micromachines

Article

Autonomous Spiral Motion by a Small-Type Robot on an Obstacle-Available Surface

Shinya Tokunaga [1], Chinthaka Premachandra [2,*], H. Waruna H. Premachandra [3], Hiroharu Kawanaka [4], Sagara Sumathipala [5] and B. S. Sudantha [5]

1 Department of Electronic Engineering, School of Engineering, Shibaura Institute of Technology, Tokyo 135-8548, Japan; AG14074@shibaura-it.ac.jp
2 Department of Electronic Engineering, School of Engineering/Graduate School of Engineering and Science, Shibaura Institute of Technology, Tokyo 135-8548, Japan
3 ICT Center, Wayamba University of Sri Lanka, Gonawila 60170, Sri Lanka; warunaprema@yahoo.com
4 Department of Electrical and Electronic Engineering, Graduate School of Engineering, Mie University, Mie 514-8507, Japan; kawanaka@elec.mie-u.ac.jp
5 Faculty of Information Technology, University of Moratwua, Moratuwa 10400, Sri Lanka; sagarasns@gmail.com (S.S.); bh.sudantha@gmail.com (B.S.S.)
* Correspondence: chintaka@sic.shibaura-it.ac.jp

Abstract: Several robot-related studies have been conducted in recent years; however, studies on the autonomous travel of small mobile robots in small spaces are lacking. In this study, we investigate the development of autonomous travel for small robots that need to travel and cover the entire smooth surface, such as those employed for cleaning tables or solar panels. We consider an obstacle-available surface and target this travel on it by proposing a spiral motion method. To achieve the spiral motion, we focus on developing autonomous avoidance of obstacles, return to original path, and fall prevention when robots traverse a surface. The development of regular travel by a robot without an encoder is an important feature of this study. The traveled distance was measured using the traveling time. We achieved spiral motion by analyzing the data from multiple small sensors installed on the robot by introducing a new attitude-control method, and we ensured that the robot returned to the original spiral path autonomously after avoiding obstacles and without falling over the edge of the surface.

Keywords: spiral robot motion; fall prevention; obstacle avoidance; sensor fusion; motion analysis

check for updates

Citation: Tokunaga, S.; Premachandra, C.; Premachandra, H.W.H.; Kawanaka, H.; Sumathipala, S.; Sudantha, B.S. Autonomous Spiral Motion by a Small-Type Robot on an Obstacle-Available Surface. *Micromachines* **2021**, *12*, 375. https://doi.org/10.3390/mi12040375

Academic Editor: Duc Truong Pham

Received: 2 March 2021
Accepted: 26 March 2021
Published: 1 April 2021

1. Introduction

With the evolution of computers and related technologies, several studies have been conducted on robots that can perform tasks deemed too risky or difficult for humans. To that end, the focus has been on developing wheeled robots [1–9], flying robots [10–18], and snakelike robots [19–26]. Many recent studies focus on developing not only robots that perform risky or difficult tasks for humans, but also production robots in workplaces that face labor shortages [27–30], cleaning robots to support housework to meet the demands of our busy lives [31–35], and robots to support the medical sector. Besides fixed robots (e.g., arm robots), most robots move and perform tasks in a wide range of environments; if robots have a wide range of movement, their large size does not constitute a problem. For example, the currently available home-cleaning robots are of a certain size, and because they are equipped with several high-performance functions, they can efficiently perform cleaning tasks, including smooth travel over smooth surfaces. In cleaning robots, path planning is very important. There are many studies in the literature regarding the path planning and autonomous movement of robots in comparatively larger indoor and outdoor areas [36–39]. Some of them have been targeted to achieve smooth movement within obstacle-available environments [40–42]. We experimentally found that applying these methods to small-type robots to achieve certain movement on a space like a table or a solar panel is difficult, since the computational capacity of a small-type robot is limited. In this paper, we mainly address this problem by making an appropriate problem setting and studying the problem with a small-type robot as mentioned below.

The motivation of this work is to achieve spiral travel of a small-type mobile robot to touch the maximum area of an obstacle-available surface (like a table), only with on-board lightweight hardware. Here, we investigate smooth travel for robots that can perform tasks such as cleaning tables or detecting cracks in solar panels. If the area of activity of the robot is small, the robot itself should be small. Therefore, in this study, we consider robots with a 10 cm $\times$ 10 cm footprint as small robots. We study regular travel on surfaces in small spaces as the basic technology for such robots to perform tasks such as cleaning tables or solar panels, and detecting defects in solar panels.

Further, we study robot operation under the scenario in which an obstacle is present during travel, which the robot should avoid, and then return to its regular travel path autonomously. Figure 1 shows the specific movement that is considered in this study. In particular, this movement is a spiral movement that combines straight motion and U-turns. We believe that this movement will enable regular travel while covering most of the smooth surface, thereby allowing efficient performance in cleaning tasks.

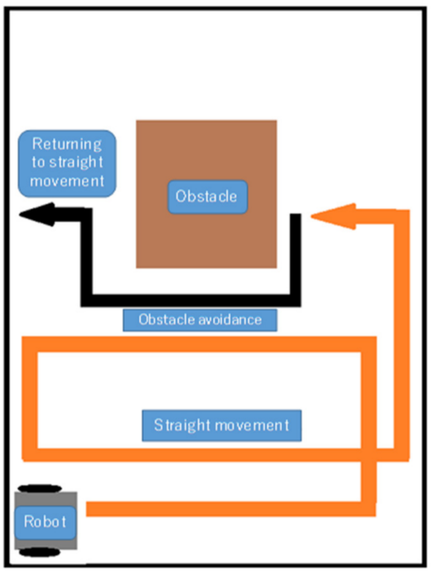


Figure 1. Robot movement to touch the majority of the moving surface.

There are previous studies that investigate regular travel [43–46]; however, they focus on simulations and not actual robot operations. The distance traveled by a mobile robot is conventionally measured using an embedded encoder [6,7]. The robot used in this study is not equipped with an encoder; we equipped a compact crawler robot with small hardware. The traveled distance was measured using the traveling time. To develop a small-type robot for which compactness is required, this kind of travel-time-based distance calculation can be applied. The development of regular travel, as shown in Figure 1, by a robot without an encoder is an important feature of this study. In this paper, we introduce a new attitude-control method. In this method, we obtain Euler angles from the onboard magnetic-field and acceleration sensors, and then convert them into quaternions and perform attitude control by updating the quaternions obtained from the onboard gyroscope sensor values. This is also one of the major contributions of this work. We achieve regular travel as shown in Figure 1, by analyzing the data from multiple small sensors installed on the robot, and we ensure that the robot returns to the original path autonomously after avoiding obstacles and without falling over the surface. This is a unique motion generation of this study.

We manufactured the robot described above and achieved the regular travel and fall prevention for this robot. All the processing that is necessary to achieve desirable autonomous robot motion is implemented by an onboard compact single-board microcontroller.

The remainder of this manuscript is structured as follows. Section 2 describes the hardware structure of the robot developed to carry out this work. Section 3 presents the proposed robot's attitude-control method, and Section 4 describes method for fall prevention and obstacle avoidance. Section 5 presents and discusses the experimental verification of this work in detail. Finally, Section 6 concludes the paper.

2. Hardware of Developed Robot

2.1. Hardware Installed on the Robot

The small robot designed in this study was created by fitting a commercially available Zumo robot—a crawler robot with an attitude and heading reference system (AHRS)—with new sensors, a single-board microcontroller, and additional small sensors. As shown in Figure 2, the robot was equipped with two additional ultrasonic sensors (front and side) and three distance sensors (GP2Y0A21YK; two on the front and one on the back). The system processes and robot attitude control were performed using the installed single-board microcontroller (Arduino UNO).

