Proceedings of the 1999 IEEE
International Conference on Robotics & Automation
Detroit, Michigan • May 1999

Development of a Visual Space-Mouse

Tobias Peter Kurpjuhn
kurpjuhn@lpr.e-technik.tu-muenchen.de
Technische Universität München
Munich, Germany

Kevin Nickels
knickels@trinity.edu
Trinity University
San Antonio, TX, USA

Alexa Hauck
hauck@lpr.e-technik.tu-muenchen.de
Technische Universität München
Munich, Germany

Seth Hutchinson
seth@uiuc.edu
University of Illinois at Urbana-Champaign
Urbana, IL, USA

Abstract

The pervasiveness of computers in everyday life coupled with recent rapid advances in computer technology have created both the need and the means for sophisticated Human Computer Interaction (HCI) technology. Despite all the progress in computer technology and robotic manipulation, the interfaces for controlling manipulators have changed very little in the last decade.

Therefore Human-Computer interfaces for controlling robotic manipulators are of great interest. A flexible and useful robotic manipulator is one capable of movement in three translational degrees of freedom, and three rotational degrees of freedom. In addition to research labs, six degree of freedom robots can be found in construction areas or other environments unfavorable for human beings.

This paper proposes an intuitive and convenient visually guided interface for controlling a robot with six degrees of freedom. Two orthogonal cameras are used to track the position and the orientation of the hand of the user. This allows the user to control the robotic arm in a natural way.

1 Introduction

In many areas of our daily life we are faced with rather complex tasks that have to be done in circumstances unfavorable for human beings. For example heavy weights may have to be lifted, or the environment is hazardous to humans. Therefore the assistance of a machine is needed. On the other hand, some of these tasks also need the presence of a human, because the complexity of the task is beyond the capability of an autonomous robotic system or the decisions that have to be made demand complicated background knowledge.

This leads to the demand for a comfortable 3D control and manipulation interface. A very special kind of controlling device is the space-mouse that has been developed recently by [1] and [2]. The space-mouse is a controlling device similar to the standard computer mouse, but instead of moving it around on a table, which causes plane related reactions on a computer program, it consists of a chassis that holds a movable ball. This ball is attached to the chassis in such a way that it can be moved with six degrees of freedom: three translations and three rotations. This makes it possible for the user to control six degrees of freedom with one hand. Because of this, even complex robotic devices can be controlled in a very intuitive way.

One step further to a more intuitive and therefore more effective controlling device would be a system that can be instructed by watching and imitating the human user, using the hand of the user as the major controlling element. This would be a very comfortable interface that allows the user to move a robot system in a natural way. This is called the *visual space mouse.*

The purpose of this project was to develop a system that is able to control a robotic system by observing the human and directly converting intuitive gestures into movements of the manipulator. The hand serves as the primary controller to affect the motion and position of a robot gripper. For the observation of the user, two cameras are used. A precise calibration is not required for our method. In fact, the only calibration that is required is the approximate knowledge of the directions "up", "down", "left" and "right". If a translation or rotation of the camera moves the controlling hand of the human out of the view of the camera, the system will fail. The gripper of a PUMA 560 robot arm with six degrees of freedom is used as a manipulator. One camera is placed on the ceiling providing a vertical view of the controlling hand and one camera is placed on a tripod on the floor to provide a horizontal view (see Figure 1). Together, the cameras create a 3D work-space in which the user is allowed to move.

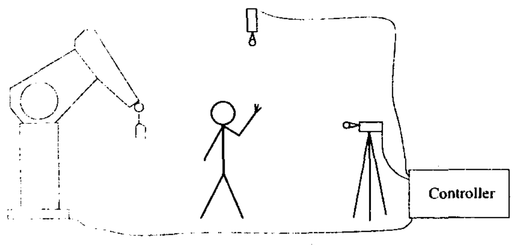

Figure 1: Structure of the Visual Space-Mouse.

The structure of the system yields some very powerful advantages. The first group of advantages is determined by the structure itself: the system provides a quantitative and cheap control unit without any aids or moving parts. That means there is no physical wear in the controlling system. This eliminates one potential source of failure, thereby making the system more robust.

