computers

IMPACT FACTOR 2.6

CITESCORE 5.4

Article

Low-Cost Multisensory Robot for Optimized Path Planning in Diverse Environments

Rohit Mittal, Geeta Rani, Vibhakar Pathak, Sonam Chhikara, Vijaypal Singh Dhaka, Eugenio Vocaturo and Ester Zumpano

Special Issue

Advances in Database Engineered Applications 2023

Edited by

Prof. Dr. Richard Chbeir, Prof. Dr. Yannis Manolopoulos, Prof. Dr. Mirjana Ivanović and Dr. Claudio Silvestri

MDPI

https://doi.org/10.3390/computers12120250

computers

MDPI

Article

Low-Cost Multisensory Robot for Optimized Path Planning in Diverse Environments

Rohit Mittal [1], Geeta Rani [1,*], Vibhakar Pathak [2], Sonam Chhikara [1], Vijaypal Singh Dhaka [1], Eugenio Vocaturo [3,4] and Ester Zumpano [3,4]

[1] Computer and Communication Engineering, Manipal University Jaipur, Jaipur 303007, India; rohit.mittal@jaipur.manipal.edu (R.M.); sonam.209301273@muj.manipal.edu (S.C.); vijaypalsingh.dhaka@jaipur.manipal.edu (V.S.D.)
[2] Department of Information Technology, Arya College of Engineering and Information Technology, Jaipur 302028, India; vibhakar@aryacollege.in
[3] Department of Computer Engineering, Modeling, Electronics and Systems, University of Calabria, 87036 Rende, Cosenza, Italy; e.vocaturo@dimes.unical.it (E.V.); e.zumpano@dimes.unical.it (E.Z.)
[4] National Research Council, Institute of Nanotechnology (CNR-NANOTEC) Ponte P. Bucci, Cubo 31/C, 87036 Rende, Cosenza, Italy
* Correspondence: geetachhikara@gmail.com

Abstract: The automation industry faces the challenge of avoiding interference with obstacles, estimating the next move of a robot, and optimizing its path in various environments. Although researchers have predicted the next move of a robot in linear and non-linear environments, there is a lack of precise estimation of sectorial error probability while moving a robot on a curvy path. Additionally, existing approaches use visual sensors, incur high costs for robot design, and ineffective in achieving motion stability on various surfaces. To address these issues, the authors in this manuscript propose a low-cost and multisensory robot capable of moving on an optimized path in diverse environments with eight degrees of freedom. The authors use the extended Kalman filter and unscented Kalman filter for localization and position estimation of the robot. They also compare the sectorial path prediction error at different angles from 0° to 180°and demonstrate the mathematical modeling of various operations involved in navigating the robot. The minimum deviation of 1.125 cm between the actual and predicted path proves the effectiveness of the robot in a real-life environment.

Keywords: sectorial error; multi-sensor; optimize; robot; Kalman filter

Citation: Mittal, R.; Rani, G.; Pathak, V.; Chhikara, S.; Dhaka, V.S.; Vocaturo, E.; Zumpano, E. Low-Cost Multisensory Robot for Optimized Path Planning in Diverse Environments. *Computers* **2023**, *12*, 250. https://doi.org/10.3390/computers12120250

Academic Editors: Richard Chbeir, Yannis Manolopoulos, Mirjana Ivanović and Claudio Silvestri

Received: 31 July 2023
Revised: 21 November 2023
Accepted: 22 November 2023
Published: 1 December 2023

1. Introduction

Estimating the next move of a robot in an unknown environment is a key concern [1–4]. Simultaneous Localization and Mapping (SLAM) technology is employed to estimate the current position of a robot on a known map. It has a representation of objects in terms of their location. The identification of these objects is important to avoid obstacles during the movement of the robot [5]. SLAM contains a feature representation of a location. It follows two approaches, namely the localization approach and the map approach, for determining the next move of a robot. The location approach considers points from a fixed location, number of wheel revolutions, and data captured from sensors to determine the amount and direction of next move. In the map approach, a robot knows its position inside the given map and decides the next move based on its current position and surrounding environment [6]. The Kalman filter (KF) is used for predicting linear movement at $t + 1$ time. The extended version of KF is used for the non-linear movement of the robot [7,8]. Both are implemented on a robot for localization and to estimate the position using its previous coordinates [9]. KF is an optimal estimator based on various parameters of interest from indirect, inaccurate, and uncertain observations. The SLAM with EKF is applied where the dynamics of the state model and measurement model are moderately non-linear. It figures out the mapping approach in the frame of the linear filter in which it is utilized [6]. The

unscented Kalman filter (UKF) uses an Unscented Transformation (UT) for optimizing the path of a robot on a non-linear path. The authors in reference [10] discussed the combination of measurement and motion models to identify the precise estimator. They estimated the position and motion of a robot moving with a constant velocity, acceleration, and turn rate, as well as its orientation. On similar lines of research, the authors in [11], implemented the path planning of aerial vehicles using the UKF [11]. Next, the team of researchers in [12] used the Visual SLAM algorithm that allows Universal Asynchronous Receiver Transmitter (UART) communication to obtain the real-time pose coordinates of a robot.