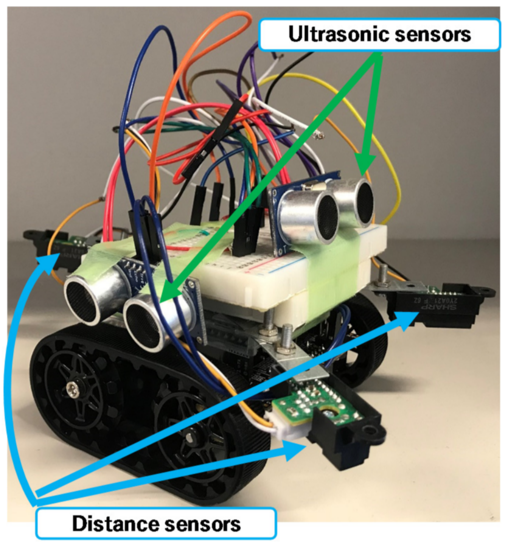


Figure 2. The robot developed to carry out this work.

2.2. Hardware Roles

Table 1 lists the installed hardware and the original hardware of the Zumo robot, including their functions.

Table 1. Hardware roles.

Hardware	Role
AHRS	Robot attitude control
Ultrasonic sensor $\times$ 2	Front and side obstacle detection
Distance sensors $\times$ 3	Fall prevention; 2 on the front, 1 on the back
Arduino UNO	System processes and robot control

The AHRS provides *z*-axis orientation output with the embedded gyroscope, acceleration, and magnetic field sensors. The main processing flow of achieving autonomous travel of the robot is shown in Figure 3. At the beginning, the robot starts traveling with a straight movement. If the robot finds the obstacle, it stops and begins the obstacle avoidance. The obstacle avoidance is done based on the outputs from the two onboard ultrasonic sensors. After avoiding the obstacle, the robot returns to the original movement and continues traveling. When the robot comes to the edge of the traveling space (table), falling prevention is conducted based on the outputs from the distance sensors. The falling situation is determined by using a predefined threshold value. After the falling prevention, the robot makes a U-turn and original spiral travel is conducted. During the U-turn and other movements, robot attitude control is conducted using the AHRS, as detailed in the next section.

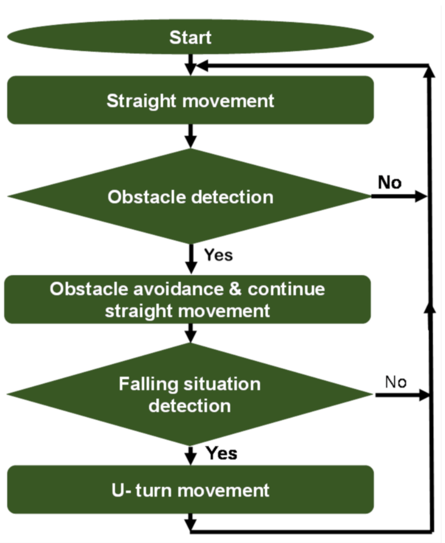

Start
Straight movement
Obstacle detection
No
Yes
Obstacle avoidance & continue straight movement
Falling situation detection
No
Yes
U- turn movement

Figure 3. Flowchart of the robot's movement generation.

3. Robot Attitude Control

In this study, regular travel refers to spiral motion. To ensure that the robot moved such that it covered the entire surface, we established a regular travel route where the traveled path is duplicated (Figure 1). Obstacle avoidance and robot attitude control were necessary to prevent the robot from falling off the table during motion.

3.1. Control Using AHRS

The rotational yaw angle, which uses the Z axis of the robot (vertical axis of the object) as the rotational axis, is the parameter used for control; the attitude of the robot is controlled using AHRS. Angular velocity is detected through the gyroscope sensor, and the angle is obtained via integration (Figure 4).

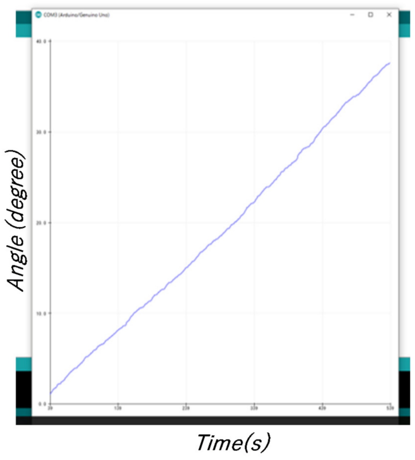


Figure 4. Angle detection only using a gyroscope sensor (when the robot was stationary).

The limitation of using only a gyroscope sensor to obtain the angle is that an angle is obtained even if there is no robot rotation (drift phenomenon). Therefore, it is difficult to achieve attitude control using only gyroscope sensors, and thus, we studied attitude control with AHRS.

$$\theta = \theta_{t-1} + \omega \cdot dt \tag{1}$$

where θ is the Euler angle, θ_{t-1} is the previous Euler angle, ω is the angular velocity, and dt is the integral time.

The angles expressed in Equation (1) are Euler angles that denote the respective degrees of rotation in relation to the three axes. In the Euler angle representation, results vary based on the order of rotation. Therefore, we used quaternions to obtain the angles easily. We obtained Euler angles from the magnetic field and acceleration sensors, and then converted them into quaternions and performed attitude control by updating the quaternions obtained from the gyroscope sensor values [47]. These steps are outlined below.

First, we describe quaternions. Quaternions can be used to represent arbitrary angles to axes with arbitrary orientations. Figure 5 shows a quaternion representation. Since a vector designated as an axis must be normalized, an adjustment is performed. Rotation of angle A in relation to arbitrary orientation B in a 3D space can be represented as summarized below.

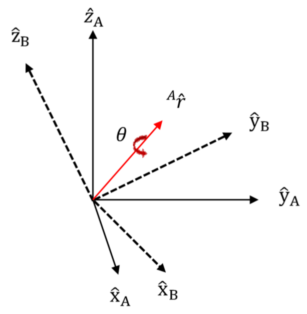


Figure 5. Quaternion representation.

In Figure 5, $\hat{x}_A, \hat{y}_A, and\ \hat{z}_A$ are normalized coordinate axes in relation to arbitrary orientation A; $\hat{x}_B, \hat{y}_B, and\ \hat{z}_B$ are normalized coordinate axes in relation to arbitrary orientation B; ${}^A\hat{r}$ is the vector in relation to arbitrary orientation A; and θ is the angle. The unit matrix for A in relation to orientation B, ${}^A_B\hat{q}$, is shown in Equation (2). Here, r_x, r_y, r_z are the vectors for each element.

$${}^A_B\hat{\boldsymbol{q}} = [q_1\ q_2\ q_3\ q_4] = [\cos\frac{\theta}{2} - r_x \sin\frac{\theta}{2} - r_y \sin\frac{\theta}{2} - r_z \sin\frac{\theta}{2}] \tag{2}$$

The values obtained from AHRS were Euler angles. Quaternions require fewer computations than Euler angles, and therefore, we converted them into quaternions for the computations and converted them back into Euler angles later. For the conversions, we used Equations (3)–(5).

