A Mobile Platform for Nursing Robots

by
JOHANN BORENSTEIN AND YORAM KOREN[1]

Abstract

This paper describes a computer-controlled vehicle which is part of a nursing robot system currently under development at the Technion-Israel Institute for Technology. The platform of this vehicle can also be used for household robots. Design considerations, control algorithms, and the necessary sensory devices are discussed. The vehicle applies a motion control strategy which avoids slippage and minimizes position errors. Experimental results, performed on a prototype vehicle, are described as well.

1. Introduction

Even though experts seem to disagree on the feasibility of the all-around household robot [1], [2] some mutants of this species are already about to invade the private home [3]. On one hand, there are personal robots, advanced toys for the hobbyist, which are already commercially available, but of little practical use in the household. On the other hand, there are highly specialized, sophisticated robots, which are used for security tasks [4] or, as is the case here, for performing services for the disabled.

The "nursing robot" system is designed to serve bedridden patients by performing simple services such as operating electrical appliances or bringing objects to the patient's bedside according to the patient's spoken request. The nursing robot, however, is not supposed to apply any medical treatment to the patient. The workplace of such a robot would be usually confined to one room, either in a hospital or in the patient's home. This definition is important, since the constant presence of the patient as a supervisor for the robot's activities greatly facilitates the design of the robot in general and of the robot's mobile base in particular.

Most of the design considerations of the nursing robot are also applicable to household robots. Thus the mobile base will be discussed throughout this paper in general terms as a mobile platform for either nursing or household robots.

---

[1]Manuscript received June 1994; revised January 1985. This paper was supported in part by the Technion under Grant 003-712. The authors are with the Faculty on Mechanical Engineering, Institute for Technology, Technion, Haifa, Israel.

II. Design Considerations

There are several major differences between industrial robots and nursing or household robots.

1. A nursing or household robot must be mobile in order to reach a variety of working sites within the house, whereas an industrial robot is usually stationary. Even when mobility is required of the industrial robot, high accuracy is usually obtained by means of rails or guided wires imbedded beneath the factory floor. Obviously such means can not be used at home.

2. The domestic environment is largely disordered. This is not a disrespectful comment on housewives, but rather a comparison to the well-defined work site of a robot in industry. It is for this reason that object acquisition by a nursing/household robot must be based upon detection by sensors, rather than on absolute memorized locations.

3. **W**hile high-speed and low-cycle time are of major importance for the industrial robot, these attributes are less important for the household robot, and even less so for the nursing robot.

A. The Mechanical Design

One design of a mobile robot is used on the HERO 1 personal robot [5]. It employs a dc motor-driven wheel, which is also rotated about the vertical axis with the help of a stepping motor. Two additional, independent wheels on the rear axle provide stability. Even though the drive wheel is supplied with an optical encoder for position feedback, it has been found impossible to achieve acceptable path repeatability with this drive configuration [6].

The design frequently used for computer-controlled vehicles consists of two drive wheels, each with its own controlled dc motor or stepping motor [7] - [9]. One or two free-wheeling castors provide stability.

A similar design was chosen for the platform of the nursing robot, as shown in Fig. 1. Two dc motors with built-in reduction gears and optical encoders drive two rubber wheels, constituting the front axle of the vehicle. In the rear, two free-wheeling castors provide for static stability. Castors have been said to cause slipping at direction changes [10] , but this is not necessarily so, as shown in the Appendix. Another point to consider is the distance between the two drive wheels which depends on the width of the platform. It is desirable to place the two drive wheels as far apart as possible, for the following reasons.

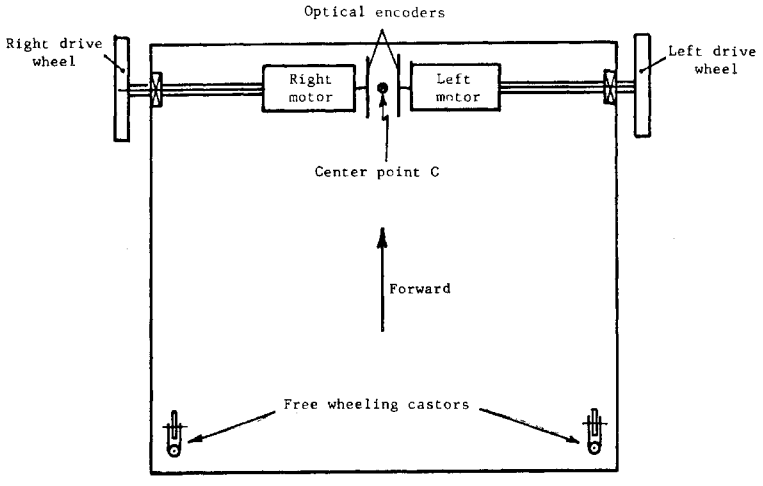

Figure 1: Bottom view of the mobile platform.

