Image Coordinate

Hypertext help links the student with a paper describing this method, and then the student can actually use the method to perform a calibration. The first step is in identifying the centroids on a calibration plate (see figure 3) from two camera stations. The tutorial gives the student a choice: the student can use the built-in centroid finding algorithm or the student can provide the centroid data. Thus, an instructor can assign a project where the students have to develop their own method of identifying and ordering the centroids. The students will never be able to complete the task without accurate centroid data, thus they will have to solve the identification problem to succeed. A student who chooses to write his own centroid finder is provided with a copy of the image (including a description of the image format) and can compute his own file of centroid locations to be used by the calibration algorithm. The student will find that the image has noise and needs a robust method of segmentation since the images are real, with shadowing, specularities and non-uniform gray scales. Success is also measurable since the actual centroid locations are known, and can be compared with student derived output.

2.2.2 Module Provided Centroid-finding Algorithm

The identifying algorithm is based on an algorithm for finding "blobs" in an image in Ballard [7]. The algorithm assigns every different region a different color. It does this by iteratively scanning the entire image with an L-shaped window from left to right and top to bottom.

Once all the "blobs" are identified, they must pass several tests in order to qualify as centroids. The first test checks to see if the number of pixels in the region is within a preset range of values. This test eliminates

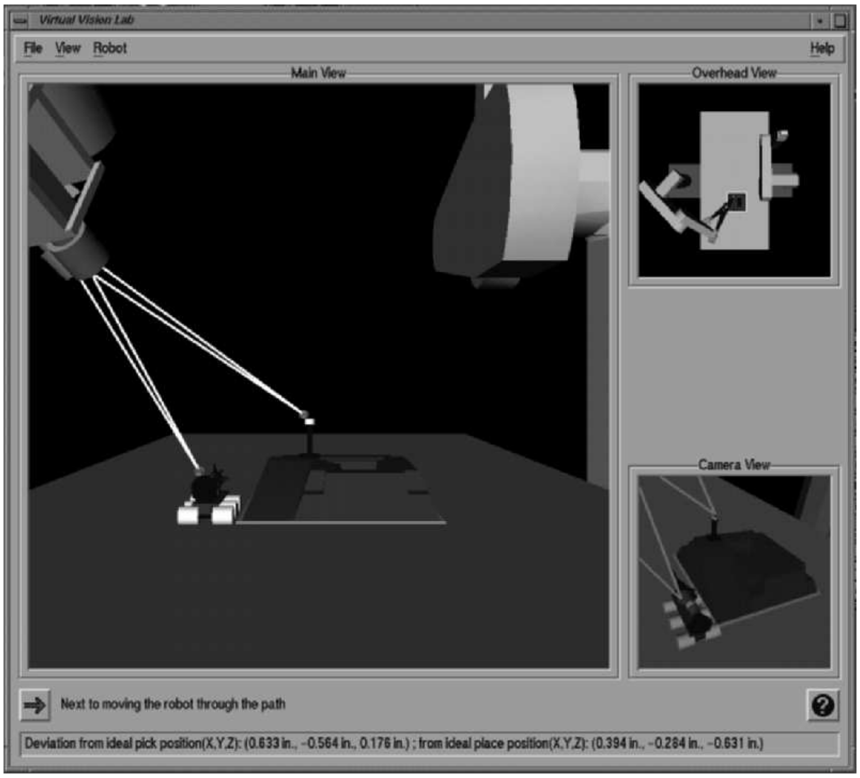

Figure 5: Triangulation

noise specks and large blobs. The second test calculates the ratio of the height of the region to the width. A centroid will be close to a circle, thus it will have a height to width ratio close to one. The ratio test is best at finding and eliminating regions that are one pixel wide or one pixel long. The third test compares the average x and y values of the region and the midpoint of the region. In a true centroid, these values will be very close. The final test verifies that the region is not on the edge of the image. Upon passing all of these tests, the region is labeled a centroid. This method is described in a help screen if the student wishes to create his own version of the algorithm.

2.3 Pick and Place Task

Once the student completes camera calibration, they are ready to try to succeed in the pick and place task. We tried to develop a task that was more interesting than the traditional "peg-in-hole" task that is associated with pick and place tasks. To do this, we used a LEGO environment consisting of a rocket and a launch pad. The student has to instruct the robot to pick up the rocket and place it on the launch pad.