$$\boldsymbol{q} = \begin{bmatrix} cos\left(\frac{\varphi}{2}\right)cos\left(\frac{\theta}{2}\right)cos\left(\frac{\phi}{2}\right) + sin\left(\frac{\phi}{2}\right)sin\left(\frac{\theta}{2}\right)sin\left(\frac{\varphi}{2}\right) \\ sin\left(\frac{\varphi}{2}\right)cos\left(\frac{\theta}{2}\right)cos\left(\frac{\phi}{2}\right) - cos\left(\frac{\phi}{2}\right)sin\left(\frac{\theta}{2}\right)sin\left(\frac{\varphi}{2}\right) \\ cos\left(\frac{\varphi}{2}\right)sin\left(\frac{\theta}{2}\right)cos\left(\frac{\phi}{2}\right) + sin\left(\frac{\phi}{2}\right)cos\left(\frac{\theta}{2}\right)sin\left(\frac{\varphi}{2}\right) \\ cos\left(\frac{\varphi}{2}\right)cos\left(\frac{\theta}{2}\right)sin\left(\frac{\phi}{2}\right) - sin\left(\frac{\phi}{2}\right)sin\left(\frac{\theta}{2}\right)cos\left(\frac{\varphi}{2}\right) \end{bmatrix} \tag{3}$$

where θ, ϕ, φ are the angles around axes $x, y, and\ z$, respectively.

$$ {}^{A}_{B}\boldsymbol{R} = \begin{bmatrix} 2q_1^2 - 1 + 2q_2^2 & 2(q_2q_3 + q_1q_4) & 2(q_2q_4 - q_1q_3) \\ 2(q_2q_3 - q_1q_4) & 2q_1^2 - 1 + 2q_3^2 & 2(q_3q_4 + q_1q_2) \\ 2(q_2q_4 + q_1q_3) & 2(q_3q_4 - q_1q_2) & 2q_1^2 - 1 + 2q_4^2 \end{bmatrix}, \quad (4) $$

where q_1, q_2, q_3, q_4 are quaternion elements and ${}^{A}_{B}\boldsymbol{R}$ is the Euler angle of A in relation to orientation B.

To determine the degree of rotation in relation to the arbitrary axes, we needed to update the rotation in relation to the arbitrary axis and the three axes based on the gyroscope sensor values. The quaternions were updated based on gyroscope sensor values given by Equation (5).

$$ \mathbf{q} = \begin{bmatrix} \mathrm{q}_1 \\ \mathrm{q}_2 \\ \mathrm{q}_3 \\ \mathrm{q}_4 \end{bmatrix} = 0.5 \times \mathrm{dt} \begin{bmatrix} -(\omega_\mathrm{x} \times \mathrm{q_b} + \omega_\mathrm{y} \times \mathrm{q_c} + \omega_\mathrm{z} \times \mathrm{q_4}) \\ \omega_\mathrm{x} \times \mathrm{q_a} - \omega_\mathrm{y} \times \mathrm{q_4} + \omega_\mathrm{z} \times \mathrm{q_c} \\ \omega_\mathrm{x} \times \mathrm{q_4} + \omega_\mathrm{y} \times \mathrm{q_a} - \omega_\mathrm{z} \times \mathrm{q_b} \\ \omega_\mathrm{y} \times \mathrm{q_b} - \omega_\mathrm{x} \times \mathrm{q_c} + \omega_\mathrm{z} \times \mathrm{q_a} \end{bmatrix} \quad (5) $$

where $q_a, q_b, and\ q_c$ are the pre-updates $q_1, q_2, and\ q_3$, and $\omega_x, \omega_y\ and\ \omega_z$ denote the angular velocity for each element.

3.2. Angle Detection by Each Sensor

The angles need to be calculated from the AHRS values. We studied the subsequent estimation of the current angles based on these respective angles. The AHRS values were used after offset correction and averaging.

(1) Gyroscope sensor

The gyroscope sensor measures the angular velocity around an axis. The sensor parameters are given by Equation (6), and angle vector is determined based on Equations (7) and (8).

$$ {}^{\mathrm{s}}\boldsymbol{\omega} = \begin{bmatrix} 0 & \omega_\mathrm{x} & \omega_\mathrm{y} & \omega_\mathrm{z} \end{bmatrix} \quad (6) $$

where $\omega_x, \omega_y, and\ \omega_z$ denote the angular velocity for each element.

$$ {}^{\mathrm{S}}_{\mathrm{E}}\dot{\mathbf{q}}_{\omega,\mathrm{t}} = \frac{1}{2}{}^{\mathrm{S}}_{\mathrm{E}}\hat{\mathbf{q}}_{\mathrm{est,t-1}} \otimes {}^{\mathrm{s}}\boldsymbol{\omega}_\mathbf{t} \quad (7) $$

where ${}^{S}_{E}\hat{\boldsymbol{q}}_{est,t-1}$ is the previously normalized angle, $\otimes$ is the cross product, and ${}^{s}\boldsymbol{\omega}_t$ is the angular-velocity vector.

$$ {}^{\mathrm{S}}_{\mathrm{E}}\mathbf{q}_{\omega,\mathrm{t}} = {}^{\mathbf{S}}_{\mathbf{E}}\hat{\mathbf{q}}_{\mathrm{est,t-1}} + {}^{\mathrm{S}}_{\mathrm{E}}\dot{\mathbf{q}}_{\omega,\mathrm{t}}\Delta\mathrm{t} \quad (8) $$

where ${}^{S}_{E}\boldsymbol{q}_{\omega,t}$ is the angle vector.

As shown in Equations (7) and (8), the derivative for the quaternion is given by ${}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t}$, and the current rotation angle is estimated through integration.

(2) Acceleration sensor

As there is no change in the acceleration angle, it is difficult to calculate the yaw angle. Further, if the sensor accelerates, it cannot be calculated. Each detected acceleration value must meet the conditions provided in Equation (9):

$$ \sqrt{\mathrm{a_x}^2 + \mathrm{a_y}^2 + \mathrm{a_z}^2} = 1\mathrm{G} = 9.8 \quad (9) $$

where a_x, a_y, and a_z denote the acceleration of each axis. When this condition is met, the rotation pitch around axis x and the rotation roll around axis y can be expressed by Equations (10) and (11):

$$ \mathrm{pitch} = \tan^{-1}\frac{\mathrm{a_x}}{\sqrt{\mathrm{a_y}^2 + \mathrm{a_z}^2}} \quad (10) $$

$$\mathrm{roll} = \tan^{-1}\frac{\mathrm{a_y}}{\mathrm{a_z}} \tag{11}$$

(3) Magnetic field sensor

If the roll and pitch information is known, the correct orientation can be estimated by the magnetic field sensor using these attitude data. When the magnetic field sensor output is $m_x, m_y,\ m_z$ and the roll and pitch angles obtained through Equations (10) and (11) are $\phi\ and\ \varphi$, respectively, the obtainable yaw angle θ is as shown by Equation (12):

$$\theta = \tan^{-1}\frac{\mathrm{m_z}\sin\phi - \mathrm{m_y}\cos\phi}{\mathrm{m_x}\cos\varphi + \mathrm{m_y}\sin\varphi\sin\phi + \mathrm{m_z}\sin\varphi\cos\phi} \tag{12}$$

3.3. Estimation of Current Angle

The current angle is estimated from the angles obtained from each sensor. The methods to estimate the current angles when there is a slight change in the acceleration and a substantial change in the acceleration is shown below.

(1) Slight change in acceleration

The quaternion is updated based on the angular velocity obtained using the gyroscope sensor, where the amount of change is based on the angles obtained from the acceleration and magnetic field sensors (Equations (10)–(12)).

(2) Substantial change in acceleration

If there is change in relation to the initial acceleration vector after obtaining the angles from the acceleration and magnetic field sensors (Equations (10)–(12)), the displacement is corrected using the Madgwick filter [46] following Equation (13).

$${}^{S}_{E}\boldsymbol{q}_{est,t} = {}^{S}_{E}\hat{\boldsymbol{q}}_{est,t-1} + {}^{S}_{E}\dot{\boldsymbol{q}}_{est,t}\Delta t \tag{13}$$

where ${}^{S}_{E}\dot{\boldsymbol{q}}_{est,t}$ is the rate of change of orientation, ${}^{S}_{E}\boldsymbol{q}_{est,t}$ is the estimated angle, ${}^{S}_{E}\hat{\boldsymbol{q}}_{est,t-1}$ is the previous normalized angle, and Δt is the change over time. The correction method uses the gradient method based on the acceleration sensor vector. Thus, we can achieve attitude control for the traveling robot.