The second group of advantages is determined by the basic concept of the system which provides upgrading possibilities. One possibility would be the use of sensor data combined with an intelligent robot control system. This would lead to a robotic manipulator with teleassistance and all its advantages [3]. Another possibility would be the implementation of hand gestures as a communication language with the system to produce a high level control-interface [4]. Additionally the work-space created by the two cameras is determined by the angle of view of the two cameras. Therefore it can be individually adjusted to the needs of the controlling task. A movement guided initialization sequence tells the system which hand serves as the primary input device. It is possible to keep track of several objects to perform more complex tasks. The scaling factor that translates the hand movement of the user into manipulator motion is fixed, but freely adjustable.

The remainder of the paper is organized as follows. In Section 2 we describe the major components of the system. We begin in Section 2.1 by giving a brief overview of the image processing unit. In Section 2.2 we describe the robot control portion of the system. Section 3 discusses two different approaches realizing the implementation of the visual space-mouse.

2 System Overview

The system of the visual space mouse can be divided into two main parts: image processing and robot control. The role of image processing is to perform operations on a video signal, received by the video cameras. The aim is to extract desired information out of the video signal. The role of robot control is to transform electronic commands into movements of the manipulator.

2.1 Image Processing

Our image processing unit consists of two greyscale CCD (charge coupled device) cameras connected to an image processing device, the Datacube MaxVideo 20. This Datacube performs operations on the video output. The operations of the Datacube are controlled by a special image processing language: VEIL [5], which is run on a host computer. In this way the data collected by the camera can be processed in a convenient way. This makes it possible to extract the desired information from the video output. In our case we identify, track and estimate the position of the hand of the user.

A special feature of VEIL is the use of blobs. A *blob* in VEIL is defined as a connected white region within a darker environment. The use of blobs makes it possible to detect and track special regions in the image.

The image processing unit is supposed to detect and track the hand. To achieve this task, within the environmental constraints imposed on the project, an image processing task was set up as following. The video output of the camera is convolved with a blurring filter and then thresholded. After thresholding the image blob-detection is applied.

Since the purpose of this project is to generate a prototype validating the benefits of the visual space-mouse, extra constraints were placed on the image processing. In particular, a black background in combination with a dark clothing is used. By doing so, the output image of the threshold operation gives a black and white image of the camera view where objects such as the hands or the face are displayed as pure white regions. This image is then imported to the blob-routine, which will mark every white region as a blob and choose one of the blobs to be the control blob.

To gain tracking of the desired hand, a motion guided initialization sequence was added. In the initialization sequence, the user waves the controlling hand in the workspace. The image processing unit records the image differences for several successive images and creates a map of these differences. After applying a blurring filter to suppress pixel-noise, resulting in spikes in this map, the image processing unit chooses the control blob. The blob in the current image that is closest to the region that changes most is chosen to be the control blob.

The values of this blob are stored in a global data structure to make it accessible to the robot control unit. Blobs other than the control blob are ignored in the controlling process. To ensure tracking of the hand after initialization without any sudden changes in the control blob, the bounding box is only allowed to change up to S_{cb} pixels each cycle in each direction. In the current implementation, this threshold value is set to 5 pixels. This causes the blob to get stuck to the hand and not to jump to other objects that are near the hand. One advantage to this is some measure of robustness to occlusion of the hand. If an object (either dark or bright) passes between the camera and the hand, the bounding box for that object will not match the bounding box for the controlling blob. Thus, the controlling blob will remain in the position it was before the occluding object appeared.

The orientation of the major axis of the object is computed by using the centered second moments of the object as follows (see [6]):

$$\varphi = \frac{1}{2}\arctan\frac{\widehat{m}_{xy}}{\widehat{m}_{xx}-\widehat{m}_{yy}} \qquad (1)$$
$$\widehat{m}_{ab} = m_{ab} - m_a \cdot m_b, \qquad a,b \in \{x,y\}$$

where $\widehat{m}_{xx}$, $\widehat{m}_{yy}$, and $\widehat{m}_{xy}$ are the centered second moments about the horizontal, vertical, and 45° axes, respectively. The second equation is used to compute these centered second moments, with m_{ab} and m_a representing the non-centered second and first moments about the appropriate axes, respectively. By using these equations, angles between $+\pi/4$ and $-\pi/4$ can be measured. When the real object oversteps an angle of $\pm\pi/4$ the result of Equation (1) will change its sign. The routine to measure the orientation of the hand takes care of this effect by causing the angle returned to be clipped to $\pm\pi/4$ when the orientation of the blob oversteps this border.

As there are two cameras each providing a different view, those computations have to be done for each image source. The image processing task is set up in such a way that it processes first the horizontal, and after that the vertical view. Both views are treated in one cycle.