1. The static and dynamic stability of the vehicle are improved.

2. The influence of the encoder resolution on the orientational error of the vehicle is decreased. This may be seen by assuming that the vehicle is at rest. If one of the wheels is then turned an amount just within the encoder's resolution unit, and the other wheel remains at rest, then the vehicle would change its orientation by rotating about the fixed wheel and cause an error in the subsequent motion. The effect of this phenomenon is reduced by increasing the distance between the drive wheels.

3. During straight-line motion, mechanical and electrical disturbances will cause the motors to run at different angular speeds, resulting in a temporarily curved path. It can be seen by trigonometry that the radius of the curved path is directly proportional to the distance between drive wheels.

In the present design the distance between the two drive wheels is 600 mm.

B. The Controller Hardware

Fig. 2 shows a block diagram of the platform controller. For each motor, the computer issues an 8-bit binary speed command which is converted into an analog signal, amplified, and used to drive the motor. An optical encoder produces two 90° phase-shifted pulse trains which are fed into a directional sensing circuit (DSC) that issues an appropriate pulse train to a 4-bit, up-down counter. The

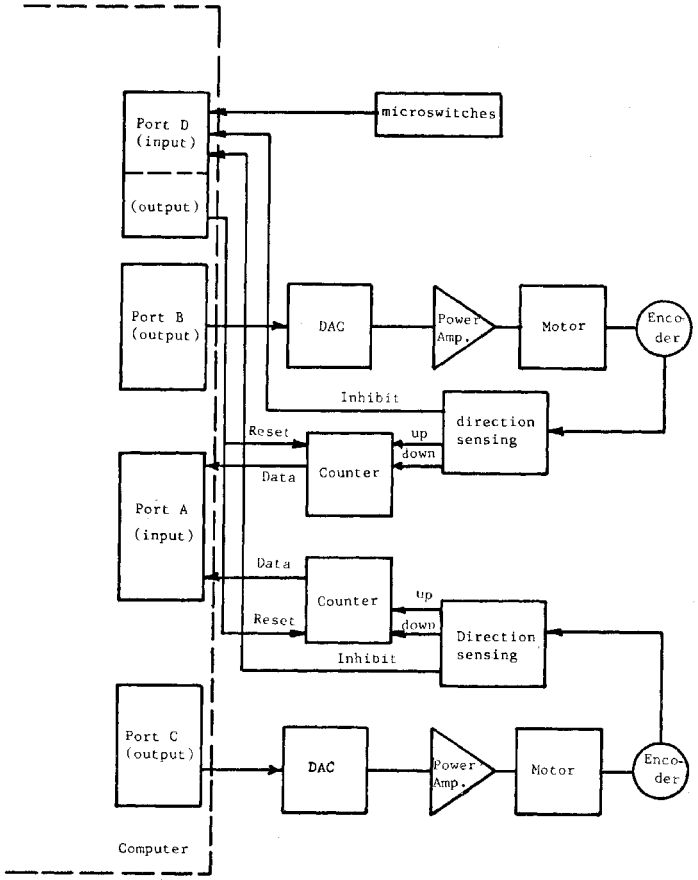

Figure 2: The controller hardware.

counter serves as a buffer, since the encoder pulses are transmitted faster than can be sampled. Both counters are sampled simultaneously and reset before the next encoder pulse arrives. An inhibit signal is provided by the DSC in order to avoid the counter reading at the instant when its state is changed.

The optical encoders are mounted on the respective motor shafts and their resolution is such that one pulse represents 2 mm travel of the drive wheel.

Several mechanically interconnected microswitches are positioned around the vehicle so that collisions may be avoided in time by bringing the vehicle to a stop before hitting an object. The controller also enables manual steering of the vehicle, with a joystick, which is not shown in Fig. 2.

C. The Programming Language

The control algorithm has been implemented in FORTH on a low-cost personal computer. As opposed to the approach of either writing in assembler language, or using high-level language on a development system and downloading the object code to a task computer, using FORTH offers the following advantages [11].

1. FORTH is much easier to write than assembler language and is only slightly slower in most applications.
2. Since FORTH is very compact, it may be resident in the task computer with all the peripheral devices (e.g., ADCs, DACs, I/O ports, timers) connected. The programmer may thus address these devices interactively. This is especially important for eliminating faults (debugging) which occur in conjunction with signal flow between the software and the hardware devices.

III. Motion Control

Motion Control means the strategy by which the platform approaches a desired location, and the implementation of this strategy.