4. Fall Prevention and Obstacle Avoidance

Fall prevention is an important part of ensuring the robot's regular travel on a flat surfaces such as a table or solar panel. Further, obstacle avoidance and the subsequent return to the regular travel path is important. For obstacle avoidance, the aim is to avoid an obstacle of a certain size such as a cardboard box, and subsequently, to return to the regular travel path.

4.1. Fall Prevention

Fall prevention was achieved using the three distance sensors installed on the robot. Only one sensor was installed on the back of the robot because backward movement is performed by the robot in only a few limited scenarios. A threshold was set for the distance sensors to distinguish between the table (ON) and the other objects (OFF). When both sensors were switched to ON, the robot was stopped. Figure 6 shows the ON and OFF statuses of the sensors when the robot was stopped; these thresholds were determined experimentally for this study. After the robot temporarily stopped during its regular travel (Figure 1), it moved backward, made a U-turn, and continued its operation. On the other hand, if the one of two sensor pair was ON, the robot stopped the movement, and turning behavior was not done.

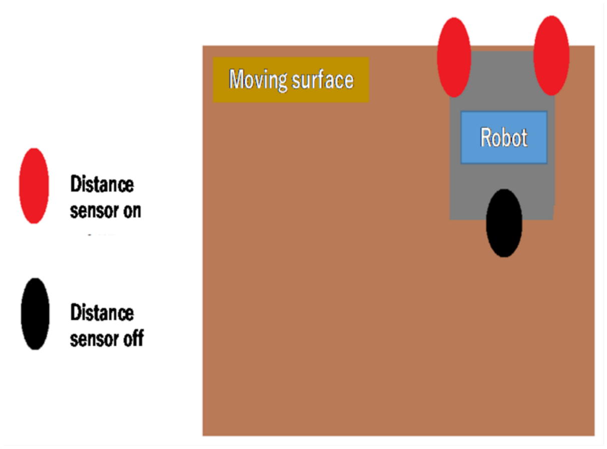


Figure 6. Distance sensor ON and OFF.

4.2. Obstacle Avoidance and Subsequent Return to Regular Travel

We studied the autonomous obstacle detection of an object of certain size by a robot, its avoidance, and the subsequent return of the robot to its regular travel path. This operation was performed using ultrasonic sensors installed on the front and side, in the sequence shown in Figure 7. The steps are listed below.

1. The front sonic ultrasonic sensor detects an obstacle.
2. The robot performs a 90° left turn, switches on the ultrasonic sensor on the right side, and moves straight ahead until the right-hand ultrasonic sensor exceeds the threshold. Then, the time of the rectilinear movement is obtained, and using the acceleration sensor value, the current coordinates of the robot are calculated.
3. The robot performs a 90° right turn and moves straight ahead until the threshold of the right-hand ultrasonic sensor is exceeded. The time of the rectilinear movement is then obtained and the current coordinates of the robot are calculated following them. In the straight movement, the travel distance is calculated using the velocity × travel time, since in this study, the robot velocity is constant.
4. The robot performs a 90° right turn and moves straight ahead until its x coordinate is 0.5. The robot performs a 90° left turn and returns to its original regular travel. The coordinate calculations of the robot at this time are given by Equations (14) and (15):

$$x = \frac{(v \cdot time)}{1000} \cdot cos\left(\frac{(pi \cdot degree)}{180}\right) \tag{14}$$

$$y = \frac{(v \cdot time)}{1000} \cdot sin\left(\frac{(pi \cdot degree)}{180}\right) \tag{15}$$

where v is robot velocity, time is the time of rectilinear movement, *pi* is the ratio of the circumference of the circle to its diameter, and degree is the rotation angle. These coordinates need to be adjusted in line with the actions of the robot; this is done only when required.

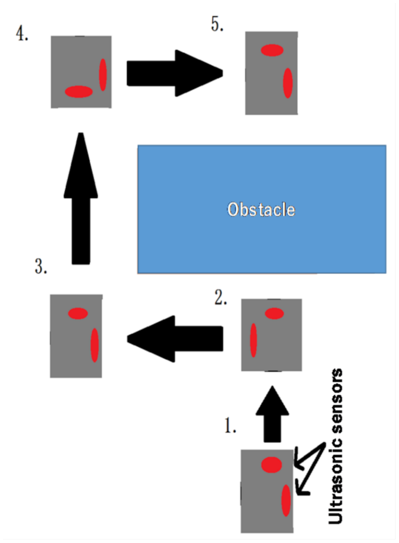


Figure 7. Returning to regular travel after avoiding obstacle.

5. Experimental Verification

We conducted verification experiments for the proposed methods using an actual robot.

5.1. Experimental Environment

We tested the individual proposed methods and verified the system as a whole. Attitude control was verified by allowing the robot to travel on a table; we did this by making the robot perform a 90° rotation and checking its attitude-control performance.

For fall prevention, we made the robot move on a table and, as shown in Figure 8, checked that the robot stopped when reaching the edge of the table.

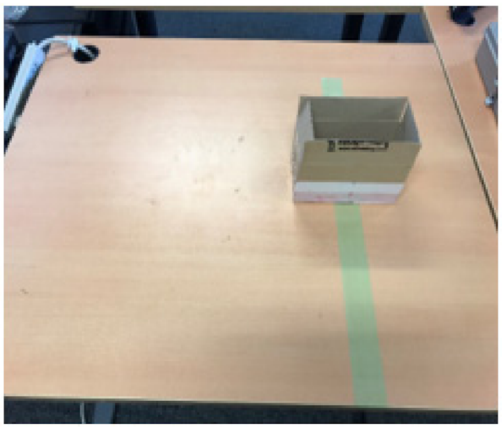


Figure 8. Example of a driving environment in the experiment.

Obstacle avoidance and the subsequent return to the original travel path were verified by placing an obstacle on the table. The environment is shown in Figure 7. We verified whether the designed robot detected the obstacle and returned to its original travel path while avoiding the obstacle. Further, we verified the accuracy by placing the robot on a green line of table and checking whether the robot had returned to its location on its original travel path after avoiding the obstacle. In this work, the on-robot microcontroller (Arduino UNO) implemented all the robot motions.

5.2. Experimental Results

(1) Attitude control—verification results

The attitude control of the robot was checked using AHRS by verifying its accuracy when performing a 90° rotation. The operation involved repeated 90° rotations. Figure 9 shows rotations between 100 ms and 400 ms, and between 500 ms and 900 ms. Figure 10 shows the robot pre- and postrotation.

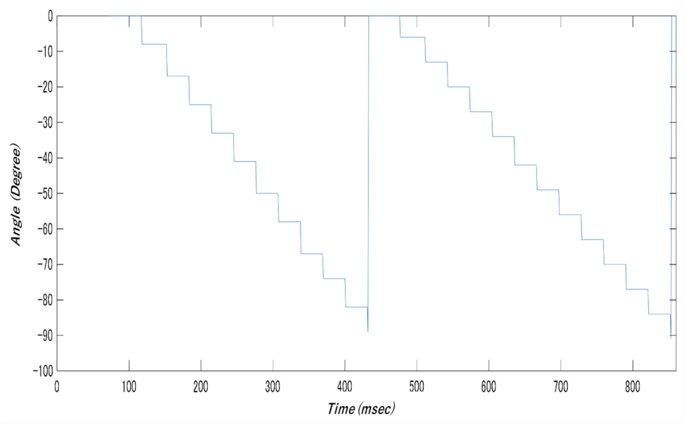


Figure 9. Attitude control test results (left-hand rotation).