Because the blob search is run on a host computer, the image has to be transmitted from the Datacube to a workstation over a bus network. The bus network is the bottleneck of the whole image processing unit, as illustrated in Figure 2. By shrinking the image to $\frac{1}{16}$ of the original size, much transfer time can be saved with an acceptable loss of accuracy.

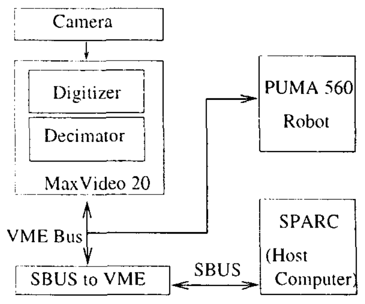

Figure 2: Bus structure and data-flow.

2.2 Robot Control

The second unit of the visual space mouse system is the robot control unit. The main elements of the robot control unit are the controller task, written in RCCL (Robot Control C Library) as a task level robot control language [7] [8], run on the host computer, and the manipulator itself.

Describing a manipulator task requires specifying positions to be reached in space (the *where*) as well as specifying aspects of the trajectory (the *how*). RCCL describes target positions using either Cartesian position equations or sets of joint angles. [9].

Cartesian position equations consist of several transform matrices that are multiplied. Each transform matrix describes a rotation and translation of the coordinate system. Together they form two systems of coordinate transformations: one on the right side and one on the left side of a position equation. Equation (2) describes the relationship of the two coordinate transformations.

$$\mathbf{T}_{start} \cdot \mathbf{T}_{variable} = \mathbf{T}_{base} \cdot \mathbf{T6} \cdot \mathbf{T}_{tool}, \qquad (2)$$

where $\mathbf{T}_{base}$, $\mathbf{T6}$ and $\mathbf{T}_{tool}$ represent the homogeneous coordinate transformations from the world frame to the robot's base frame, from the robot's base frame to a frame attached to link 6 of the robot, and from, and from a frame attached to link 6 of the robot to the tool frame. The transform $\mathbf{T}_{start}$ represents a homogeneous coordinate transform from the world coordinate frame to the initial position and orientation of the tool, and $\mathbf{T}_{variable}$ is a variable homogeneous coordinate transformation matrix that is continuously updated, thereby causing the manipulator to move to a goal position and orientation.

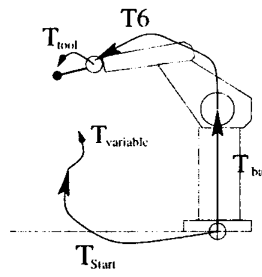

Figure 3: Effect of the position equation.

As both sides of the equations are said to be equal, both coordinate frame transformations have the same effect. That means that both sides of the equation start shifting from the same point and reach the same destination point. Equation (2) is solved for the matrix **T6**, describing the desired position of the manipulator arm:

$$\mathbf{T6} = \mathbf{T}_{base}^{-1} \cdot \mathbf{T}_{start} \cdot \mathbf{T}_{variable} \cdot \mathbf{T}_{tool}^{-1} \qquad (3)$$

To reach a point in space with the manipulator, you have to create the position equation, solve for **T6** in Cartesian space and transform the solution into joint space to achieve the desired values of the joint angles of the manipulator. With these joint angles the manipulator is able to reach the destination point. The trajectory generator in RCCL will then plan a path to the desired joint angles and update it as necessary.

The only inputs for the control unit are the two blob data-structures, described in Section 2.1. These data-structures represent the spatial position and orientation of the object being tracked. The controlling unit looks at the center of the blob rectangles in the image planes, which each contain a pixel-coordinate-system. The center of the camera views are said to be the origin of the coordinate systems. These pixel coordinates are translated into global coordinates for the manipulator. This is done by directly mapping the movements of the blobs into movements of the manipulator: blob motion in the image causes the manipulator to move in the corresponding direction.

The orientations of the hand can also be observed (see Section 2.1) and are transformed into manipulator movements. Two orientations, the rotation of the hand about the optical axis of each camera, can be observed directly from the images. The third orientation, the rotation of the hand about the horizontal axis, has to be computed from the image data. The value of the orientation of the (constant sized) hand could be computed, for example, from knowledge of the size, position, and orientation of the projections of the hand in the two images.

