A LOCOMOTION CONTROL SYSTEM FOR MOBILE ROBOTS

Jun'ichi Iijima, Yutaka Kanayaroa,* and Shin'ichi Yuta**

* The University of Electro-Communications, Chofu, Tokyo 182 Japan
** University of Tsukuba, Sakura, Ibaraki 305 Japan

AK TRACT

This paper discusses how to control and describe the smooth movement of a mobile robot which has two independent driving wheels (iDWs) that perform both steering and moving functions. Robots which have an IDW leg system can move to any point by using any combinations of two basic operations: revolving and going straight.

It is easy to describe movement with two basic operations only, but this may not be sufficient to guarantee smooth motion. For instance, when the robot is moving, a slight modification to its track direction is impossible without stopping it. Smooth motions are also needed for minimizing the time required to complete the movement.

Theoretically, an IDW leg system can draw a circular track with radius 0 - « if one controls the ratio of the two wheel speeds. This principle underlies much of the design of this control system.

The control system described in this paper uses three basic motion commands, supplemented by other commands that support these. A language has been designed for describing leg control algorithms. Examples of the usage of these commands and the language are shown here.

0. INTRODUCTION

Several mobile robots have been developed for AI research. Their mobile mechanisms have two essential functions:
(1) to ensure smooth movement, and
(2) to know the current position and direction in the world.
Steering methods perform the first function by one of the following schemes:
(1) steering and moving by driving wheels and steering wheels.
(2) steering and moving by driving wheels that have a steering function [I].
(3) steering and moving by two independent driving wheels (IDWs) that make the robot move in any arbitrarily selected direction [2][3][^1.

This research was supported by grants in aid for scientific research from the Ministry of Education of the Japanese Government.

We call the last one an IDW method. Robots with JDW $^{}_{eg}$ systems are widely used because, without any separate steering mechanism, they can move to any point by a combination of revolving and going straight.

It is easy to achieve movement using only these two basic operations, but this does not guarantee smooth motion. For instance, when the robot is moving, a slight modification to its track direction requires stopping the forward motion. Smooth motions are useful to minimize the time required to complete the movement.

Theoretically, an IDW leg system can draw a circular track with any arbitrary radius 0 ~ °°, given control of the ratio of the two wheel speeds. Furthermore, the radius of the track can be changed smoothly by increasing or decreasing the wheel speeds with constant acceleration. In order to control these capabilities, the command system described in this paper has been designed and has been installed on a mobile robot, Yamabico 3.1 [6].

Yamabico 3.1 has two microcomputer systems *for* the brain and the legs, and has a communication channel between the two. When the robot is to go to a destination, path finding and command sequence planning are executed by the brain, while the leg subsystem performs low level motor control functions. A similar functional distribution is used on the robot reported in reference [5].

1. TH£ LEG STATUS

Important information about the motion of the robot includes its position, direction, velocity and travelling distance, and this is called leg status. The brain issues commands to the leg subsystem to reference and update the leg status and to control movement, using the interface channel between these two subsystems. The leg subsystem confirms to the brain that each command has been executed. Variables of the leg status are listed in Table 1.

This information can be divided into static and dynamic information. Static information includes:

(I) The position of the robot
A point C in the robot is defined as the center of the robot. A direction D is defined on

Table 1 Leg status

variable	value
XX YY DD	x coordinate y coordinate direction } of the position of the robot
SS	travelling distance of the robot's center (odometer)
XXO YYO DDO	offset value of XX offset value of YY offset value of DD } in a command
SSO	offset value of SS in a command
VR VL	present speed of the right driving wheel present speed of the left driving wheel
STR	{true, if the robot is moving straight, false, otherwise.
CNS	{true, if the robot is moving at a constant speed, false, otherwise.

the robot, and called the reference direction. The robot lives in a world of two-dimensional standard coordinates. The position of the robot, P, is expressed in a 3-tuple (x, y, θ), where x and y are the standard coordinates of its location and θ is the angle between D and the X axis. The standard coordinates are given by an initialization process after the robot "awakens". (See Fig. 1)

In the leg status, the position (x, y, θ) is represented by XX, YY and DD in this order.