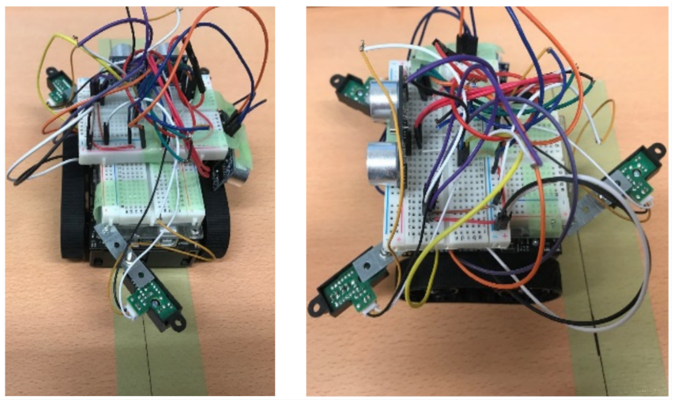


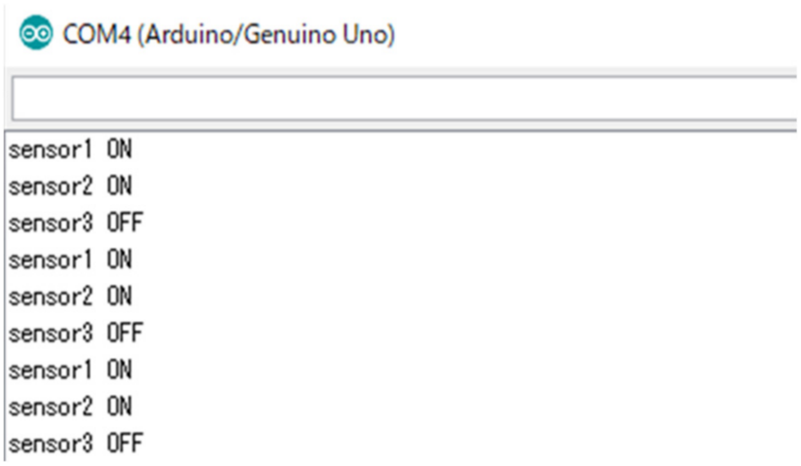
Figure 10. Robot prerotation (**left**) and postrotation (**right**).

We performed multiple experiments in the same way, and the results of the attitude control performance checks were the same as in these experiments. There were few errors over multiple robot-rotation operations; however, sometimes errors gradually accumulated during travel.

(2) Fall prevention method—verification results

Since we focused on preventing the robot from falling forward, the results are for operation using the two front sensors. If sensors were placed above the table, they were considered OFF, and if not, they were considered ON. Figures 11 and 12 show the actual performance results and robot position at the time. The experiment for this fall prevention technique was conducted 100 times, and it had a success rate of 99%, which is a good result. The failure case occurred due to a sudden slip of the robot.

(3) Obstacle avoidance and subsequent return to the original path—experiment


COM4 (Arduino/Genuino Uno)

```
sensor1 ON
sensor2 ON
sensor3 OFF
sensor1 ON
sensor2 ON
sensor3 OFF
sensor1 ON
sensor2 ON
sensor3 OFF
```

Figure 11. Output of distance sensors for fall prevention.

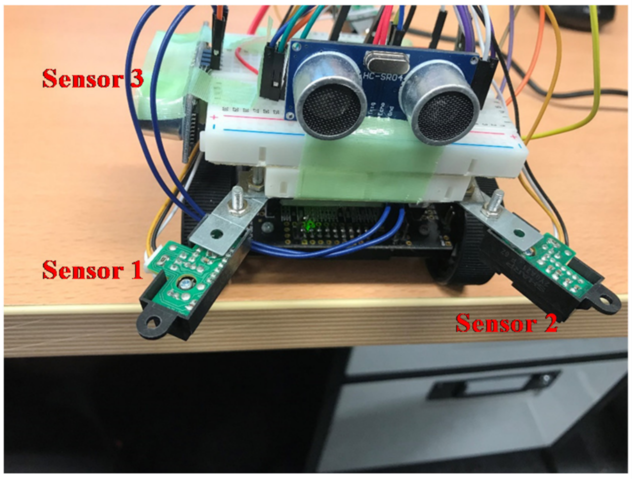


Figure 12. Robot at the time of the sensor output shown in Figure 11.

The robot's operation involved rectilinear movement until the obstacle was found, and the avoidance of this obstacle so that the robot could return to its original travel path. Figure 13 shows the position of the robot when it detected the obstacle, and the random coordinate location of the robot during avoidance measured using a ruler. The measured coordinates of the robot when these measurements were taken are shown in Figure 14, and both were virtually consistent. Please see Supplementary Materials Video S1 for the obstacle avoidance performance.

(4) Overall regular travel—experimental results

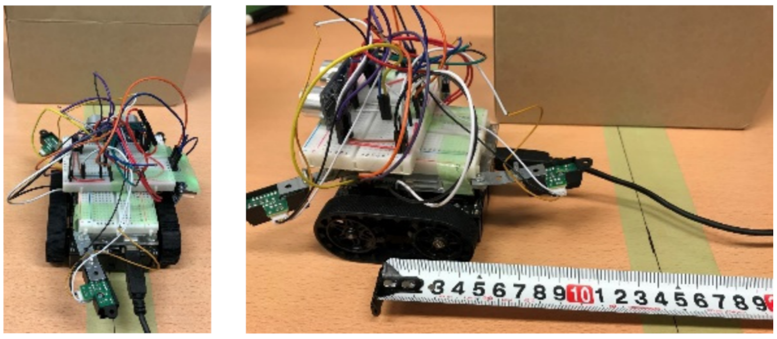
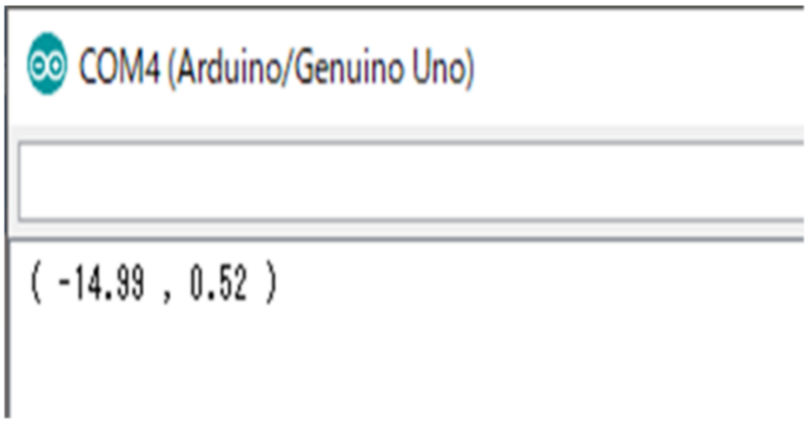


Figure 13. Robot detecting the obstacle (**left**); robot during obstacle avoidance (**right**).



```
( -14.99 , 0.52 )
```

Figure 14. Robot's own coordinate calculation results.

First, we conducted an experiment regarding the overall regular travel in the experimental environment shown in Figure 7. The experimental results of an overall travel of the robot on the table are demonstrated in Supplementary Materials Video S1. In the experiment shown in Supplementary Materials Video S1, the travel-path length and time were 465 cm and 75 s, respectively. For the same experiment, the measurement results for the errors in the location of the robot during travel are shown in Figure 15, and the trajectory of the expected travel and the real robot movement are shown in Figure 16. The errors were measured by drawing a straight line on the tape we applied on the table, and by having the robot travel autonomously on that line. The errors showed a tendency to increase depending on the traveled distance.