A. The Control Algorithm

In order to represent the platform location relative to a fixed coordinate system (Fig. 3), three values must be given: the X- and Y-coordinate of the centerpoint C, which is located midway between the two drive wheels; and θ, which is the angle between the vehicle's longitudinal axis and the X-axis. If the vehicle has to travel from a known present location (x_0, y_0, θ_0) to a new location (x_f, y_f, θ_f), then the following procedure is performed (see Fig. 4). First, the length d and the slope ϕ of the straight line connecting the present and final locations are calculated as

$$\phi = \arctan\frac{y_f - y_0}{x_f - x_0} \qquad \textbf{(1)}$$

$$d = \sqrt{(x_f - x_0)^2 + (y_f - y_0)^2} \qquad \textbf{(2)}$$

Subsequently, the following strategy is performed.

1. The vehicle first turns about its centerpoint through an angle θ_1, which is calculated by $\theta_1 = \phi - \theta_0$.

2. The vehicle then travels along a straight line through a distance *d*. As a result the centerpoint will be at (x_f, y_f).

3. Finally, the vehicle turns about its centerpoint, through an angle θ_2, where $\theta_2 = \theta_f - \phi$.

For each of these steps a certain number, called the *terminal pulse count* (TPC), is calculated. The TPC represents the number of pulses that each motor has to produce in order to complete the command which can be either rotation or straight-line motion along a distance *d*. The TPC is always equal for both motors. However, during the platform rotation both motors rotate in opposite directions, but during straight line motion they rotate in the same direction.

Any movement between two given locations is performed in the sequence described above. The peculiarity of this approach is that it actually uses only two distinct kinds of motion: either motion in a straight line, where both wheels run at the same angular speed in the same direction, or rotation about the centerpoint C, where both wheels run at the same angular speed but in opposite directions. This simplification offers numerous advantages.

1. Since in either case the only task of the controller is to maintain equal angular velocities (measured in pulses per time unit), a relatively simple control system may be utilized. This system will be discussed later.

2. Both wheels are either simultaneously running or standing. Therefore, no case may occur where one wheel is running while the other one is standing, which would inevitably cause severe slippage.

3. The platform path is always predictable.

4. The platform always travels through the shortest possible distance (straight line or rotation "on the spot").

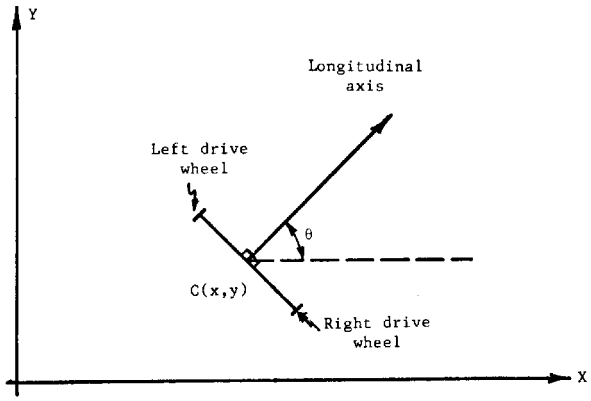

Figure 3: Representation of the platform location relative to a given coordiante system.

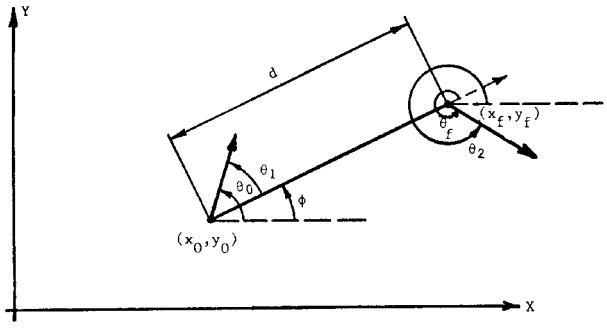

Figure 4: Procedure for traveling to a new location.

B. The Controller

A conventional controller for a mobile robot would consist of two independent velocity control loops, one for each motor, similar to the control loops used to drive the worktable in CNC milling machines or Cartesian robots [12]. Motion coordination in these systems is achieved by adjusting the reference velocities of the control loops, but the loop of one axis receives no information regarding the other. Any load disturbance in one of the axes causes an error which is corrected only by its own loop, while the other loop carries on as before. This causes an error in the resultant path. An improvement in the path accuracy can be achieved by providing cross-coupling, whereby an error in either axis affects the control loops of both axes. Such a cross-coupling method was applied to a Japanese mobile robot [13]. In this design each loop uses the position error of the other loop, but a signal proportional to the resultant path error is not generated.

The controller used here, however, applies an approach similar to the cross-coupled controller, which has been found advantageous for two-axis NC and CNC systems [14]. In this design the path error is calculated and fed as a correction signal to the loops. The main difference between the present design and the one used in CNC systems is that here the controller always maintains the maximum allowable speed of the motors.