From the above discussion, it is apparent that the researchers extended the applications of EKF and UKF for optimizing the paths of robots on a linear as well as non-linear paths [6,8,10,11]. But, calculating the deviation of a robot from its actual path when it moves on a curvy path is underexplored. This leaves a scope to estimate the Sectorial Error Probability (SEP) between the actual and predicted pose of a robot using non-visual sensors. SEP is the localization probability on a map when a robot moves along a sectorial path with (x, y) coordinates concerning its actual pose. It is useful in determining the actual, optimized, and predicted positions of a robot. In this research, we calculated the sectorial turn of the robot and evaluated the impact of friction on movement of a robot. The calculated values vary from 0° to 15°; 15° to 30°; 30° to 45°; 45° to 90°; 90° to 120°; 120° to 180°.

Also, the existing works focus on SLAM-based augmented reality devices and restrict the degrees of freedom (DOF) to six for the movement estimation and four for the pose estimation of a robot. Here, DOF means how independently sensors mounted on a robot can collect and transmit data without interfering each other; 6-DOF involves data collection by six independent sensors. For example, the 3D coordinates of curve landmark parameters are collected in the form of three control points relative to body frame of camera. The remaining three parameters are collected by stereo camera, an IMU, and GPS.

This leaves a scope to extend the DOF to eight while controlling the movements of a robot. The authors in this paper resolve the above-identified challenges and propose a robot design equipped with multi-sensors to determine the environment and localization, which is subtended to EKF and UKF for better path estimation with 8-DOF.

Here, 8-DOF includes the dynamic acceleration, orientation, distance from an object, and change in state per unit time of a robot. The design of a robot includes the eight modelling parameters, such as A_x, A_y, and A_z, which are the dynamic acceleration of the robot in the X, Y, and Z axes respectively. G_x, G_y, and G_z are the orientation of a robot moving along X, Y, and Z axes, respectively. The distance (ϕ) of an object from a robot is measured using SONAR. The change in state of a moving robot per unit time (θ) is detected using INFRARED.

To overcome the above-identified challenges, we propose a design of an autonomous robot that senses the known environment with the help of data captured in real-time. It uses a set of ancillary sensory components for localization and to improve the movement accuracy of a robot in a known mapped environment. It uses a three-dimensional (3D) accelerometer sensor from Texas Instruments (TI eZ430-Chronos watch) that is interfaced and secured with a programmable system [6,7]. We extend the applicability of the Chronos watch to move the robot in the X, Y, and Z axes. This avoids the use of an accelerometer sensor and saves the cost of designing a robot. Now, the applied EKF uses the two-step prediction corrector algorithm for estimating the updated state of the robot. Its first step projects the approximate current state of the robot and an estimated error from the previous state. Based on these values, it predicts the state of the robot. The second step applies a measurement model to estimate the updated state of a robot. This results in a blending of forces and modifies the robot to establish quick control [6,10,11]. In addition to applying EKF and UKF, the authors modified the EKF by augmenting its results with values of environment coefficient obtained from various sensors. The overview of the proposed approach is demonstrated in Figure 1. Major contributions of this paper are listed below:

1. Designing a low-cost, multi-sensor fusion robot with low hardware requirements.

2. Reducing sectorial error probability and increasing degree of freedom to eight.
3. Predicting the path of a robot using SLAM-based modified EKF and UKF with 8-DOF.
4. Improving the stability of a robot while moving on different surface types.