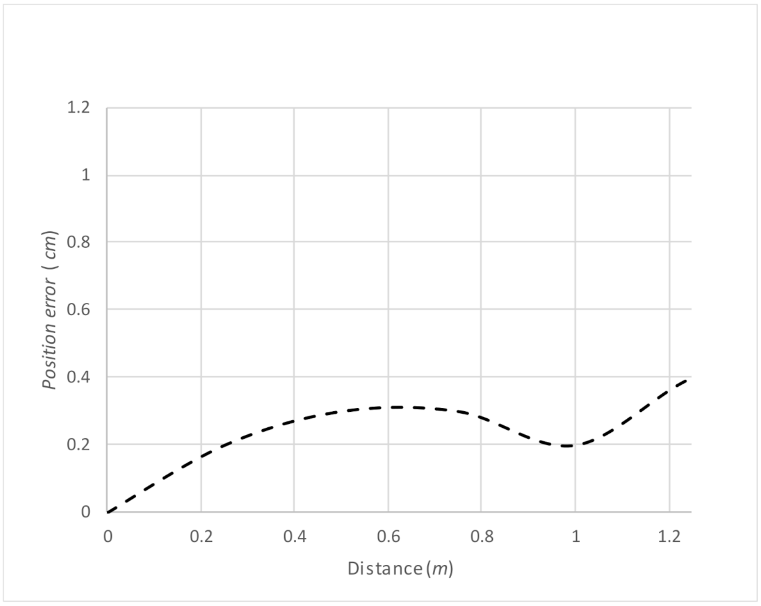


Figure 15. Errors in robot location during travel in the environment shown in Figure 8.

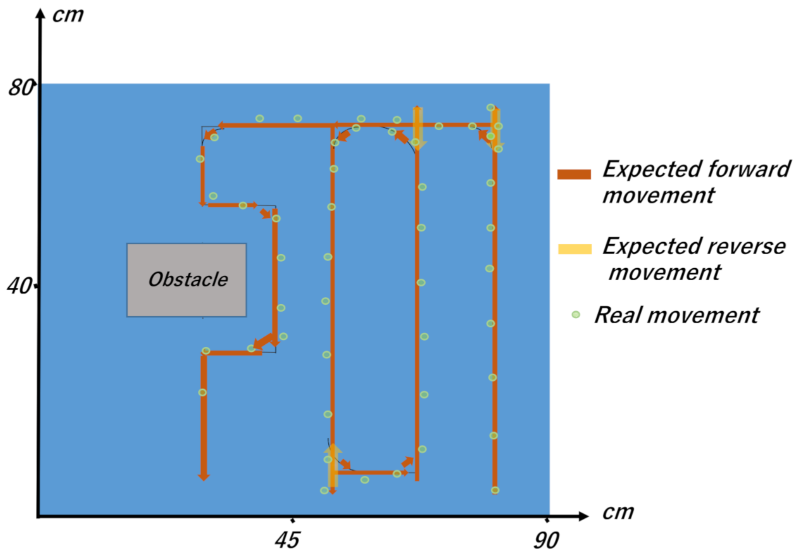


Figure 16. Trajectory of the expected travel and the real robot movement.

Except for the environment shown in Figure 8, we further conducted experiments regarding the overall regular travel in different environments. Some of these environments are shown in Figure 17. Here, the experiments were conducted by changing the position of the obstacles as well.

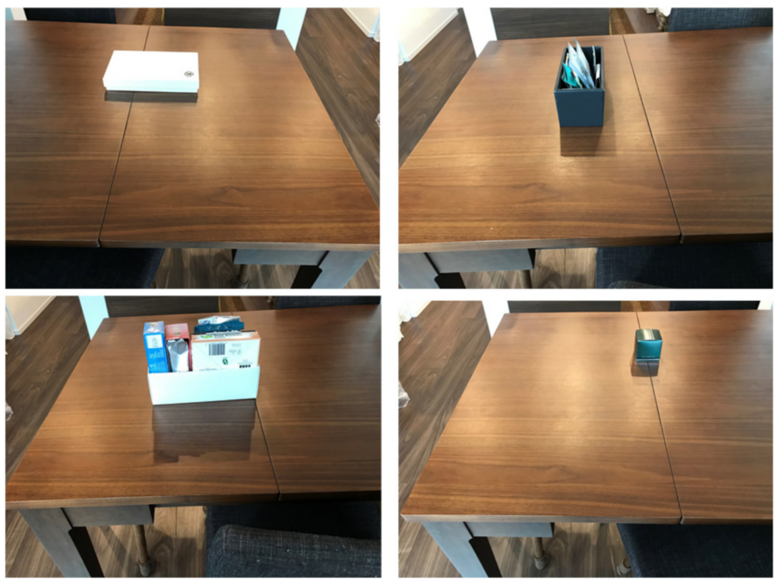


Figure 17. Different experimental environments.

In this paper, the evaluation was conducted based on the average position error during the motion. We separately measured the position error of straight motion and the position error of obstacle-avoidance motion. Table 2 illustrates the measured information. The average position error varied from 0.9 cm to 1.7 cm during the straight motion, while it varied from 1.3 cm to 2.1 cm during the obstacle avoidance motion. In addition to that, the average position error during the obstacle avoidance was always higher than that of straight motions. Here, during the obstacle avoidance, the robot made a motion following the information from the ultrasonic sensors. The main reason for the errors was that the ultrasonic sensors' output included errors for certain levels.

Table 2. Average position error.

Exp. Number	Average Position Error during Straight Motion (cm)	Average Position Error during Obstacle Avoidance (cm)
1	1.4	1.8
2	1.4	1.9
3	1.4	1.6
4	1.3	1.5
5	1.6	1.5
6	1.7	2.1
7	1.5	1.7
8	0.8	1.3
9	1.2	1.6
10	1.3	1.7
11	1.4	1.5
12	1.7	1.8
13	1.1	1.5
14	0.9	1.4
15	1.3	1.5

5.3. Discussion Regarding Conventional Methods

As mentioned above, the target of this work was to achieve spiral travel of a small-type mobile robot while touching the maximum surface of an obstacle-available flat area (like a table), only with onboard lightweight hardware. There are many interesting studies that have been conducted for robot motion generation. Many of them have conducted only simulation experiments to make their evaluations [43–45]. However, in this work, we conducted all the experiments after manufacturing a real machine (mobile robot). Famous algorithms like the lawnmower algorithm could not be applied [48] in this work, since the implementation of the algorithm needs comparatively larger sensing devices like Lidar, which could not be installed on the robot of this work. On the other hand, some conventional studies have also been conducted to move the robot to a specific target while avoiding the obstacles with real machines [40,41]. The problem settings of those studies were different from this study. However, we could apply some of those works to achieve only straight motions in this study. Furthermore, many studies can be found in the literature for obstacle detection and avoidance [5,33,40,41,45,49–52]. However, in this study, the obstacle avoidance was done by keeping a constant distant from the obstacle, as shown in Figure 8. In addition to that, after avoiding the obstacle, the robot returned to the spiral motion and continued that motion. As mentioned above, in this work, the obstacle avoidance and returning to the original spiral motion was done based on onboard ultrasonic sensor outputs. According to our observations, we could not find any study in the literature that conducted a similar kind of obstacle-avoidance process. It is a unique motion generation of this study.

6. Conclusions

The study focused on developing a small robot that can travel on a flat surface, such as a table or solar panel, while covering most of the surface area. We equipped a robot with small sensors and a single-board microcontroller, and investigated the spiral travel path of the robot on an obstacle-available surface. We processed data obtained from these sensors and used the sensors to achieve attitude control and regular travel, obstacle avoidance, subsequent return to the original path, and fall prevention. In the experiments conducted so far, some level of error occurred in the case of regular travel, since the expected robot position did not coincide with the real robot position. We plan to work on correcting these errors in future studies. We successfully achieved obstacle avoidance for obstacles of a certain size and shape; we plan to study the avoidance of smaller obstacles in the future. In addition to that, achieving the spiral movement in moving obstacle-available environments would also be an interesting future stage.