The block diagram of the proposed controller is shown in Fig. 5. Pulses from the encoders are counted in the hardware up-down counters, which also indicate the direction of rotation of the motors by counting up for clockwise rotation and counting down for counterclockwise rotation. Just prior to being reset, the contents of both counters are simultaneously sampled (approximately every 50 ms) and added to the associated software counters. Thus each software counter holds a number that represents the total number of pulses generated since the beginning of a certain motion. Comparison between the absolute values of both software counters produces the error signal E_i where

$$E_i = | N_{1i} | - |N_{2i} | . \tag{3}$$

This error signal might generate a variable M_i defined by

$$M_i = K_p | E_i | \tag{4}$$

where K_p is a proportional gain.

A nonzero E_i indicates that one motor has been running faster then the other, and the sign of E_i identifies that motor. The speed of the faster motor is then reduced by subtracting M_i from its reference-velocity R, and leaving the reference velocity of the slower motor unaltered (this is

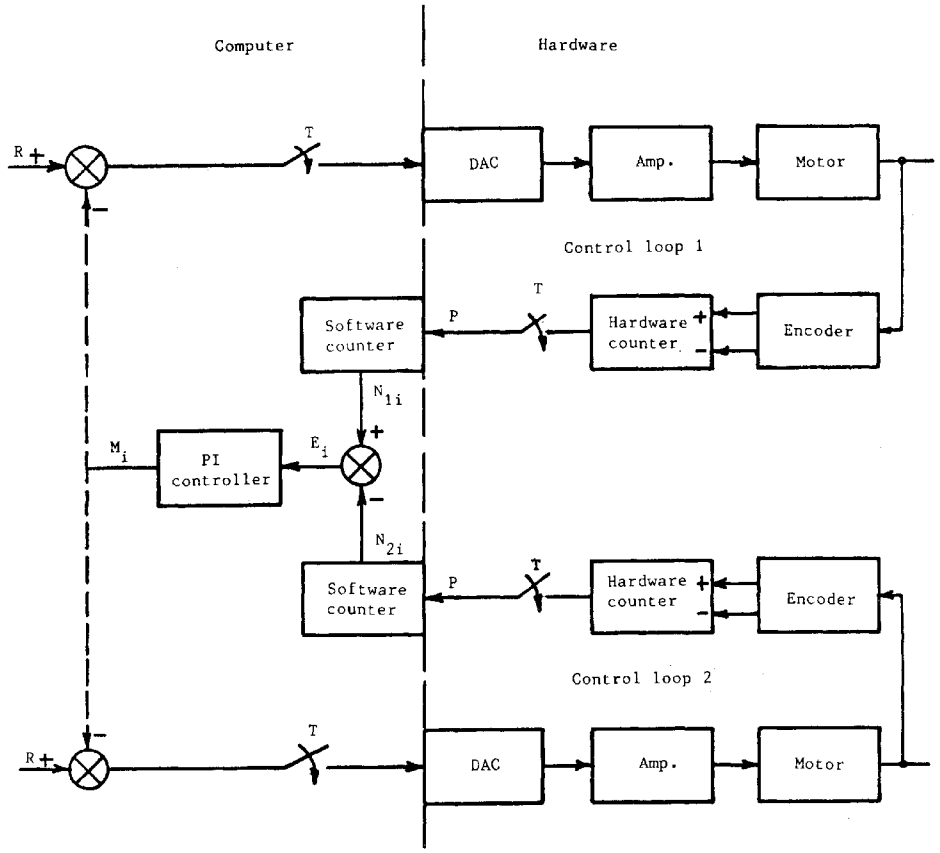

Figure 5: Cross-coupled control loops.

indicated by the dashed line in Fig. 5). Thus the velocities of both motors are effectively equalized.

Unless a disturbance occurs, both motors are fed by their maximum allowable voltage (except for the acceleration and deceleration phases). However, when a disturbance occurs, an error signal is generated, which, in turn, produces the correction variable M_i. Since the motors are already fed by their maximum voltage, the controller always subtracts the manipulated variable from the appropriate reference-velocity. This is legitimate, since it is the relative velocity of the motors, rather then their absolute velocities, which is of concern and must, therefore, be controlled.

Any temporary disturbance of the steady-state velocities will be successfully corrected by this proportional (P)-type controller. However, a *continuous* disturbance, as might be caused by different friction forces in the bearings (e.g., due to asymmetric load distribution), requires that *different* voltages be supplied *continuously* to the motors for a straight line motion. The P-type controller will supply different voltages only if a *constant difference* between the pulse counts of both motors is maintained. This is the case when the vehicle has traveled through a short arc, and has thus (undesiredly) changed orientation before resuming the straight line motion again (Fig. 6a).

In order to overcome this problem, an integration (*I*) action must be added into the controller. The *PI*-controller provides not only equal velocities, but also equal overall pulse count from the beginning of each motion. Therefore, this controller guarantees a zero steady-state orientation error of the platform for any constant continuous disturbance.