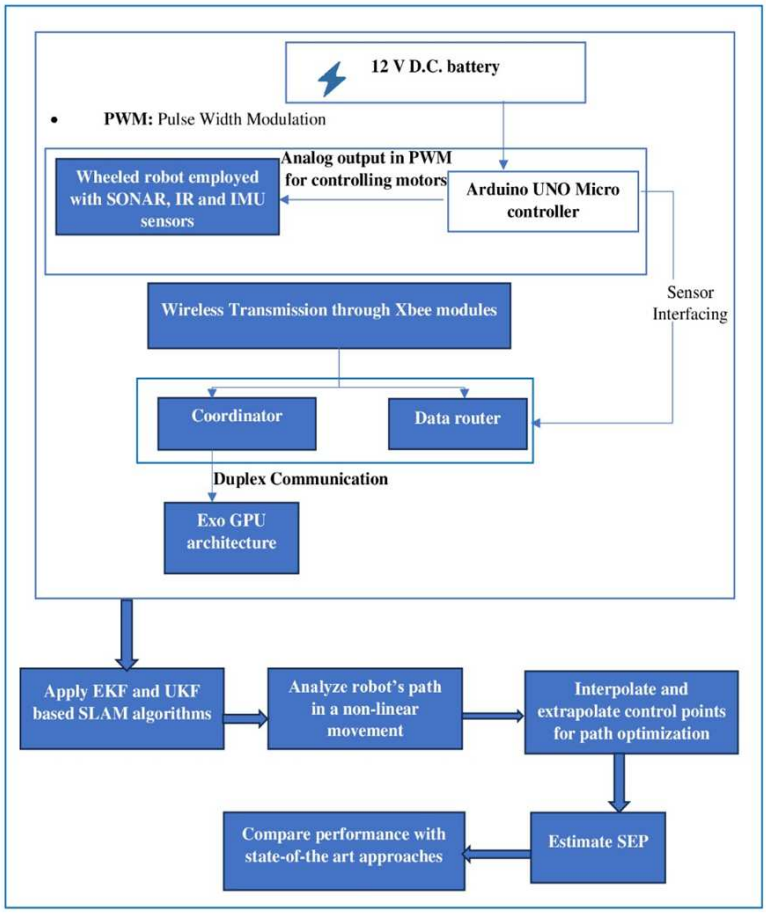

Figure 1. Overview of the proposed approach.

2. Related Works

The extensive study of the literature reveals that the authors focused on mobile robots use EKF as an asymptotic observer for predicting the state of a robot on a given map [13]. The important parameters of a robot are the localization coefficient between the body of the robot and the ground; a moment of inertia; and physical parameters such as friction, impulse, and force [5]. The authors in [14] used EKF to analyze mean covariance and proportional derivative controller as a parameter to evaluate the performance of non-linear systems. The authors in [8] analyzed the force exerted when a robot was moving on a loose surface. However they could not achieve path optimization for a robot with 8-DOF. Next, the authors implemented SLAM with EKF and the blending of two control points for smoothing the path [15]. They successfully computed the shortest path between two designated points but did not focus on smoothing and optimizing the path for ground vehicles. The team of researchers proposed an integration of laser sensors and EKF to estimate the position of a robot [7] using 3-DOF. Similarly, the authors in [16] mentioned the 6-DOF pose of an autonomous robot, which uses a camera as one of the vision sensors and an Inertial Measurement Unit (IMU) as a digital sensor. Now, the authors in [17] employed the localization method for maintaining 6-DOF as C-SLAM-based on the Canny operator for the detection of edges in images. However the path optimization of robots based on 8-DOF could not be achieved [18]. The authors in reference [13] discussed the navigation of the robot on an unknown map with unknown localization. They used Bluetooth sensors to calculate the force and impulse exerted on the EZ 430 Chronos watch. They did not

increase the DOF to eight. The authors in [19] focused on 6D pose estimation of robots using SLAM techniques.

Similarly the researchers in [20] estimated the pose of a robot with 6-DOF and compared curveSLAM algorithms with visual inertial SLAM. This work leaves the scope to integrate more sensors such as IMU, Infrared, SONAR, Xbee modules in a robot to increase its applications. Moreover, the curveSLAM algorithm can be modified for estimating the 8-DOF pose of robot. Further, the work presented in [21], follows a relative localization-based approach for measuring the distance between two SLAM-based augmented reality devices and estimating the the position of a robot in 4-DOF for 2D and 3D systems. This work lacks in measuring the deviation of the robot's actual, predicted, and optimized path. It is devoid of multisensory fusion technology. In reference [22], an indoor localization method with IMU and Ultra-Wide Band (UWB) position method on Unmanned Aerial Vehicles (UAV) was simulated. The path planning of UAVs was implemented using UKF only. However, multisensory fusion technology based on UKF can improve the accuracy of solving non-linear equations. Here, the path optimization of the multisensory robot with EKF and UKF was not discussed. The authors in [23], extended the work in [22] and employed the autoVisual SLAM algorithm to obtain real-time pose coordinates for the UAV's 3D map of the surrounding. This work lacks multisensory fusion technology and verification of the Arduino simulator with BSP on Microsoft platform.