Supplementary Materials: The following are available online at https://www.mdpi.com/article/10.3390/mi12040375/s1, Video S1: Video of the experimental results.

Author Contributions: Conceptualization, S.T.; methodology, S.T. and C.P.; software, S.T.; validation, S.T. and C.P.; formal analysis, H.K., S.S., B.S.S.; investigation, H.W.H.P.; resources, C.P.; data curation, S.T.; writing—original draft preparation, S.T. and H.W.H.P.; writing—review and editing, C.P.; visualization, S.T.; supervision, C.P.; project administration, C.P.; funding acquisition, C.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Branding Research Fund of Shibaura Institute of Technology.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, L.; Yang, Y.Z.; Wang, R. Inertia wheel pendulum robot balance control based on double closed-loop control system. In Proceedings of the 2018 37th Chinese Control Conference (CCC), Wuhan, China, 25–27 July 2018; pp. 5021–5028.
2. Yekinni, L.A.; Dan-Isa, A. Fuzzy logic control of goal-seeking 2-wheel differential mobile robot using unicycle approach. In Proceedings of the 2019 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS), Selangor, Malaysia, 29 June 2019; pp. 300–304.
3. Wang, X.; Ge, H.; Zhang, K.; Chen, Y. System Design and Analysis of Outdoor Obstacle Surmounting Experiments for the Robot with Foldable Wheels. In Proceedings of the 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, China, 15–17 March 2019; pp. 266–270.
4. Liao, J.; Chen, Z.; Yao, B. Model-based coordinated control of four-wheel independently driven skid steer mobile robot with wheel–ground interaction and wheel dynamics. *IEEE Trans. Ind. Inf.* **2019**, *15*, 1742–1752. [CrossRef]
5. Premachandra, C.; Gohara, R.; Ninomiya, T.; Kato, K. Smooth automatic stopping system for ultra-compact vehicles. *IEEE Trans. Intel. Vehicles* **2019**, *4*, 561–568. [CrossRef]
6. Premachandra, C.; Murakami, M.; Gohara, R.; Ninomiya, T.; Kato, K. Improving landmark detection accuracy for self-localization through baseboard recognition. *Int. J. Mach. Learn. Cybernet.* **2017**, *8*, 1815–1826. [CrossRef]
7. Ihalage, T.L.; Perera, A.N.; Sarathchandra, H.A.H.Y.; Premachandra, C. SLAM-based autonomous indoor navigation system for electric wheelchairs. In Proceedings of the 2020 International Conference on Image Processing and Robotics (ICIP), Negombo, Sri Lanka, 6–8 March 2020.
8. Tsunoda, M.; Premachandra, C.; Sarathchandra, H.A.H.Y.; Perera, K.L.A.N.; Lakmal, I.T.; Premachandra, H.W.H. Visible Light Communication by Using LED Array for Automatic Wheelchair Control in Hospitals. In Proceedings of the 2019 IEEE 23rd International Symposium on Consumer Technologies (ISCT), Ancona, Italy, 19–21 June 2019; pp. 210–215.
9. Premachandra, C.; Okamoto, Y.; Kato, K. High performance embedding environment for reacting suddenly appeared road obstacles. In Proceedings of the 2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014), Bali, Indonesia, 5–10 December 2014; pp. 2394–2397.
10. Yu, W.; Lin, D.; Song, T. Adaptive Control for UAV Close Formation Flight against Disturbances. In Proceedings of the 2018 3rd International Conference on Robotics and Automation Engineering (ICRAE), Guangzhou, China, 17–19 November 2018; pp. 196–201.
11. Premachandra, C.; Takagi, S.; Kato, K. Flying control of small-type helicopter by detecting its in-air natural features. *J. Electr. Sys. Inf. Technol.* **2015**, *2*, 58–74. [CrossRef]
12. Zhang, D.; Chen, Z.; Xi, L. Adaptive dual fuzzy PID control method for longitudinal attitude control of tail-sitter UAV. In Proceedings of the 2016 22nd International Conference on Automation and Computing (ICAC), Colchester, UK, 7–8 September 2016; pp. 378–382.
13. Premachandra, C.; Otsuka, M.; Gohara, R.; Ninomiya, T.; Kato, K. A study on development of a hybrid aerial/terrestrial robot system for avoiding ground obstacles by flight. *IEEE/CAA J. Autom. Sin.* **2018**, *6*, 327–336. [CrossRef]
14. Nakajima, K.; Premachandra, C.; Kato, K. 3D environment mapping and self-position estimation by a small flying robot mounted with a movable ultrasonic range sensor. *J. Electr. Syst. Inf. Technol.* **2017**, *4*, 289–298. [CrossRef]
15. Yao, Z.; Wu, S. Intermittent Gliding Flight Control Design and Verification of a Morphing Unmanned Aerial Vehicle. *IEEE Access* **2019**, *7*, 40991–41005. [CrossRef]
16. Premachandra, C.; Thanh, D.N.H.; Kimura, T.; Kawanaka, H. A study on hovering control of small aerial robot by sensing existing floor features. *IEEE/CAA J. Autom. Sin.* **2020**, *7*, 1016–1025. [CrossRef]