With the *PI*-controller, a continuous disturbance will only cause a temporary change in direction. After correction by the controller, the former direction will be resumed, leaving only a parallel distortion (ϵ) of the actual path (Fig. 6b).

For the *PI*-controller the equation of the controller becomes

$$M_i = K_c \sum_{n=0}^{i} E_n + K_p E_i \qquad (5)$$

where K_c is the integration constant.

The platform controller is easily implemented and requires only minimal computational effort. Since part of the required calculations (i.e., d, ϕ, θ_1, and θ_2) are performed before the nursing robot actually begins to move, they may be performed by the platform computer without affecting the sampling rate.

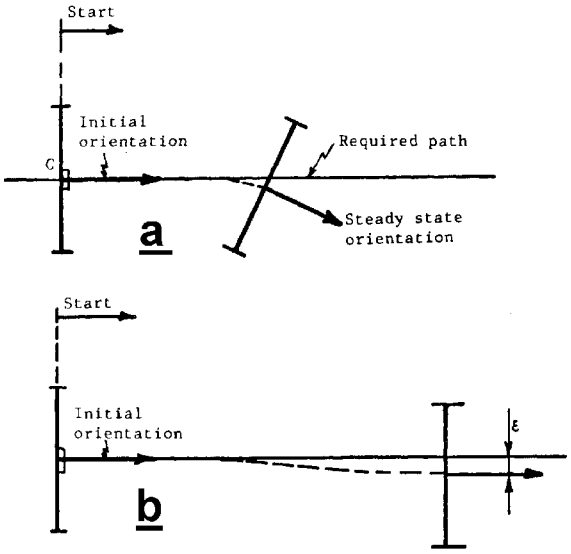

Figure 6: The effect of a continuous asymmetric load on the path.
a. With *P*-controller. b. With *PI*-controller.

IV. Experimental Results

A prototype of the platform has been built and tested. Experiments with the prototype have shown that the parallel distortion, inherent in the kind of control employed, is very small, on the order of magnitude of 10 mm per 10 m straight line travel.

Fig. 7 shows a typical path of the platform, carrying an asymmetrically distributed load (note the different scales for the *X*- and *Y*-axis in Fig. 7). The plot was obtained by real-time calculation of the momentary position of the centerpoint, which will be explained later. As may be seen from the graph, any disturbance causes a temporary deviation from the original direction. This disturbance however, is corrected and the vehicle continues in the original direction.

During the very last phase of the motion, the vehicle is decelerated. Lowering the input voltage to the motors gradually emphasizes the influence of friction in the bearings, which affects the control loop as a ramp disturbance. This effect causes an orientational error, which is corrected automatically after the deceleration phase by an overshoot correction phase (typically only a few pulses long, thus not recognizable on the graph). The final error of the platform (due to controller-dependent effects, but not including position errors introduced by mechanical inaccuracies) in this experiment was less than 3 mm after traveling a distance of 4 m.

Note that even though decelerated, the vehicle approaches the desired position with a certain velocity which causes an overshoot by a few millimeters. In order to correct this overshoot, each motor is independently moved until it reaches the previously calculated TPC which corresponds to the desired location. This action is performed without using the speed controller.

In another test, the vehicle was programmed to travel along a figure eight path. The path actually followed was calculated in real-time and is shown in Fig. 8. After returning to the original starting location, the vehicle's calculated position error was only 8 to 10 mm transversal and less than 1 ° rotational. Again, this error does not include mechanically caused inaccuracies. The real position error, including mechanical inaccuracies, was about 5 to 8 cm transversal and 1 ° rotational. These