It is evident from the above discussion that researchers worked to predict the optimized path of a robot in linear as well as non-linear environments. The EKF-based SLAM's mathematical model linearizes the path on two control points. There is a lack of precise estimation of SEP during the movement of a robot on a curvy path. This raises the need to analyze the movement of a robot on a sectorial turn. Also, there is a need to evaluate the motion stability of a robot on a diverse set of surfaces such as boulder tiles, marble flooring, glaze tiles, etc. Movements on such surfaces give an estimation of the frictional parameter between the tires of the robot and the surface [6,8,24]. For example, there is a possibility of collision of a losing structure on the surface, and disturbance in SEP. In such scenarios, the curve-based EKF and UKF SLAM algorithm can distinguish feature points while navigating robots on structured maps in a given environment. Furthermore, the existing approaches could achieve path optimization with 6-DOF. However, there is a scope to optimize the path of a robot in a multi-sensory environment with 8-DOF, and to estimate its precise SEP.

3. Materials and Methods

In this section, the authors present a detailed discussion on the proposed design of a robot, and experiments conducted to evaluate the potency of the designed robot.

3.1. Robot Design

The wheeled robot consists of a 12 V D.C. motor with an independent accelerometer mode in eZ430 Chronos(Texas Instrument, Jaipur, India). It includes a Texas Instrument (TI) eZ430-Chronos watch [25,26], which can be integrated with the system. The Chronos watch is a portable device and has a high degree of precision. It has a built-in and adjustable accelerometer. Sensors like SONAR, Infrared, and Inertial Measurement Unit (IMU) are mounted on the Arduino UNO, which works as a controller board for the robot. Now, the sensors are interfaced via Zigbee modules, as shown in Figure 2. All the sensors are interfaced with a pair of Xbee modules for wireless transmission. Then, the Bluetooth module HC-06 was deployed on the robot to navigate it using the Arduino remote control application [6]. Here, Vmicro is used as a Board Support Package (BSP) for the Arduino UNO board [7]. Now, the designed robot is connected to our laptop that supports the Exo GPU architecture.

The robot is designed with 8-DOF that includes: A_x, A_y, and A_z as the dynamic acceleration of the robot in the X, Y, and Z axes, respectively; G_x, G_y, and G_z as the orientation of the moving robot in the X, Y, and Z axes respectively; distance (ϕ) of an

object from the robot measured using SONAR; and the motion (θ) of a robot detected using INFRARED. All these parameters are important for the navigation and path planning of a robot. The robot design proposed in this research applies force in one direction. Also, the robot uses frictional steering instead of rack-and-pinion steering for moving on a sectorial turn. Further, the wheel profile is designed without suspension, making it rigid and fixed on a round plate. Each wheel generates different power, enabling the robot to maneuver effectively on diverse surfaces, which creates a vector in the X and Y directions. Whenever the vectorization of the pose changes, then the robot turns in a sector.

This research focuses on improving the movement of a robot on both linear and curvy paths. To serve this purpose, a three-wheeled robot was designed, with two traditional wheels and a caster wheel. The caster wheel allows the robot to move easily in any direction and maneuver heavy objects. The traditional wheels have more than four nearby thin treads that improve grip on various surfaces such as marble floors, glaze tiles, rough surfaces, and boulder surfaces.

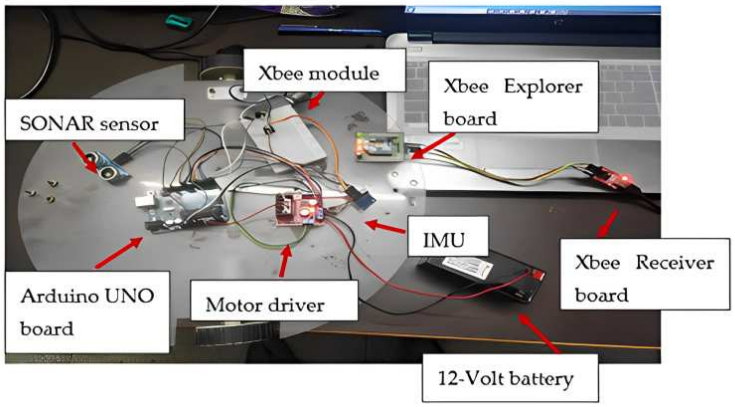

Figure 2. Interfacing of sensors with system.