(2) The travelling distance

The leg status has an odometer, SS, which records the travelling distance of the robot from the initial location to its present location.

(3) Offset values in a command

XXO, YYO, DDO and SSO give, respectively, the difference between the current values and the values at the beginning of the most recent command affecting XX, YY, DD and SS. If the position is (x_0, y_0, θ_0) and the travelling distance is s_0 at the beginning of a command, then XXO, YYO, DDO and SSO are determined as:

$$XXO = XX - x_0$$
$$YYO = YY - y_0$$
$$DDO = DD - \theta_0$$
$$SSO = SS - s_0$$

where XX, YY, DD and SS represent the current values of the position and the travelling distance.

Dynamic information includes:

(4) The velocity of the robot

The velocity of the robot is expressed by the time derivatives of position: $\frac{dx}{dt}$, $\frac{dy}{dt}$ and $\frac{d\theta}{dt}$. In a robot controlled by the IDW method, one of these derivatives can be expressed by the other two. Therefore a pair of driving wheel speeds can express the velocity of the robot.

We adopted this representation. VR and VL are the current speeds of the right and left wheels, respectively, and take positive values when the robot is moving forward.

(5) Flags

There are two flags, STR and CNS in the leg status. The flag STR is true if the robot is moving straight, otherwise false. The flag CNS is true if the robot is moving at a constant speed, and false otherwise.

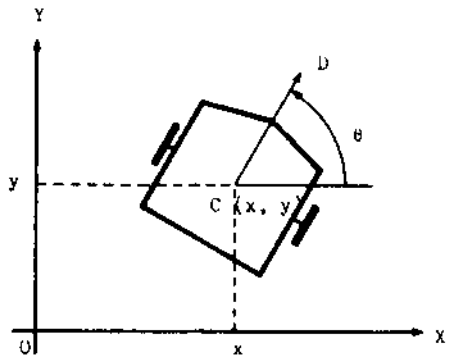

Fig. 1 Position of the robot

2. THE COMMAND SYSTEM

The brain sends orders for robot movement to the leg subsystem through a command sequence. There are two kinds of commands:

(1) motion commands
(2) control commands.

Motion commands are associated with the movement of the robot; control commands are related to leg status and to the control of the execution of a command sequence. Each command in the sequence is interpreted and executed by the leg subsystem. When a command has been executed, the leg subsystem returns replies to the brain. The information in the replies includes:

(1) the termination of the command (normal or abnormal termination), or
(2) the value of the variable in the leg status, or
(3) signalling that an interval of distance or angular movement has been travelled.

Syntax and semantics of the commands are described in 2.1 and 2.2, respectively, where the syntax is a reference language for describing leg controlling algorithms. Examples of the command sequences are shown in 2.3.

2.1 Syntax for the Command

The context-free syntax of the skeleton of the command system is described using a variant of the Backus-Naur Form, where braces { } denote zero or more repetitions and square brackets [ ] enclose optional items.

(1) Command sequences

```
<command sequence> ::= <command> {;<command>}
```

(2) Commands

```
<command> ::= <motion command> |
              <control command>
```

(3) Motion commands

```
<motion command> ::= <operation>
                     <speed specification>
                  [until] <terminating condition>]
<operation> ::= gostraight | turn with
                <radius> | spin
<speed specification> ::= in <speed>
<terminating condition> ::= <variable 1> =
                            <value> | speed
                            = <speed> | CTR |
                            CNS
```

(4) Control commands

```
<control command> ::= <get command> |
                      <update command> |
                      <set command> |
                      <reset command> |
                      <queue command>
<get command>     ::= get (<variable 1>)
<update command>  ::= update (<variable 2>,
                      <value>)
<set command>     ::= set (<interval>,
                      <value>)
<reset command>   ::= reset (<interval>)
<queue command>   ::= terminate | cancel
```

(5) Miscellaneous items

```
<variable 1> ::= <variable 2> | XX0 | YY0 |
                 DD0 | SS0 | VL | VR
<variable 2> ::= XX | YY | DD | SS
<interval>   ::= DDI | SSI
```

Detailed definitions of <speed>, <radius> and <value> are not given here.