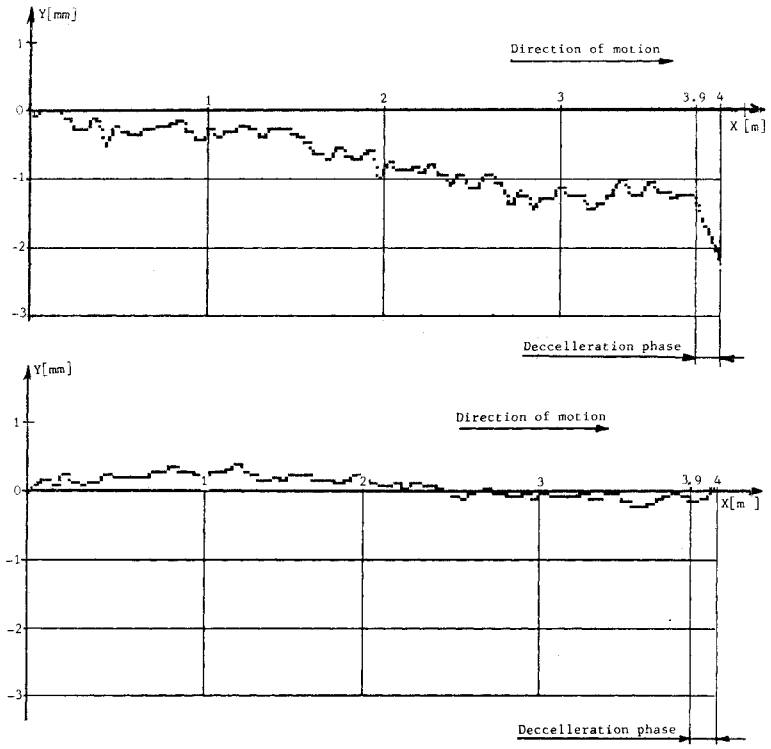

Figure 7: Parallel distortion from the required path.
a. With asymmetric load. b. Without load.

results compare favorably to the results of a similar experiment, described in [15]. However, th e vehicle used in [15] was faster, heavier, and did not halt at the corners of the programmed path.

The odometric technique, which was employed to calculate the actual momentary location of the centerpoint is described in [8] and [15]. Constantly updating this information allows us to find the platform's final position, independent of the control algorithm.

By comparing the actually measured final position of the vehicle to either the final positio n achieved by the cross-coupled control or to the odometrically calculated final position, no significant difference of accuracy for determining the actual final position could be found. This indicates that any inaccuracies in the final position are caused by control-independent effects, some of which are listed below.

1. The most significant inaccuracy is caused by directional uncertainty due to the limited resolution of the encoders. This problem may be partly solved by using encoders with very high resolution, but this would require higher sampling rates in order to maintain smooth control. Increasing the distance between drive wheels will also improve accuracy, since a single encoder pulse will have less influence on the platform's direction.

2. It is difficult to obtain rubber wheels with exactly the same diameter. In addition, unequall y distributed loads will slightly squeeze one wheel more than the other, thus changing its diameter. Wheels with different diameters cause the vehicle to travel along an arc, rather then along a straight line, even if the motors are running at exactly equal speeds, The radius of this arc and the orientational error are easily found from trigonometry

$$R = \frac{b}{u}(D+u) \cong \frac{bD}{u} \tag{6}$$

$$\alpha \cong \frac{L}{R} \cong \frac{Lu}{Db} \tag{7}$$

where
R = radius of the curved path due to different wheel diameters,
α = orientation error, in radians,
L = distance traveled,
u = difference of diameters of both wheels,
b = distance between drive wheels,
D = nominal diameter of drive wheels.

In our platform, b = 600 mm and D = 1 14 mm. For an assumed u = 1 mm, the platform performs a curved path with R = 600 × 114/1 = 68400 mm = 68.4 m. If the platform travels along a straight line, through a distance L = 10 m, the orientational error becomes α = 10/68.4 = 0.14 rad = 8.3 °. Obviously, this is an intolerably large error which emphasizes the necessity for a rigid wheel design.

3. There is a contact area, rather than a contact point, between the wheel and the floor. This causes an uncertainty about the effective distance between the drive wheels, creating inaccuracies when turning.

V. Sensors for Absolute Positioning

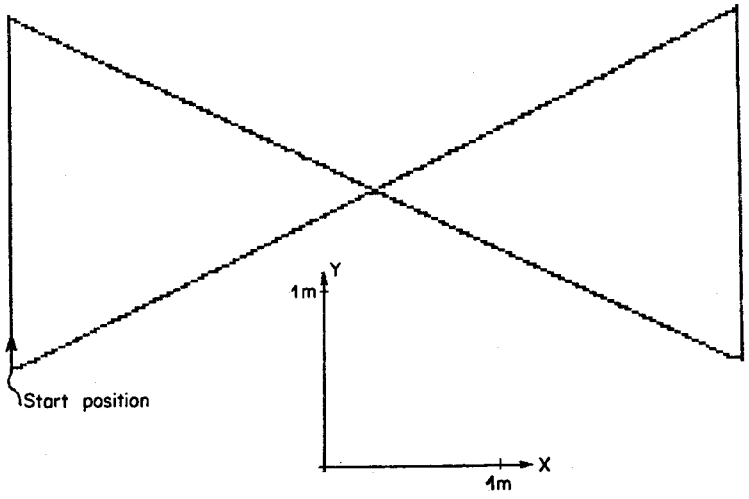

Figure 8: Actual trajectory for the figure-eight shape path.

As has been pointed out before, a wheel-driven vehicle cannot be expected to reach a given location with absolute reliability, because of wheel slip, errors introduced by crossing small obstacles on the floor, etc. Therefore, some sort of absolute position measurement (e.g., by means of navigation beacons) must be employed in order to determine the exact position. However, since these measurements require relatively long computation times, it has been suggested not to repeatedly perform absolute position measurements in order to control the motion [8].