3.2. Experiments

In this research, we experimented on the designed robot integrated with SLAM [27]. Our experiments depend on designing and integrating our robot with SLAM algorithms. The design of the robot involves hardware and software components like sensors, localization, mapping, control software, and proper communication through the Xbee module. Integration and testing require sensor calibration for acquiring accurate data, software integration, and testing in diverse environments. Thus, we used the Bluetooth controller module and an Android-based application for moving the robot. We installed VS 2.0 simulator [28–30] on the local system and imported Vmicro library [6] for establishing the connections of the robot with an Arduino UNO board.

We used a motor without encoders and a remote area network to move the robot on an area of 4 $\times$ 6 ft in the robotics lab of the university, which has marble flooring. Next, we analyzed its Euclidean distances, impulse, and the forces applied by the robot on the X, Y, and Z coordinates of all the above-mentioned surface types. The robot is found to be the most stable while moving on the marble flooring [6,31]. Therefore, further experiments were conducted on marble flooring only. The values of Euclidean distances and impulse obtained on all the above-mentioned surfaces are shown in Supplementary Materials. These values are important for selecting the most suitable surface for moving a robot. Then we collected real-time data from the sensors mounted on the robot and the accelerometer mode of the eZ430 TI Chronos. The Chronos is also used for transmitting acceleration information to the data center.

To receive stable values for the accelerometer, the authors calculated an average of two consecutive readings, as demonstrated in Table 1. Values of Euclidean distance for an actual path given for pulse number 1 and 2, as highlighted in blue colour in Table 1, are used to calculate the Euclidean distance of pulse number 3, as highlighted in green

colour. The same method is applicable to the predicted and optimized paths. We make effective use of the VS 2.0 simulator for the analysis of data collected at the data center. The simulator uses the predictor-corrector concept to estimate the state of the robot in a non-linear transition model. Initially, the robot maps and then localizes itself on that map. Now, we use ancillary sensory components for data capturing and evaluating the SEP. We employed the SLAM-based EKF and UKF for predicting the path of a multi-terrain robot.

Table 1. Average calculation of Accelerometer using Euclidean distances (cm).

No. of Pulse	Actual Pose		Predicted Pose		Optimized Pose		Actual Path	Predicted Path	Optimized Path
	X-Axis	Y-Axis	X-Axis	Y-Axis	X-Axis	Y-Axis			
1	95	51	79	53	87	52	3.6055	2.2360	2
2	99	55	104	54	102	54	5.6568	25.0199	15.1327
3	110	57	120	58	115	58	11.1803	16.4924	13.6014

Now, objects are moved at random in a simulated environment of obstacles. At each point of time, the robot is monitored by estimating its X and Y coordinates. It is mapped on a multi-touch pad and screen. To estimate the pose of the robot with 8-DOF, the model is equipped with multi-sensors that have IMU and SONAR. These provide the values of multiple parameters and a profiled map of the unknown environment. The sensor components used for testing the robot on the above-mentioned surface are shown in Figure 3. Then, a GUI interface in Visual Studio reads and visualizes the data collected from multiple sensors. Visual Micro plugins provide a platform to execute Arduino-based cross-platform programs. These plugins are compatible with the Arduino UNO board and use the same libraries, development code, and tools.

In the above experiment, the authors used the non-visual sensors Zigbee modules, HC-05, IMU, SONAR, and Infrared to collect real-time data. Thus, we observed an approximately 30% lower cost than the robots designed in the literature using visual components and sensors.

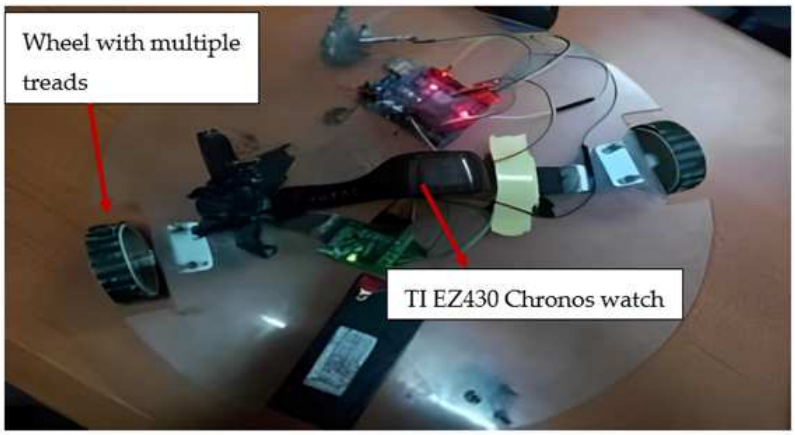

Figure 3. Sensor components with chronos watch [2].