Examples

(1) Motion commands

```
gostraight in v until SS0 = X
turn with r in v until DD0 = d
spin in v
```

(2) Control commands

```
get (SS)                update (XX, 0)
set (SSI, 10)           reset (DD)
terminate               cancel
```

2.2 Semantics for the Command

Each command in the sequence from the brain is accepted by the leg subsystem in its order in the sequence. Each motion command in a command sequence is stored in a first in first out (FIFO) storage and each control command is immediately executed by-passing the FIFO store. After the first command in the storage is executed and terminated, it is discarded and the next command is taken to be processed. If there exist *no* commands in the FIFO storage, the command 'gosfraip.ht in 0' is executed; this command never terminates.

Operations of the motion commands are described in 2.2.1. Section 2.2.2 describes operations; of the control commands.

2.2.1 Motion Commands

The motion command causes the robot to do the designated movement. The command is interpreted and executed when the command is placed in the first position of the FIFO storage. When the terminating condition is satisfied, the current command is terminated and the next command occupies the first position of the FIFO storage.

There is a transient state between the execution of two commands. In a transient state, each driving wheel moves with uniform acceleration, as discussed later.

The motion command consists of three specifications: an operation, a speed specification, and a terminating condition; the last of these can be omitted. There are three operations: going straight, moving along a circle of radius r, and revolving. The speed specification determines the speed of the robot's center when it is going straight or moving on a circular path, and the speed of the right wheel when the robot is revolving.

The operation and the speed specification of the command jointly determine two other values: the steady-state right and left wheel speeds. These are called the right target speed (rts) and the left target speed (its), respectively.

Details of the motion commands are:

(1) Gostraight command

This command causes the robot to move straight at the specified speed v until the terminating condition is satisfied. The speed v is positive if the robot is to move forward and negative if backward. The target speed of each wheel is v. The robot is placed in the position p_0 (x_0, y_0, θ_0) at the beginning of the command, and travels a distance s by executing the command (See Fig. 2). The leg status after the execution of the command changes as follows:

$$SS = s_0 + s$$
$$XX = x_0 + s \cos \theta_0$$
$$YY = y_0 + s \sin \theta_0$$
$$DD = \theta_0$$

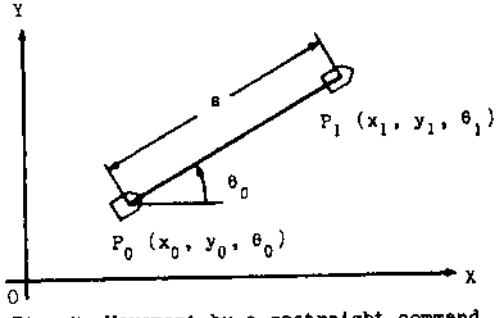

Fig. 2 Movement by a gostraight command

where s_0 is the value of SS at the beginning of the command execution.

(2) Turn command

This command causes the robot to move along a circule of radius r, specified by the <radius>, and it is terminated when the terminating condition is satisfied (See Fig. 3). If the robot is directed to move at a speed v then the target speeds rts and lts are $\frac{r + dt}{r} v$ and $\frac{r - dt}{r} v$, respectively, where dt is half of the tread distance, and r is the radius of the circle which is followed by the robot. If r is positive then the robot steers toward the left, otherwise toward the right. The speed v is positive if the robot moves forward, negative if it moves backward.