The relative position measurement, based on the odometric technique as mentioned before, is much faster and is performed during motion. The disadvantage of this technique is the accumulating error due to wheel slip. Therefore, it is suggested that both techniques be used: relative position measurement during motion, and absolute position measurement when the vehicle is at rest (waiting for new commands).

Presently, no sensor has been installed on the vehicle, but it is planned to employ an ultrasonic range-finder for absolute position measurement (such as in [9]). If non-stationary furniture obstructs the walls of the room, one corner of the room would have to be cleared and declared as the "home" position of the vehicle. After performing a few tasks based solely on the relative position control, the robot would return "home" to update its absolute position. Obviously, this is not an absolutely reliable solution, applicable in the general case of the household robot. However, in the case of the nursing robot, where the patient can fulfill a supervisory function (considered a legitimate robotics approach [16]), no absolute reliability is required. In case the robot "gets lost," the patient would steer the robot manually, with the help of the joystick, to the approximate home position.

An ultrasonic device is also advantageous in that it may fulfill the additional functions of collision avoidance and path planning [9].

VI. Conclusions

A control strategy for mobile robots has been presented. Even though some aspects of this strategy apply only to human-supervised robots, the main control algorithm is generally applicable.

Experimental results show that the accuracy obtained by the control strategy has exceeded expectations as well as (in the case of the nursing robot application) requirements. These requirements are defined in relation to positioning errors due to unequal wheel diameters, encoder resolution, slip, and small obstacles overrun by the wheels.

The experiments proved our assumptions regarding the minor influence of the free-wheeling castors to be correct, as given in the Appendix.

Appendix:
Force Analysis of the Free-wheeling Castor

For quasi-static equilibrium (Fig. 9)

$$\Sigma F_x = 0 = F_r \sin\theta - F_s \cos\theta - C_x \quad (8)$$
$$\Sigma F_y = 0 = F_r \cos\theta + F_s \sin\theta - C_y \quad (9)$$
$$\Sigma M_c = 0 = T - d\,F_s \quad (10)$$

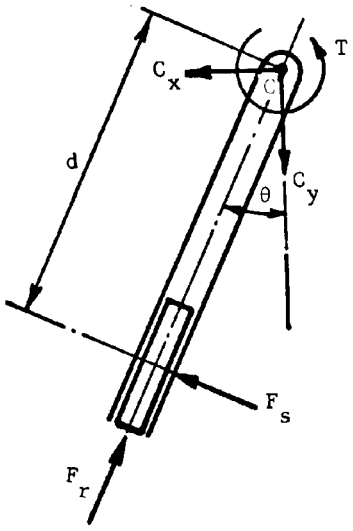

Figure. 9: Forces acting on a free-wheeling castor.

where

- T = moment transferred by friction at the horizontal bearing of the castor,
- F_r = rolling friction of the castor wheel,
- F_s = lateral friction force on the castor wheel,
- C_x = x component of the force at the vertical castor bearing,
- C_y = y component of the force at the vertical castor bearing,
- d = length of castor arm.

The moment T is determined by the friction in the horizontal bearing, and the maximal lateral disturbance on the vehicle occurs at $\theta = 90°$. Then, $C_x = F_r$, $C_y = T/d$ and the equations of equilibrium for the platform can be formulated in terms of T and F_r. (Fig. 10):

