Autonomous Polyurethane Applier

Heejae Kim, Dae Yeon Jung, Eric Waller, Sahil Kohli
Polytechnic University, YES Summer Program 2004

Abstract

Polyurethane is a potentially hazardous material that emits nauseous fumes that can lead to lung irritation, eye irritation, mucus membrane irritation if exposed for a long enough time. The purpose of this paper is to shows one method of creating an autonomous Polyurethane applier using mechatronics, a combination of mechanical engineering, electrical engineering and robotics. Namely, the most important aspects of the robot to consider are the directing of Polyurethane, the moving of the robot, and the self-guiding mechanisms needed for autonomous function. While designs for the chassis can vary according to the desires of the creator; however, the movements of the robot should remain constant with our floor sweep and wall following motions. The two program process not only simplifies the programming, but also assures that most of the floor is covered, requiring at most for the homeowner to finish off the corners with a small brush. Some future modifications to our robot would be to regulate the flow of the liquid falling from the tank, use more dependable motors, and change the roller placement.

Introduction

The technology presented in the futuristic cartoon, the Jetson's, includes a panorama of technological fantasies that include flying cars, automatic hair-stylers, robots with artificial intelligence. Although farfetched, the Jetson's world may be around the corner as advances in Mechatronics, the combination of mechanical engineering and electronic controls, have made purposeful robots, especially autonomous ones a reality. For example, the company iRobot created Roomba, the autonomous vacuum cleaner, making a world of automatic household robots more realistic [1].

Though cleaning robots are popular these days, a Polyurethane applier proves to be a more worthwhile cause. Not only does this robot get rid of the tedium of applying the floor finish, but also protects the user from the harmful fumes one receives from Polyurethane. Polyurethane, a popular hard wood finish that gives wood floors the glossy look, emits potentially harmful fumes that can lead to lung, eye, nose and throat irritation [2]. If a robot instead takes the place of a human, contact with the fumes would obviously be minimized.

Moving, or rather the plans for the motions, starts with a zigzag pattern (fig.1) that covers most of the open space of the floor, followed by a wall-following program (fig.2) that allows the robot to cover up the missed edges. However, because of a roller attachment that resembles a trailer attached to a truck, the turns need to be wide and not miss any spots on the floor. Also, the 180 degree turns must be followed by a straightening algorithm in order to continue without any complications. Similarly, the 90 degree turns