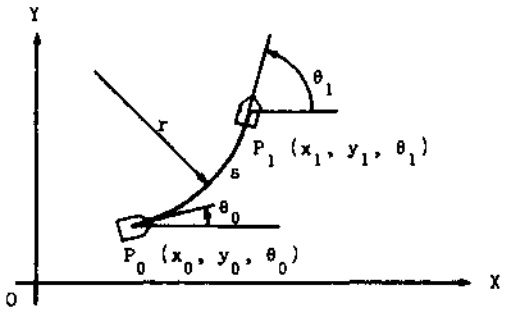

Fig. 3 Movement by a turn command

(3) Spin command

This command causes the robot to pivot on its center until the terminating condition is satisfied. If the speed v, given by the speed specification, is positive then the robot revolves counter-clockwise, and if negative then it revolves clockwise. The target speed rts is v and lts is -rts, so that the wheels move in opposite directions. (See Fig. 4) Each variable of the leg status changes as follows:

SS, XX, YY : no change

DD = θ_1, if the new direction is θ_1

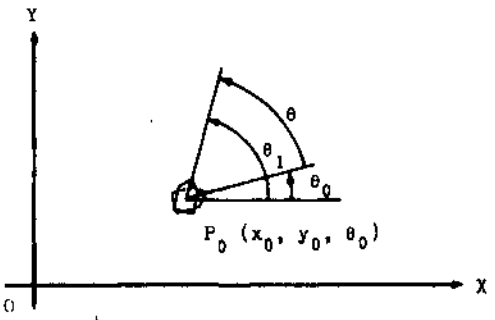

Fig. 4 Movement by a spin command

(4) Moving in the transient state

When a command is terminated and the next one is executed, there is a change of target speeds.

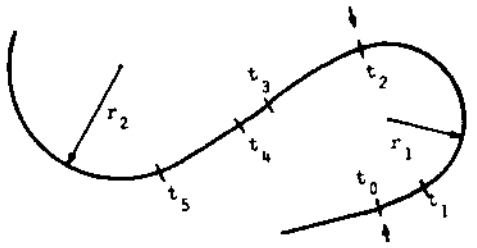

Fig. 5 Path with transient states

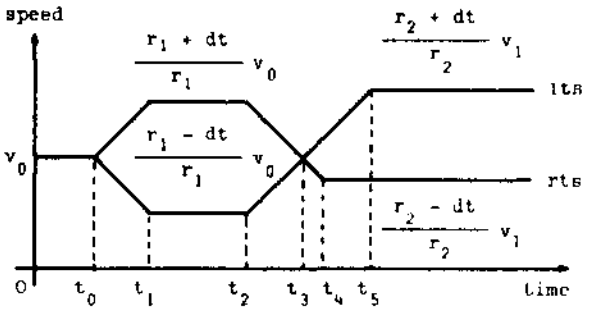

Fig. 6 Speed control in transient states

From that time, driving wheels approach the new target speed at a uniform acceleration. If the center velocity v is kept constant, the path is always a trochoid whose curvature changes linearly with respect to the travelling distance.

Suppose, for example, the following command sequence is executed:

gostraight in v_0 until C0;
turn with r_1 in v_0 until C1;
turn with $-r_2$ in v_1 until C2

where C0, C1 and C2 are terminating conditions.

The path for this command sequence is shown in Fig. 5. Fig. 6 shows the change of the speed rate for the sequence.

The robot has the following target speeds at the end of the execution of the first command:

$$lts = v_0, \qquad rts = v_0$$

At that time, the real speeds of the wheels are equal to the target speeds.

The second command is started at the time t_0 (an arrow ↑ in Fig. 5 denotes the beginning of a command). New target speeds are given to draw a circle of radius r_1 at speed v_0:

$$lts = \frac{r_1 - dt}{r_1} v_0, \qquad rts = \frac{r_1 + dt}{r_1} v_0$$

Each driving wheel approaches its new target speed at uniform acceleration. In this case, the transient path is a trochoid because the speed of the robot's center is not changed.