$$\Sigma F_x = 0 = A_x + B_x + 2C_x \quad (11)$$

$$\Sigma F_y = 0 = 2\,C_y - A_y - B_y \quad (12)$$

$$\Sigma M_B = 0 = -2T + 2C_x h - A_y b + C_y b \quad (13)$$

This analysis assumes a distance b between the castors. Assuming symmetry

$$A_x = B_x = C_x = F_r \quad (14)$$

$$Ay = 1/b\,(C_y b + 2C_x h - 2T) \quad (15)$$

$$= \frac{1}{b}\left(\frac{Tb}{d} + 2F_r h - 2T\right)$$

$$= \frac{1}{b}\left[T\left(\frac{b}{d} - 2\right) + 2F_r h\right]$$

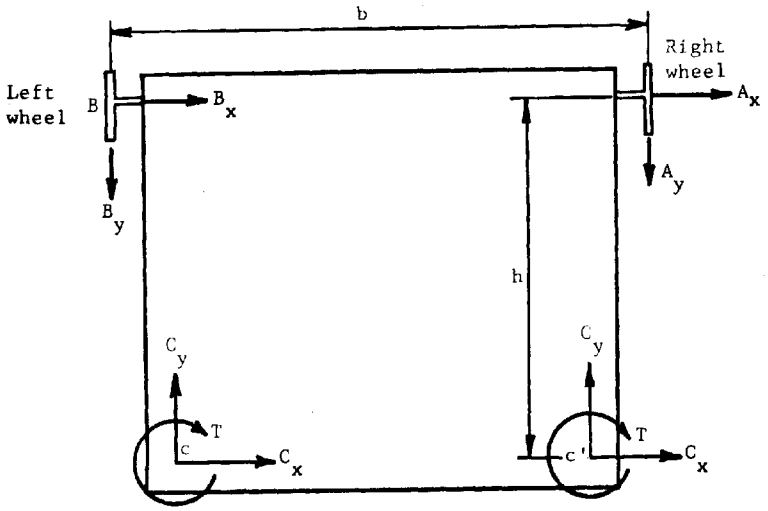

Figure 10: Forces acting on wheels of the platform.

where

A_x, A_y = x and y components of friction force at wheel A,
B_x, B_y = x and y components of friction force at wheel B,
b = distance between drive wheels,
h = distance between drive wheel axis and castors.

Thus in order to prevent slippage, the following inequality must be satisfied.

$$Nf > \sqrt{A_x^2 + A_y^2} \tag{16}$$

or

$$Nf > \sqrt{F_r^2 + \frac{1}{b^2}\left[T\left(\frac{b}{d} - 2\right) + 2F_r h\right]^2} \tag{17}$$

where
N = normal force on the drive wheels,
f = coefficient of friction.

It may be seen from (17) that in the ideal case of $F_r = 0$ and $T = 0$, no slippage will occur. This ideal case does not exist, of course. However, since N, F_r, *and* T depend strongly on the load distribution, (17) may be satisfied by positioning the load (normally the robot arm) close to the drive wheels (thus far from the free-wheeling castors). This will Increase N and decrease F_r and T. It is also evident from (17) that b should be as large as possible, while h should be as small as possible. This design will be limited by static and dynamic stability considerations.

References

[1] J. F. Engelberger, *Robotics in Practice,* K. Page, Ed. London: Aveburry, 1980, p. 135.

[2] M. W. Thring, *Robots and Telechirs.* Chichester, England: Ellis Horwood Ltd., 1983, p. 224.

[3] Y. Koren, *Robotics for Engineers.* New York: McGraw-Hill, 1985, pp. 8-10.

[4] Denning Mobile Robotics, Inc., 21 Cummings Park, Wobom, MA 01801, U.S.A.

[5] *Tech. Manual,* HERO Robot, Model ET-18, Heath Company, Benton Harbor, MI, 1982.

[6] C. Helmers, "Ein Heldenleben, *" Robotics Age,* pp. 7-16 and pp. 44-45, Mar./Apr. 1983.

[7] G. Bauzil, M. Briot, and P. Ribes, "A navigation sub-system using ultrasonic sensors for the mobile robot HILARE, " in *1st Int. Conf. on Robot Vision and Sensory Controls,* (Stratford-upon-Avon, U.K.), pp. 47-58, Apr. 1981.

[8] M. Julliere, L. Marce, and H. Place, "A guidance system for a mobile robot," in *Proc. 13th Int. Symp. on Ind. Robots and Robots,* (Chicago, IL), pp. 13.58-13.68, Apr. 1983.

[9] R. A. Cooke, "Microcomputer control of free ranging robots," in *Proc. 13th Int. Symp. on Ind. Robots and Robots,* (Chicago, IL), pp. 13.109-13.120, Apr. 1983.

[10] B. Carlisle, "An omni-directional mobile robot," *Developments in Robotics 1983.* Kempston, England: IFS Publications, 1983.

[11] D. R. Yoerger, P. Peterson and A. Buharali, "Control system implementation using FORTH, *" ASME 83 WA/DSC 28.*

[12] Y. Koren, *Computer Control of Manufacturing Systems* . New York: McGraw-Hill, 1983.

[13] S. Fujii *et at.* "Computer control of a locomotive rob ot with visual feedback," in *Proc. 11th Int. Symp. on Ind. Robots,* Tokyo, pp. 219-226, 1981.

[14] Y. Koren, "Cross- coupled biaxial computer control for manufacturing systems, *" Trans. ASME. J. of Dynamic Syst. Meas. and Contr*. vol. 102, pp. 265-272, Dec. 1980.

[15] T. Tsumura, N. Fujiwara, T. Shirakawa, and M. Hashimoto, "An experimental system for automatic guidance of roboted vehicle following the route stored in memory," in *Proc. 11th Int. Symp. on Ind. Robots,* (Tokyo, Japan), pp. 187-193, 1981.

[16] H. McIlvaine Parsons and G. P. Kearsley, "Human factors engineering in robotics," in *Proc. 13th Int. Symp. on Ind. Robots and Robots,* (Chicago, IL), pp. 9.41-9.46, Apr. 17-21, 1983.