3.3. Mathematical Modelling

In this sub-section, we demonstrate the mathematical modelling of the proposed work applied on the real-time dataset and collected using sensors equipped on the robot with 8-DOF. For estimating the movement of a robot, the mathematical approach relies on the parameters of acceleration and gyroscope [5,32]. The equations using the variables A_x, A_y, A_z, G_x, G_y, G_z, distance between an object and robot measured by SONAR, and the motion of a robot detected by Infrared are illustrated below. Each equation is associated with an activity performed to demonstrate the working of the robot, predict the impact of various parameters and the evaluation of its efficacy in various environments.

1. Analyzing discrete time steps and measurements: The time rates $(x(t+1))$ of EKF are given in Equation (1). Here, we extend the KF to non-linear system modeling for estimating the state of the system from time 't' to updated time '$t+1$'. In this equation, $F(t)x(t)$ is the state dynamics function, and $G(t)u(t)$ is the discrete input gain. Its value is preset to 0 based on the discussion given in reference [33].

$$x(t+1) = F(t)x(t) + G(t)u(t) \tag{1}$$

2. Predicting the relation between two states of motion: The next state of a robot is also dependent on its current state, as shown in Equation (2). In this equation, $x_i\ y_i$ is the initial state, and $\dot{x}\ \dot{y}$ is the next state of the motion of a robot.

$$x = \sum_{i=1}^{8} [x_i, y_i, \dot{x}, \dot{y}] \tag{2}$$

3. Predicting the impact of physical parameters: The path of a robot is also dependent on physical parameters such as the robot's pose, infrared, SONAR, and friction variables. The relationship is shown in Equation (3). In this equation, x, y represents the robot's pose; θ, ϕ, and ψ represent the Infrared, Sonar, and friction variables in navigation, respectively; and $\dot{\beta}$ is the projection of the robot's angular velocity bias gyroscope measurement. The state of motion of a robot on different surfaces is estimated using the outcomes of Equations (1)–(3).

$$\mathbf{x}_n = [x, y, \theta, \phi, \psi, \dot{\beta}] \tag{3}$$

4. Impact of Friction: The state of motion of a robot on different surfaces is estimated using the outcomes of Equations (1)–(3). The state and movement of a robot is also dependent on the friction x_0, as calculated in Equation (4). In this equation, S_x and S_y are the surface variables on the X-axis and Y-axis, respectively. V_x and V_y are velocity variables along the X-axis and Y-axis, respectively. T signifies the transpose of the surface and velocity variables.

$$\mathbf{x}_0 = \begin{bmatrix} S_x \\ S_y \\ V_x \\ V_y \end{bmatrix} \tag{4}$$

5. Predicting the state of robot without using information about the current state: In some cases, the information about the current state of the robot is unknown. Then, we follow Equations (5) and (5a) to predict its next state. We first estimate the state $\hat{x}(t+1 \mid t)$ using the state–space time updation as shown in Equation (5). Then, we obtain the state covariance matrix $\hat{P}(t+1 \mid t)$ at time t using the previous covariance matrix $\hat{P}(t+1 \mid t)$ at time t. The state covariance matrix holds the uncertainty of the states and contains the covariance of each feature, as shown in Equation (5a) [34]. The state covariance matrix deals with the time stamps and velocities of the robot at time t.

$$\hat{x}(t+1 \mid t) = f(t, \hat{x}(t \mid t)) \tag{5}$$

$$f(t, \hat{x}(t \mid t)) = \begin{bmatrix} \theta_{t+1} \\ \phi_{xy,t+1} \end{bmatrix} \overset{where}{=} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} (\theta_t \mid \phi_{xy,t})$$

$$\hat{P}(t+1 \mid t) = F(t)\hat{P}(t \mid t)F^T(t) \tag{5a}$$

$$F(t)\hat{P}(t \mid t)F^T(t) \overset{where}{=} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \hat{P}(t \mid t) \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

6. Measurement Prediction and Minimizing Uncertainty: For precisely estimating the current state of the designed robot, we smoothen the sensor measurements. In the beginning, our state model predicts the next state. Then the measurement prediction model takes the predicted input (A_x and A_y) with the sine and cosine components of the state model, to infer new measurements. While making the predictions, we assume that in each time stamp the robot moves with a constant velocity. So, we employed the sine and cosine components to find the next state at a timestamp in the future. The model is defined in Equations (6) and (6a). In these equations, $\hat{z}(t)$ is the estimation state, and $\hat{s}(t)$ is the state variance matrix at time t. Here, the observation model minimizes the error when erroneous data are produced due to sensor failure. Now, the obstacle distance is observed through the measurement matrix defined through measurement vector z and measurement matrix h as given in Equation (7). The state and measurement predictions are associated with each other to resolve the origin uncertainty.