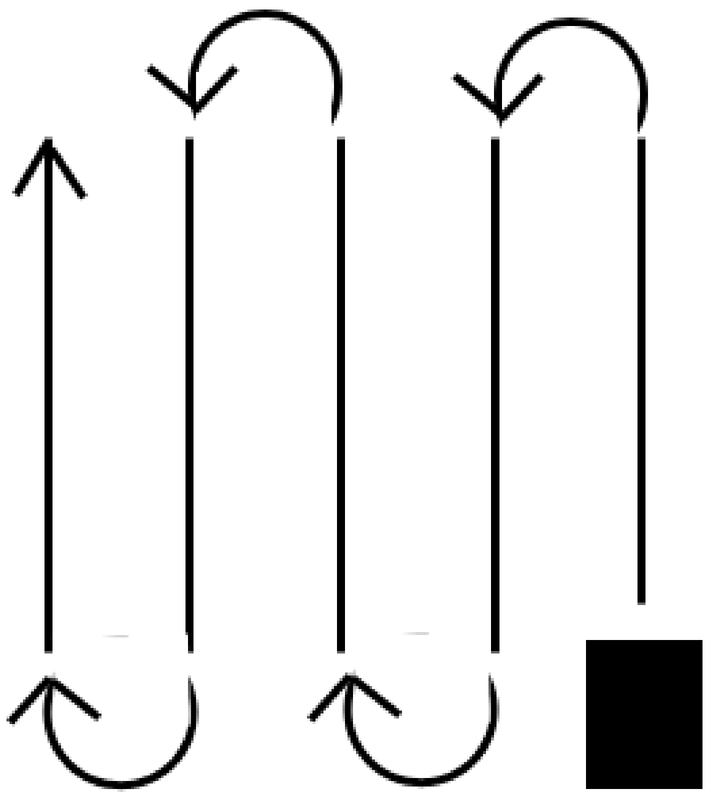


Fig.1 180 degree turns, Area cover program

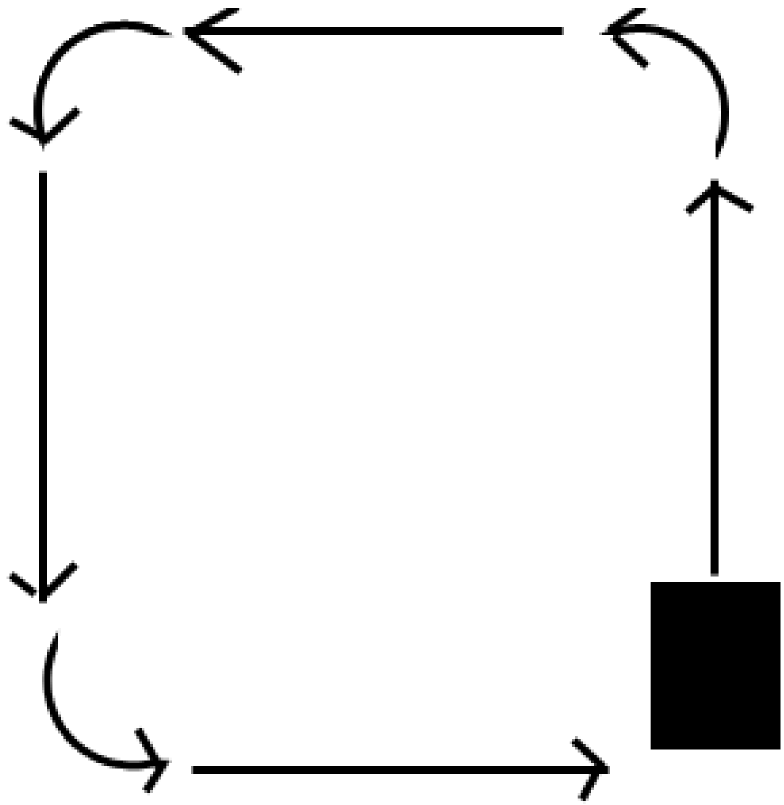


Fig.2 90 degree turns, Wall-follow program

of the wall following program need a straightening algorithm of their own while staying parallel to the walls.

For wall detection, four infrared sensors are used. On the zigzag program, only the front sensor is necessary, but the wall following program requires all four as direction control and backward collisions must be avoided. When IR LED emits an infrared frequency in the range of 40 kHz, the light bounces back; the detector will detect the light and give a signal to the microcontroller [3].