Every time these new values are passed to the control unit a new transform matrix is created with respect to the movement of the hand. This matrix is included in the coordinate transformation equation used to control the robot. The equation is solved for **T6**, the joint values of the manipulator are computed and the results are passed to the trajectory generator.

3 The Visual Space-Mouse

In some cases it is not possible to use two cameras to watch the controlling hand of the user. This can be caused by limitations on free space or on accessible hardware.

In Section 3.1 we discuss the space-mouse proposed previously, but we also suggest an approach to solve the dilemma of limited resources in Section 3.2.

3.1 Two Camera Space-Mouse

In our laboratory, only one image processing hardware device was available. Both camera views had to be processed by switching between two video channels. Combined with the transfer time via bus system (see Figure 2) this was a very time consuming procedure. So the biggest problem with the two-camera version was the speed of the image processing unit. The whole network slowed down the performance to 3 fps (frames per second). This forced the user to slow down hand motion in an unnatural way.

The use of a second image processing device would increase the image processing performance to the level seen in the one-camera version described below, allowing the user to move the hand at a natural speed. Nevertheless it could be shown that the tracking of the hand and controlling of the manipulator worked quite nicely in all six dimensions.

3.2 Space-Mouse with one Camera

In some cases limitations have to be applied to the structure of the visual space-mouse, as described above. The solution of this obstacle leads to a one-camera-version of the visual space-mouse.

By removing the overhead camera, any information about the depth of the controlling hand is lost. Any rotation with the rotation axis parallel to the image plane will just change the height and the width of the object. The sign of the rotation can not be determined easily. There are three dimensions of an object in a plane that are easily and robustly detectable: height, width and rotation in the image plane. The controlling task of a manipulator with six degrees of freedom is therefore very difficult with just 3 values. To handle this problem but keep the user interface intuitive and simple, a state machine was implemented in the controller.

The state machine consists of three different levels: two control levels and one transition level. The control levels are used to move the manipulator. The transition level connects the two control levels and affects the gripper of the robot arm.

When the palm of the hand is facing toward or away from the camera, the state machine of the controlling unit is in one of two control levels. In each control level the manipulator can be moved in a plane, by moving the hand in the up-down direction or forward-backward direction. The control levels differ in the orientation of the planes the manipulator can be moved in. The plane of control level 2 is orthogonal to the plane of control level 1 (see Figure 4). The planes intersect at the manipulator.

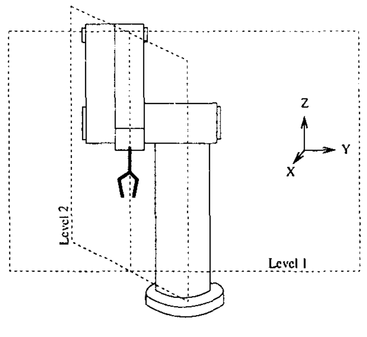

Figure 4: Motion planes of the two control levels.

To change the control levels the hand is turned so that the palm is facing down. In this mode the hand can be moved within the workspace without effecting the position of the manipulator. This mode is called the transition level (see Figure 5).

The transition level gets its name from its position between the two control levels, which are the actual steering levels. The task of the transition level is to connect both control levels and to perform additional actions on the workspace managed by the control levels. Those action are actuating the gripper and rotating the whole manipulator along its vertical axis.

By the use of the two planes, described previously, only a cubic space in front of the arm can be accessed. With the rotation along the z-axis this cube can be rotated and so the whole area around the manipulator is accessible. The rotation is initiated by rotating the hand in the plane of the image. This causes the robot to turn in steps of 10 degrees.

The gripper movement controls the opening and closing of the gripper. This movement is initiated by rotating the hand in the horizontal plane as shown in Figure 5. Placing the gesture for the gripper in the transition level has the advantage that any movement of the hand has no effect on the position of the manipulator, which will keep the gripper fixed during actuation.

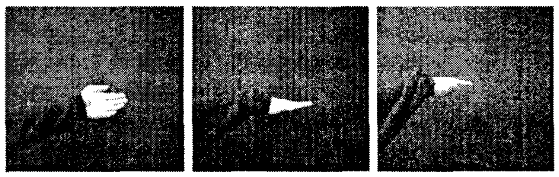

Figure 5: Control level gesture (left), transition level gesture (middle) and gripping gesture (right).