$$\hat{z}(t) = h(t, \hat{x}(t+1 \mid t)) \tag{6}$$

$$\hat{z}(t) = \begin{bmatrix} A_{x,t} \\ A_{y,t} \end{bmatrix} \overset{where}{=} \begin{bmatrix} |\phi_{xy,t}| \cdot \sin(\theta_t) \\ |\phi_{xy,t}| \cdot \cos(\theta_t) \end{bmatrix}$$

$$\hat{s}(t) = H(t)\hat{P}(t+1 \mid t)H^T(t) \tag{6a}$$

$$z = \begin{bmatrix} x & y \end{bmatrix}^T \tag{7}$$

7. Navigating the robot: We used estimated innovation covariance and updated measurement for navigating the designed robot. Equation (8) shows the Kalman gain, Equation (9) defines the state update, and Equation (10) provides the updates in state covariance. At the correction state, $\hat{x}(t+1)$ and $P(t+1)$ are obtained from the prediction state. Here, $\hat{x}(t+1)$ is calculated by using the state transition state where $P(t+1)$ is user-defined. Next, the Kalman gain or $K(t)$ is calculated to represent the trustable value of the state model and measurement variable. Here, matrices are used to handle multiple degrees of freedom that allow the representation of linear relationships between different state variables such as position, velocity, and friction. The value of h approaching 0 indicates that the measurement variable is more trustable than the state model. However, the value of $P(t+1)$ approaching 0 indicates that the state model is more trustworthy than the measurement variable. Next, the estimated data are updated based on values of $K(t)$. Finally, the error covariance is updated for the next iteration. All the parameters are updated in each iteration. Thus, the accuracy of predicted and estimated data improves continuously using the data received from the prediction states [34].

$$K(t) = \hat{P}(t+1 \mid t)H^T(t) + \hat{s}(t)^{-1} \tag{8}$$

$$\hat{x}(t+1 \mid t+1) = \hat{x}(t+1 \mid t) + K(t)v(t) \tag{9}$$

$$\hat{P}(t+1 \mid t+1) = (I - K(t)H(t))\hat{P}(t+1 \mid t) \tag{10}$$

8. Predicting sigma Points: UKF uses UT for computing sigma points for two dimensions. It performs transformation by a non-linear density function ($y = g(x)$). This function is defined in Equation (11) [34]. In this research, we use the x and y axis for moving a robot, and consider five sigma points for each dimension. Thus, a matrix (χ) of 2×5 is obtained. These sigma points as calculated by UKF are defined in recursive Equations (11a) and (11b). Here, χ is sigma points matrix, μ is mean, n is dimension-

ality, λ is the scaling factor and $\sum$ is covariance matrix. The value of n is set at 2 for this research, χ = 2n + 1, λ = 3 − n.

$$\chi^{[0]} = \mu \tag{11}$$

$$\chi^{[i]} = \mu + (\sqrt{n+\lambda}\sum)_i \tag{11a}$$

$$\chi^{[i]} = \mu - (\sqrt{n+\lambda})_{(i-n)} \tag{11b}$$

In Equations (11)–(11b), χ are the sigma points. Here, θ, ϕ, and ψ are the Infrared, Sonar, and Friction variables in the navigation of the robot respectively, $\dot{\beta}$ is the projection of the robot's angular velocity bias. Gyroscope measurement (V_x and V_y) are velocity variables along the X-axis and Y-axis, respectively. These are the same as given in Equations (3) and (4). When the sigma points are propagated through non-linear density function $(y = g(x))$, then we first estimate the predicted mean μ' using the state–space time updation as shown in Equation (12). Next, we obtain the predicted covariance matrix, $\sum'$ as shown in Equation (13). In this equation, $\sum'$ is the predicted covariance, and g is a non-linear function. The state covariance matrix holds the uncertainty of the states and contains the covariance of each feature.

$$\mu' = \sum_{i=0}^{2n} \omega^{[i]} g(\chi^{[i]}) \tag{12}$$

$$(\Sigma)' = \sum_{i=0}^{2n} \omega^{[i]} \left(g(\chi^{[i]} - \mu') \left(g(\chi^{[i]} - \mu')^t \right) \right) \tag{13}$$

9. Updating the state of a robot: In this step, the data captured from the sensors are used to compute the Euclidean distance between the predicted and actual values of mean and covariance [34]. This is defined in Equations (14)–(14b).

$$z = h(\chi) \tag{14}$$

$$\hat{z} = \sum_{i=0}^{2n} \omega^{[i]} z^{[i]} \tag{14a}$$

$$s = \sum_{i=0}^{2n} \omega^{[i]} (z^{[i]} - \hat{z})(z^{[i]} - \hat{z})^T \tag{14b}$$

Here, z denotes the sigma points after transformation in measurement space, χ is sigma points' matrix, $\hat{z}$ is mean in measurement space, s is covariance in measurement space, and h is a function that for mapping sigma points to measurement space.