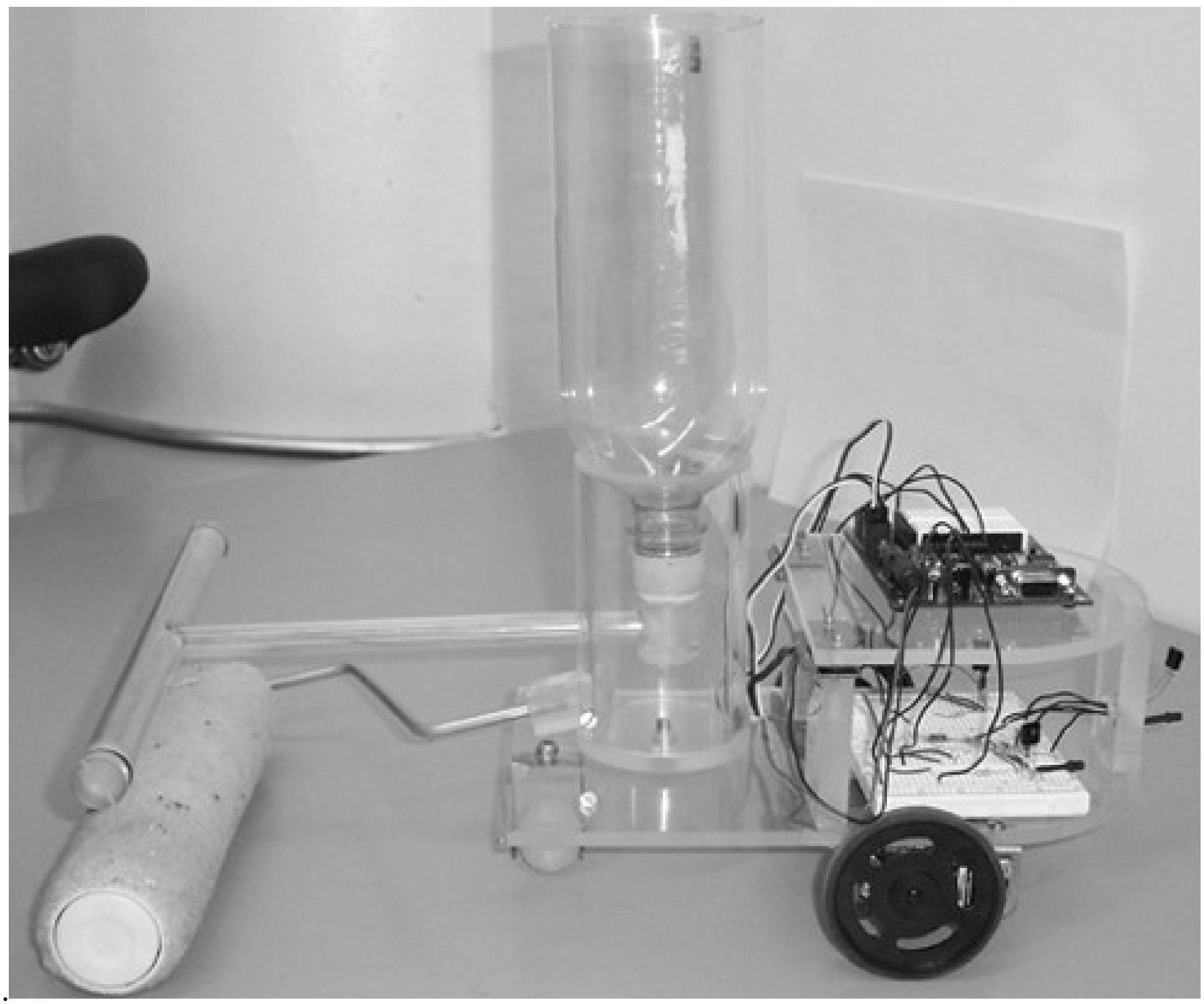
The objective then is to create an autonomous polyurethane applier that covers most of the floor with a liquid.

Methodology

The chassis was made with a circular base with platforms for extra components such as infrared sensors, the holding tank for the polyurethane, and the polyurethane dispenser attached to the back of the robot. The wheels were aligned similar to the boe-bot chassis made by Parallax Co, with two wheels attached to servo motors in the front and a ball type wheel in the back for support.

The programming of the robot involved using infrared sensors to detect obstacles and walls while moving the robot in a row by row pattern to ensure a good polish job. The challenge exists mainly in getting every square inch covered, because as the robot turns, it left a little wedge uncovered. So the robot was programmed to back up to cover the bit that was missed.

The robot was programmed to follow the signals of a universal TV remote programmed to Sony's code for televisions. The remote was programmed such that numbers two, four, six and eight directed the robot forward, left, right and back respectively, seven and nine switched between the two programs, and five stopped the robot



.

Fig.4 The robot

To test an autonomous robot, it would make sense to have obstacles; however, because the robot is a floor polisher, circumstances had to be considered: all movable furniture were removed from area prior to polishing even when people were manually applying the polyurethane. So basically the only obstacles needed to be avoided were the walls and maybe some occasional poles and steps. Because the test model was small, the area for testing was also reduced to a topless box-type arena with few obstacles. Since using actual polyurethane is expensive and hazardous, caro syrup, a liquid of similar viscosity, was used to mimic polyurethane.