17. Premachandra, C.; Ueda, D.; Kato, K. Speed-up automatic quadcopter position detection by sensing propeller rotation. *IEEE Sens. J.* **2019**, *19*, 2758–2766. [CrossRef]
18. Higuchi, S.; Arimura, R.; Premachandra, C.; Kato, K. Design of two degree of freedom controller using data conversion method. In Proceedings of the 2016 16th International Conference on Control, Automation and Systems (ICCAS), Gyeongju, Korea, 16–19 October 2016; pp. 1067–1072.
19. Wang, H.; Wang, H.; Xu, W.; Mu, Z. Development and experiment of a snake-like robot composed of modularized isomorphic joints. In Proceedings of the 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Budapest, Hungary, 9–12 October 2016; pp. 3160–3165.
20. Geng, S.; Peng, S.; Han, Y. Research on Motion Planning of Snake-Like Robot Based on the Interpolation Function. In Proceedings of the 2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, 27–28 August 2016; pp. 81–84.
21. Rano, I.; Eguiluz, A.G.; Sanfilippo, F. Bridging the Gap between Bio-Inspired Steering and Locomotion: A Braitenberg 3a Snake Robot. In Proceedings of the 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV), Singapore, 18–21 November 2018; pp. 1394–1399.
22. Sanfilippo, F.; Stavdahl, Ø.; Liljebäck, P. SnakeSIM: A ROS-based rapid-prototyping framework for perception-driven obstacle-aided locomotion of snake robots. In Proceedings of the 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, China, 5–8 December 2017; pp. 1226–1231.
23. Branyan, C.; Hatton, R.L.; Menguc, Y. Snake-Inspired Kirigami Skin for Lateral Undulation of a Soft Snake Robot. *IEEE Robot. Autom. Lett.* **2020**, *5*, 1728–1733. [CrossRef]
24. Zhou, Z.; Wang, H.; Li, D.; Deng, H. Motion Control Curve of Snake-like Robot Based on Centroid Stability. In Proceedings of the 2019 IEEE International Conference on Unmanned Systems (ICUS), Atlanta, GA, USA, 11–14 June 2019; pp. 826–830.
25. Manzoor, S.; Choi, Y. Modular design of snake robot for various motions implementation. In Proceedings of the 2016 13th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), Xian, China, 19–22 August 2016; pp. 211–213.
26. Chavan, P.; Murugan, M.; Unnikkannan, E.V.; Singh, A.; Phadatare, P. Modular Snake Robot with Mapping and Navigation: Urban Search and Rescue (USAR) Robot. In Proceedings of the 2015 International Conference on Computing Communication Control and Automation, Washington, DC, USA, 26–27 February 2015; pp. 537–541.
27. Park, C.-H.; Park, K.-T.; Gweon, D.-G. Development of Industrial Dual Arm Robot for Precision Assembly of Mechanical Parts for Automobiles. In Proceedings of the 2006 SICE-ICASE International Joint Conference, Busan, Korea, 18–21 October 2006; pp. 3059–3062.
28. Jhang, L.-H.; Santiago, C.; Chiu, C.-S. Multi-sensor based glove control of an industrial mobile robot arm. In Proceedings of the 2017 International Automatic Control Conference (CACS), Pingtung, Taiwan, 12–15 November 2017; pp. 1–6.
29. Choi, T.; Do, H.; Park, K.T.; Kim, D.; Kyung, J. Small sized industrial dual-arm robot with convenient program interface. In Proceedings of the IEEE ISR, Seoul, Korea, 24–26 October 2013; pp. 1–5.
30. Liu, B.; He, Y.; Kuang, Z. Design and Analysis of Dual-arm SCARA Robot Based on Stereo Simulation and 3D Modeling. In Proceedings of the 2018 IEEE International Conference on Information and Automation (ICIA), Fujian, China, 11–13 August 2018; pp. 1233–1237.
31. Liu, S.; Zheng, L.; Wang, S.; Li, R.; Zhao, Y. Cognitive abilities of indoor cleaning robots. In Proceedings of the 2016 12th World Congress on Intelligent Control and Automation (WCICA), Guilin, China, 12–15 June 2016; pp. 1508–1513.
32. Bae, Y.G.; Jung, S. Manipulability and kinematic analysis of a home service robot aimed for floor tasks. In Proceedings of the 2012 9th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), Daejeon, Korea, 26–28 November 2012; pp. 385–388.
33. Zhao, Z.; Chen, W.; Peter, C.C.; Wu, X. A novel navigation system for indoor cleaning robot. In Proceedings of the 2016 IEEE International Conference on Robotics and Biomimetics (ROBIO), Qingdao, China, 3–7 December 2016; pp. 2159–2164.
34. Kim, J.; Cauli, N.; Vicente, P.; Damas, B.; Cavallo, F.; Santos-Victor, J. "iCub, clean the table!" A robot learning from demonstration approach using deep neural networks. In Proceedings of the 2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), Torres Vedras, Portugal, 25–27 April 2018; pp. 3–9.
35. Kleiner, A.; Baravalle, R.; Kolling, A.; Pilotti, P.; Munich, M. A solution to room-by-room coverage for autonomous cleaning robots. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 5346–5352.
36. Cheng, Y.; Li, C.; Li, S.; Li, Z. Motion Planning of Redundant Manipulator with Variable Joint Velocity Limit Based on Beetle Antennae Search Algorithm. *IEEE Access* **2020**, *8*, 138788–138799. [CrossRef]
37. Li, Z.; Li, S. Saturated PI Control for Nonlinear System with Provable Convergence: An Optimization Perspective. *IEEE Trans. Circuits Syst. II Express Briefs* **2021**, *68*, 742–746. [CrossRef]
38. Li, Z.; Zuo, W.; Li, S. Zeroing dynamics method for motion control of industrial upper-limb exoskeleton system with minimal potential energy modulation. *Measurement* **2020**, *163*, 107964. [CrossRef]
39. Li, Z.; Li, C.; Li, S.; Cao, X. A Fault-Tolerant Method for Motion Planning of Industrial Redundant Manipulator. *IEEE Trans. Ind. Inform.* **2020**, *16*, 7469–7478. [CrossRef]
40. Pandey, A.; Panwar, V.S.; Hasan, E.; Parhi, D.R. V-REP-based navigation of automated wheeled robot between obstacles using PSO-tuned feedforward neural network. *J. Comput. Des. Eng.* **2020**, *7*, 427–434. [CrossRef]

41. Pandey, A.; Kashyap, A.K.; Parhi, D.R.; Patle, B. Autonomous mobile robot navigation between static and dynamic obstacles using multiple ANFIS architecture. *World J. Eng.* **2019**, *16*, 275–286. [CrossRef]
42. Pandey, A.; Bej, N.; Kumar, R.; Panda, A.; Parhi, D.R. Type-2 Fuzzy Controller (T2FC) Based Motion Planning of Differential-Drive Pioneer P3-DX Wheeled Robot in V-REP Software Platform. In *A Journey Towards Bio-inspired Techniques in Software Engineering*; Metzler, J.B., Ed.; Springer: Cham, Switzerland, 2020; pp. 47–57.
43. Yakoubi, M.A.; Laskri, M.T. The path planning of cleaner robot for coverage region using Genetic Algorithms. *J. Innov. Digit. Ecosyst.* **2016**, *3*, 37–43. [CrossRef]
44. Wang, Z.; Bo, Z. Coverage path planning for mobile robot based on genetic algorithm. In Proceedings of the 2014 IEEE Workshop on Electronics, Computer and Applications, Ottawa, ON, Canada, 8–9 May 2014; pp. 5–14.
45. Hasan, K.; Abdullah-Al-Nahid, K. Reza, Path planning algorithm development for autonomous vacuum cleaner robots. In Proceedings of the 2014 International Conference on Informatics, Electronics & Vision (ICIEV), Dhaka, Bangladesh, 23–24 May 2014; pp. 1–6.
46. Al-Fahoum, A.S.; Abadir, M.S. Design of a Modified Madgwick Filter for Quaternion-Based Orientation Estimation Using AHRS. *Int. J. Comput. Electr. Eng.* **2018**, *10*, 174–186. [CrossRef]
47. Demirhan, M.; Premachandra, C. Development of an Automated Camera-Based Drone Landing System. *IEEE Access* **2020**, *8*, 202111–202121. [CrossRef]
48. Chung, C.-H.; Wang, K.-C.; Liu, K.-T.; Wu, Y.-T.; Lin, C.-C.; Chang, C.-Y. Path Planning Algorithm for Robotic Lawnmower using RTK-GPS Localization. In Proceedings of the 2020 International Symposium on Community-centric Systems (CcS), Tokyo, Japan, 23–26 September 2020.
49. Ito, Y.; Premachandra, C.; Sumathipala, S.; Premachandra, H.W.H.; Sudantha, B.S. Tactile Paving Detection by Dynamic Thresholding Based on HSV Space Analysis for Developing a Walking Support System. *IEEE Access* **2021**, *9*, 20358–20367. [CrossRef]
50. Su, H.; Mariani, A.; Ovur, S.E.; Menciassi, A.; Ferringo, G.; Momi, E.D. Toward Teaching by Demonstration for Robot-Assisted Minimally Invasive Surgery. *IEEE Trans. Autom. Sci. Eng.* **2021**. [CrossRef]
51. Su, H.; Qi, W.; Hu, Y.; Karimi, H.R.; Ferringo, G.; Momi, E.D. An Incremental Learning Framework for Human-like Redundancy Optimization of Anthropomorphic Manipulators. *IEEE Trans. Ind. Inform.* **2020**. [CrossRef]
52. Su, H.; Hu, Y.; Karimi, H.R.; Knoll, A.; Ferrigno, G.; De Momi, E. Improved recurrent neural network-based manipulator control with remote center of motion constraints: Experimental results. *Neural Netw.* **2020**, *131*, 291–299. [CrossRef] [PubMed]