To determine when the state machine is supposed to change state, two threshold levels are computed and stored during initialization of the program. The first threshold level is set to $\frac{3}{4}$ of the height of the original blob(= *height*_3_4), the second one is set to $\frac{3}{4}$ of the width (= *width*_3_4) of the original blob. The state machine goes into the transition level when the height of the actual blobs falls below *height*_3_4. It goes into the opposite control level when the actual height exceeds *height*_3_4. In the transition level, the gripper is actuated when the width of the hand is reduced below *width*_3_4. The structure of the state machine is illustrated in Figure 6.

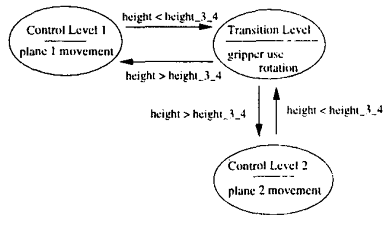

Figure 6: Structure of the state machine.

The origin of a control level plane is reset to the current position of the manipulator when the state machine enters that control level. This has the advantage that positions that are out of reach within the first attempt can be reached in the second attempt just by going into the transition mode, moving the hand to a more convenient position and then returning to the same control level.

The state machine always starts in control level 1. To visualize the state of the state machine of the control unit, the rectangle around the hand on the monitor is shown in a state-dependent color.

3.3 Discussion

An experiment was performed to validate the functions of the one-camera space-mouse. The task was to assemble a house out of three randomly placed wooden pieces. Several persons have been chosen to perform this experiment with minimal training, and each was able to successfully finish the task. The experiment showed that the state machine described above was usable. The biggest problem was that the gesture for the gripping movement was found to be unnatural. Most of the candidates not only turned the hand in the horizontal image plane, reducing the width of the hand below *width_3_4* as shown in Figure 5, but also turned their wrist. By doing so, they overstepped *height_3_4* and inadvertently transitioned into a control level.

One solution for this problem would be to introduce a third control level, as both control levels have been exhausted in terms of robustly detectable intuitive gestures. But this would require a significant change in the state machine, and the control would become less intuitive. Thus, it has not been implemented. Another possibility would have been to change the gripping gesture. This was not possible because of the limited possibilities of gestures that were bound to the robustly detectable dimensions: position, size, and orientation of the controlling blob.

3.4 Future Work

Both versions of the space mouse have several areas for improvement. Some of them are as follows:

1. Implementation of a motion model of the hand
2. Segmentation of the hand blob, for higher resolution control of the robot
3. Including sensor data for achieving teleassistance [3] (e.g. collision avoidance)
4. Implementation of a high-level gesture language
5. Adding a routine for filtering out the background
6. Implementation of a state machine in the two-camera version of the space-mouse

4 Conclusions

The objective of developing a high-level visually guided interface has been realized. As the experiment described in Section 3.3 showed, simple remote tasks can be performed with minimal training or teaching. This is a good demonstration of intuitive and convenient way in which a 3D interface can be operated. Additionally, several possible extensions to this implementation of the visual space-mouse have been proposed that would make it an even more powerful interface for control and manipulation.

References

[1] G. Hirzinger, B. Gombert, J. Dietrich, and V. Senft, "Die Space Mouse - Eine neue 3D-Mensch-Maschine-Schnittstelle als Spin-Off-Produkt der Raumfahrt," in *Jahrbuch für Optik und Feinmechanik* (W. D. Prenzel, ed.), Berlin: Schiele und Schön, 1996.

[2] G. Hirzinger, B. Gombert, and M. Herrmann, "Space mouse, the natural man-machine interface for 3D-CAD comes from space," in *ECUA '96 Conference for CATIA and CADAM users in Europe*, (Göteborg, Schweden), 1996.

[3] P. K. Pook, *Teleassistance: Using Deictic Gesture to Control Robot Action.* PhD thesis, University of Rochester, 1995.

[4] V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 384–401, July 1997.

[5] T. J. Olson, R. J. Lockwood, and J. R. Taylor, "Programming a pipelined image processor," *Journal of Computer Vision, Graphics and Image Processing*, Sept. 1996.

[6] B. K. P. Horn, *Robot Vision.* Massachusetts Institute of Technology, 1986.

[7] J. Lloyd and V. Hayward, *Multi-RCCL User's Guide.* Montréal, Québec, Canada, Apr. 1992.

[8] P. Leven, "A multithreaded implementation of a robot control C library," Master's thesis, University of Illinois at Urbana-Champaign, 1997.

[9] M. Spong and M. Vidyasagar, *Robot Dynamics and Control.* John Wiley & Sons, 1989.