The speeds of both wheels reach the target

speeds at the time t_1; the leg is then in stationary state and CNS is true. The condition C1 is satisfied at the time t_2, the second command is terminated and the last one is started. The next target speeds are given as:

$$lts = \frac{r_2 + dt}{r_2} v_1 , \qquad rts = \frac{r_2 - dt}{r_2} v_1$$

Then the left wheel speed is reduced and the right wheel speed is increased.

(5) Terminating conditions

While the command is being executed, the leg subsystem examines the terminating condition. If the condition is satisfied then the command is terminated. If the condition has the format v = k, where v is a variable in the leg status specified by the <variable 1>, then the value of v, value(v), is compared with k. The condition is satisfied if value(v) is greater than or equal to k when value(v) is increasing, or if value(v) is less than or equal to k when the value(v) is decreasing.

The condition 'speed = vl' is satisfied if the speed of the robot's center is equal to the specified value in the case of gostraight and turn commands, or if the speed of the right wheel is equal to the specified value in the case of the spin command.

The condition STR is satisfied if the robot goes straight, which occurs when the two driving wheels have equal speed. The condition CVS is satisfied if both driving wheels have zero acceleration.

Some terminating conditions may remain unsatisfied indefinitly. The terminating condition can be omitted. If the command has no condition then it is not terminated.

2.2.2 Control Commands

The acceptance and the execution of control commands do not modify the state of the FIFO storage and the execution state of the motion command, except for terminate and cancel commands. The operations of the control commands are as follows:

(1) Get command

The value of the variable, specified by <variable 1>, of the leg status is returned as a reply. The returned value is the value the leg status has at that moment.

For example, 'get (SS)' returns the value of the odometer.

(2) Update command

The command writes the value of the variable of the leg status specified by <variable 2>.

For example, 'update (XX, 0)' sets XX to 0.

(3) Set and reset commands

The leg subsystem always monitors the travelling distance, the direction of the robot, and the other variables in the leg status. It has a mechanism which can generate signals for the brain at a given distance or direction interval. For example, signals can be generated each time the robot moves 10 cm. This feature is useful, for example, when the robot measures distances to the left side wall at equal intervals. The intervals in distance and direction are recorded in SSI and DD1, respectively. DDI and SSI are not included in the leg status because they need not be read by the brain.

When a set command is accepted, the leg subsystem begins to generate signals each time the robot moves the specified distance or direction.

The reset command terminates this generation of signals.

2.3 Examples of Command Sequences

These examples show the usage of the command sequence.

(1) Moving straight-1

The robot moves a distance ℓ. It starts from a stopped state, and stops at the destination.

gostraight in v until $\ell - \ell_d$;
gostraight in 0 until speed = 0

where ℓ_d is the braking distance or distance required to stop the robot when it is moving at a speed v, and it is known.

The path and the speed rate change are shown in Fig. 7.

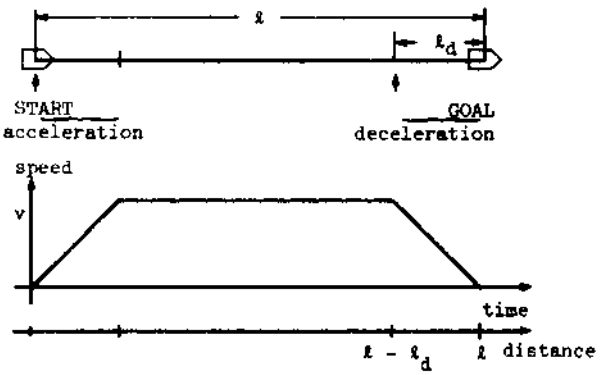

Fig. 7 Moving straight-1

(2) Moving straight-2

This is similar to (1), except that the robot slows to its minimum possible speed v_0 before it reaches the destination, then stops. This sequence is better than the sequence described in (1) for travelling an accurate distance. The speed rate is altered as shown in Fig. 8.