Data / Program

```
'{$STAMP BS2}
'{$PBASIC 2.5}

ir_det_pin CON 8
pause_time CON 20
active_low CON 0

ir_detect VAR IN8

ir_pulse VAR Word
counter VAR Nib
type VAR Nib
pulse_delay_time CON 2

debounce_time CON 20
ir_message VAR Byte
active_high CON 1
decimal_value VAR Word

main:
IF ir_detect = 1 THEN main
GOSUB find_and_display_start_pulse
GOSUB check_for_stop_bit
GOSUB convert_to_binary_number_display
GOSUB program_select
GOTO main

find_and_display_start_pulse:

FOR counter = 0 TO 15
PULSIN ir_det_pin,active_low,ir_pulse(0)
NEXT

RETURN

check_for_stop_bit:
PULSIN ir_det_pin,active_high,ir_pulse(0)
IF ir_pulse(0) > 1400 AND ir_pulse(0) <> 0 THEN continue
GOTO check_for_stop_bit

continue:
PULSIN ir_det_pin,active_low,ir_pulse(0)
PULSIN ir_det_pin,active_low,ir_pulse(1)
PULSIN ir_det_pin,active_low,ir_pulse(2)
PULSIN ir_det_pin,active_low,ir_pulse(3)
PULSIN ir_det_pin,active_low,ir_pulse(4)
PULSIN ir_det_pin,active_low,ir_pulse(5)
PULSIN ir_det_pin,active_low,ir_pulse(6)
PULSIN ir_det_pin,active_low,ir_pulse(7)
PULSIN ir_det_pin,active_low,ir_pulse(8)
PULSIN ir_det_pin,active_low,ir_pulse(9)
PULSIN ir_det_pin,active_low,ir_pulse(10)
PULSIN ir_det_pin,active_low,ir_pulse(11)

RETURN

convert_to_binary_number_display:
FOR COunter = 0 TO 10
LOOKDOWN ir_pulse(counter), < [400,800],
ir_message.LOWBIT(counter)
```

```
NEXT

RETURN

program_select:

IF (ir_message = %00000100) THEN stop0
IF (ir_message = %00000001) THEN front0
IF (ir_message = %00000011) THEN left0
IF (ir_message = %00000101) THEN right0
IF (ir_message = %00000111) THEN back0
IF (ir_message = %00000110) THEN area_cover
IF (ir_message = %00001000) THEN wall

RETURN

stop0:
DO
IF ir_detect = 0 THEN main
LOOP

front0:
FOR x = 1 TO 5
PULSOUT 14, 649
PULSOUT 15, 892
PAUSE 20
NEXT
GOTO main

back0:
FOR x = 1 TO 5
PULSOUT 14, 850
PULSOUT 15, 650
PAUSE 20
NEXT
GOTO main

left0:
FOR x = 1 TO 3
PULSOUT 14, 650
PULSOUT 15, 725
PAUSE 20
NEXT
GOTO main

right0:
FOR x = 1 TO 3
PULSOUT 14, 775
PULSOUT 15, 850
PAUSE 20
NEXT
GOTO main

'------Variables------
irmid VAR Bit
irleft VAR Bit
irright VAR Bit
irback VAR Bit
x VAR Word
a VAR Word
v VAR Word
w VAR Word
n VAR Byte
m VAR Byte

'------Constants-------
freq CON 38500


Area_Cover:
IF ir_detect = 0 THEN main


'------IR Readings-----
FREQOUT 6, 1, freq
irmid = IN4
```

```
'------Maneuvers------

IF irmid = 0 THEN turn

PULSOUT 14, 649
PULSOUT 15, 892
PAUSE 20

GOTO Area_Cover

turn:
x = x + 1
a = (x / 2) * 2
IF (x = a) THEN left
IF (x <> a) THEN right

right:
FOR n = 1 TO 220
PULSOUT 14, 743
PULSOUT 15, 850
PAUSE 20
NEXT

GOTO Area_Cover

left:

FOR x = 1 TO 209
PULSOUT 14, 650
PULSOUT 15, 757
PAUSE 20
NEXT

FOR x = 1 TO 65
PULSOUT 14, 649
PULSOUT 15, 892
PAUSE 20
NEXT

FOR x = 1 TO 65
PULSOUT 14, 725
PULSOUT 15, 892
PAUSE 20
NEXT

GOTO Area_Cover

'--------------------------------------------

wall:
IF ir_detect = 0 THEN main
FREQOUT 6, 1, freq '38500
IF IN4 = 0 THEN turn1

forward1:
PULSOUT 14, 649
PULSOUT 15, 892
PAUSE 20

GOTO wall

turn1:
FREQOUT 2, 1, freq '38500
irright = IN0

FREQOUT 10, 1, freq '38500
irleft = IN9

IF irright = 0 THEN left1

IF irleft = 0 THEN right1


GOTO wall
```

```
'--------Maneuvers------------
right1:
FOR m = 1 TO 140
PULSOUT 14, 743
PULSOUT 15, 850
PAUSE 20
NEXT

'--------straighten roller-----
FOR m = 1 TO 85
PULSOUT 14, 649
PULSOUT 15, 892
PAUSE 20
NEXT

FOR m = 1 TO 65
PULSOUT 14, 649
PULSOUT 15, 765
PAUSE 20
NEXT
GOTO back1

left1:
FOR m = 1 TO 150
PULSOUT 14, 650
PULSOUT 15, 757
PAUSE 20
NEXT

FOR m = 1 TO 70
PULSOUT 14, 649
PULSOUT 15, 892
PAUSE 20
NEXT

FOR m = 1 TO 65
PULSOUT 14, 735
PULSOUT 15, 892
PAUSE 20
NEXT
GOTO back1

back1:
PULSOUT 14, 850
PULSOUT 15, 650
PAUSE 20
FREQOUT 7, 1, 38500
IF 0 = IN5 THEN forward1
GOTO back1
```