The tutorial shows the student two images taken from the real lab of the rocket/launch pad environment. The images are from the two camera stations that the student selected to calibrate the camera. First, the student must identify the grasp feature point on the rocket in both images. Then, the student must identify the launch pad feature point in both images. This interaction is important to helping the student understand the concept of stereo triangulation.

When the student selects a point on an image, they identify a line in the three-dimensional world. The point is somewhere on the line. When they select the same feature point on another image, they identify a different line in the three-dimensional world. Again, the point they want is somewhere on the line. Since the points selected on both images are the same point (or close to it), the intersection of the two lines should be at the point in the three-dimensional world. Figure 5 shows the rays identified by the student's point selections and their intersections. In this case, the student was very accurate in both calibration and feature point selection as is evident by the accuracy of the three-dimensional points.

Once the student identifies the grasp point and the place position, the corresponding approach and departure positions are calculated. These points lead to an eight point path: start, pick approach, pick position, pick departure, place approach, place position, place departure, home. The manipulator robot moves to each position in succession. Before a move starts, the tutorial checks to make sure the position is reachable by the robot. During motion, the movement algorithm maintains a virtually constant bounding box check to detect whether the robot gripper (including rocket) collides with any part of the environment. If a collision occurs or the movement algorithm requests an impossible move, the operation aborts and the robot returns to home position. Also, there is the possibility that all moves would be possible without collision yet not accurate enough to successfully pick or place. If this occurs, the rocket is either not picked up or it is "dropped" on the table, whichever is appropriate.

If a student successfully completes the task, the tutorial displays an MPEG [8] movie of the real robots in the Robotics Lab at Columbia University doing the task. If a student fails, the help window pops up and displays information about the possible reasons for failure.

3 Conclusion

The goal of the Virtual Vision Lab tutorial is to add an active educational resource to the existing passive methods of learning computer vision. It will be an alternative, but not a substitute, to reading technical papers and books on the subject. We have consciously made the tutorial goal-based (see [9]). While such multi-media approaches are not entirely new, we have defined ways to ease the development process of such systems. Using Netscape and HTML allowed us to exploit a set of existing tools. In addition, new 3-D modeling languages such as Open Inventor have significantly decreased development time for complex simulated scenes. In summary, we think that the Virtual Vision Lab is representative of many such simulated environments that students can actually use as an alternative to hands on experience in a real laboratory. While a real laboratory is certainly preferable, it may not be economically feasible; our approach provides a possible solution.

The ability to allow students to interface their own code in the centroid identification phase gives the tutorial expandability. The expandabilty factor allows the students to solve a fundamental problem of computer vision in practice, not just in a problem set. Furthermore, it does not limit the scope of the algorithms that the tutorial can explain since the students can learn about any of a variety of algorithms or design their own to solve the identification problem. This makes for an interactive and entertaining tutorial.

Currently, there are two other modules under development. The first is an interactive tutorial on dynamic contours in vision [10] and a module on real-time tracking of robotic hands [11]. Further details can be found at http://www.cs.columbia.edu/robotics.

References

[1] http://www.sgi.com/technology/inventor.html.

[2] http://www.w3.org/hypertext/www/markup/markup.html.

[3] http://ftp.netscape.com.

[4] R. Y. Tsai. Synopsis of recent progress on camera calibration for 3D machine vision. In *The Robotic Review*. The MIT Press, 1989.

[5] R. Y. Tsai. 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*, RA-3(4), Aug. 1987.

[6] Terry Boult and Ming Chiang. Image warping: Imaging-consistent reconstruction, the integrating resampler. In *Proc. ARPA IU WOrkshop*, February 13-15 1996.

[7] D. Ballard and C. Brown. *Computer vision*. Prentice-Hall, 1982.

[8] http://www.cs.tu-berlin.de/mpegfaq/.

[9] http://www.ils.nwu.edu.

[10] A. Blake and A. Yuille, editors. *Active Vision*. MIT Press, 1992.

[11] K. Hashimoto, editor. *Visual Servoing*. World Scientific, 1993.