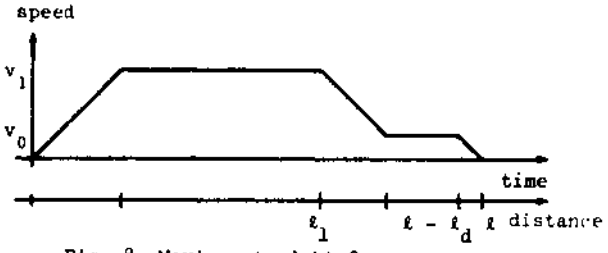

Fig. 8 Moving straight-2

```
gostraight in v1 until SSO = ℓ1;
gostraight in v0 until SSO = ℓ - ℓ1 - ℓd;
gostraight in 0  until speed = 0
```

(3) Changing the direction

The robot revolves through the angle θ (See Fig. 9)

```
spin in v2 until DDO = θ - θd;
spin in 0  until speed = 0
```

where θ_d is the angle required to stop the robot when it is revolving at speed v_2.

Using the sequences described above, the robot can move anywhere. This does not, however, assume smooth motion.

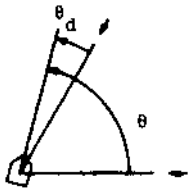

Fig. 9 Spin turn

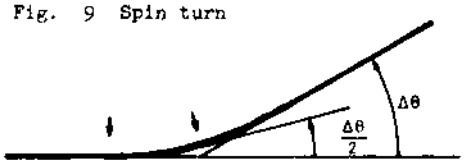

Fig. 10 Modification of the walking direction

(4) Modification of the walking direction

While the robot is moving, it modifies its walking direction by a small angle Δθ (See Fig. 10).

```
turn with r in v until DDO = Δθ/2;
gostraight in v until some terminating
condition
```

where the value of r can be any arbitrary small value.

(5) Changing lane

The robot changes from one lane to another (See Fig. 11).

```
turn with r in v until DDO = Δθ/2;
gostraight in v until STR;
turn with -r in v until DDO = -Δθ/2;
gostraight in v until STR
```

where r can take any small value, and Δθ is any small angle.

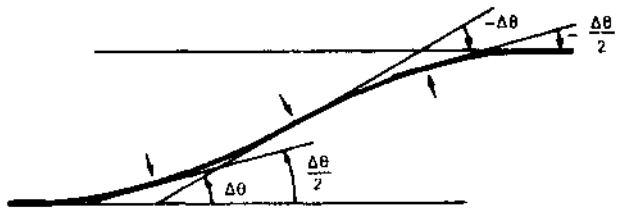

Fig. 11 Changing lanes

3. CONCLUDING REMARKS

We have presented a command system for controlling movement of a mobile robot which is steered and moved by controlling two driving wheels independently. Smooth motions can be achieved through using this command system, and the terminating condition is also useful.

We have been implementing the leg status and the command system on a mobile robot, Yamnbico 3.1. The implementation and results will be reported in subsequent papers.

ACKNOWLEDGEMENTS

The authors thank Professors J. W. Higgins of University of Tsukuba and T. Furugori of the University of Electro-Communications for their useful suggestions and critical reading of the manuscript.

REFERENCES

[1] Thompson, A. M. "The Navigation System of the JPL Robot" Proc. of 5th IJCAI, 1977, 749-757.

[2] Nilsson, N. J. "A Mobile Automation: An Application of Artificial Intelligence Techniques" Proc. of 1st IJCAI, 1969, 509-520.

[3] Coles, L. S. et.al. "Decision Analysis for an Experimental Robot with an Unreliable Sensors" Proc. of 4th IJCAI, 1975, 749-757.

[4] Giralt, G. et.al. "A Multi-Level Planning and Navigation System for a Mobile Robot" Proc. of 6th IJCAI, 1979, 335-337.

[5] Miller, J. A. "The Autonomous Guidance and Control of a Roving Robot" Proc. of 5th IJCAI, 1977, 759-760.

[6] Kanayama, Y. et.al. "A Mobile Robot with Sonic Sensors and its Understanding of a Simple World" Technical Report of Institute of Information Sciences and Electronics, University of Tsukuba, ISE-TF-81-22, Feb., 1981.