Discussion

Despite the seemingly simplistic needs of the robot, the design and the parts require a lot more attention. First the tank had been placed in the front to offset the weight provided by the roller; however, this proved to be dangerous as water would be traveling above the circuitry. Therefore, all of the circuitry was placed in the front on two tiers: one for the breadboards and one for the microcontroller. The tank was also connected to a solid instead of a flexible pipe that rotates on the same center point as the roller. Unfortunately, the roller attachment had to be slightly off-center in order to line up the roller with the robot. This disrupted the symmetry between the two turns, requiring different adjustments.

Another challenge was getting the robot to turn precisely 180 degrees and 90 degrees and go straight afterwards. Not only were the servo motors that were being used temperamental and refused to go straight in the first place, but also the roller attachment ultimately dragged and forced the robot to drift to the side of the roller. Hence an S-turn was applied to the program to straighten the roller by dragging it further into a turn [fig.3]. However, several design flaws prevented constant results from occurring; one was the rotating back wheel, which created unpredictable turns; another was the offset roller position, which created uneven drags on the two sides.

In the future the several design flaws discussed previously will need to be fixed in order to refine the turning. Also, the servo motors must be replaced with dc motors because more power may be needed to pull the roller along with the liquid. A centered roller attachment along with the back corners cut to prevent the roller from getting caught on the corners is necessary. Also the rate at which the polyurethane is falling must be regulate using

Conclusion

The robot, minus the inconsistencies in turning, successfully follows both programs, can be remotely controlled and most important of all, autonomously applies liquid and spreads it with a roller. While the Plexiglas chassis was a good idea, the rest of the components need fine tuning in terms of alignment.

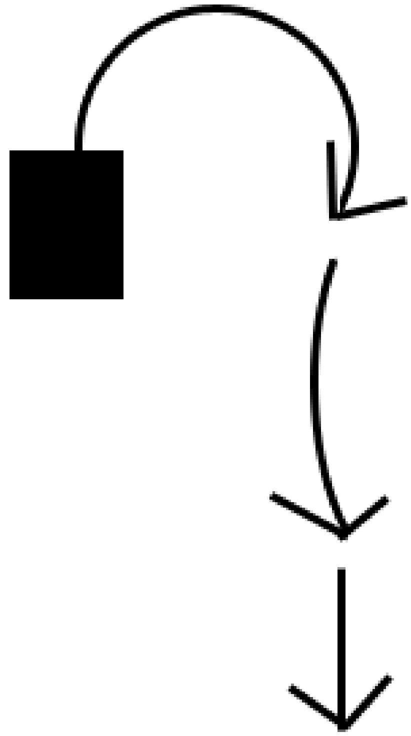


Fig.3 S-turn for 180 degrees

References

1. "Roomba," in *iRobot*, [Online]. Available WWW: http://irobot.com/consumer.
2. "Polyurethane Paint," *New York State Department of Transportation,* Nov. 1999, SB-99-4
3. B. Gopalakrishnan, S. Tirunellayi, and R. Todkar. "Design and development of an autonomous mobile smart vehicle: a mechatronics application," *Mechatronics*, vol. 14, 2004, pp. 491-514