10. Predicting error: The error may arise in the prediction step as calculated in Equation (15). In this equation, T is cross-correlation matrix between state space and predicted space, s is the predicted covariance matrix, and K is Kalman Gain. Equation (15a) is like Equation (8) of EKF. Next, the final state is predicted using Equations (16) and (16a). These equations are similar to the Equations (8)–(10) defined in EKF [34]. Here, μ is mean, $\sum$ is covariance matrix, μ' is predicted mean, $\sum'$ is predicted covariance, K is Kalman gain, z is actual measurement mean, $\hat{z}$ is mean in measurement space, and T is cross correlation matrix.

$$T = \sum_{i=0}^{2n} \omega^{[i]} (\chi^{[i]} - \mu')(z^{[i]} - \hat{z})^T \tag{15}$$

$$K = T \cdot s^{-1} \tag{15a}$$

$$\mu = \mu' + K(z - \hat{z}) \tag{16}$$

$$\Sigma = (I - KT)\Sigma' \tag{16a}$$

For EKF, the Equations (2) and (3) together are named as state vector equations. These equations are important to predict the impact of modelling parameters. For example, a measurement error may be encountered due to an environmental effect on sensors. Next, Equation (4) is the surface parameter equation. It is used to estimate the motion state of a robot on varying surfaces using outcomes of state vector equations (x, x_n). Equations (5) and (5a) together are called the initial or starting state equations, respectively. These states are the predecessors of the Kalman filter. Equations 6 and 6a form optimizing state equations. These are used to measure the prediction of the model, which takes the predicted input (A_x and A_y) with sine and cosine components of the state model, to infer new measurements. Finally, we fuse Equations (8)–(10) to evaluate $C_{efficacy}$ as defined in Equation (17). Here, matrices are used to handle multiple degrees of freedom, allowing the representation of linear relationships between different state variables such as position, velocity, and friction.

$$C_{efficacy} = W_i x + W_j x_n + W_k x_0 + W_l(\hat{z}(t) + \hat{s}(t)) \tag{17}$$

In Equation (17), W is the weight of efficacy in various environments. Its value varies from 0 to 1.

For UKF, Equations (11)–(11b) be state vector equations. Here, the x and y axes are used for moving a robot and to consider five sigma points for each axis. These sigma points as calculated by UKF and defined in Equations (11a) and (11b). Further, the target tracking of a robot is performed as per Equation (13). While reaching the target, the state of the robot changes. Updating the states of a robot is defined in Equations (14) and (14a). The data used in these equations are captured from the sensors, and the Euclidean distance between the predicted and actual values of mean and covariance is calculated. Equations (13), (14) and (14a) together form optimizing state equations. Similarly, we fuse the optimizing state equations with the error predicting Equation (15) to evaluate $C_{efficacy}$, as presented in Equation (18).

$$C_{efficacy} = W_i \boldsymbol{\chi}^{[i]} + W_j x_n + W_k \hat{z}(t) + W_l(\mu' + \Sigma') \tag{18}$$

4. Results

The robot designed in this research uses three wheels for navigation, including a castor wheel. It employs a Zigbee module for communication, and SONAR for measuring distance from an object. The replacement of LIDAR with SONAR, using an Nvidia Nano Jetson, is a cost-effective solution for designing a robot with low hardware requirements. For example, a robot designed in [35] includes a 2D LIDAR Sensor, RGB-D Camera, Nvidia Jetson Nano Board, WiFi Module, Encoder Gearmotor, and McNam Wheel, a similar-purpose robot to ours in this paper, i.e., one that can move on a surface on an optimized path. The comparison of costs is shown in Table 2. It is apparent from the costs shown in Table 2 that our design saves approximately 30% of the cost of robot design without compromising functionality.

Next, the performance of the robot is evaluated on different surface types such as marble flooring, glaze tiles, rough surface, and boulder surface. Based on the experimental results shown in Supplementary Materials, we claim that the designed robot is more stable while moving on marble flooring. Also, the robot uses frictional steering instead of rack-and-pinion steering for moving on a sectorial turn when it is moved on marble flooring. This shows its stability while moving. Real-time data is acquired from sensors equipped with the robot designed as a part of this research.

The behavior of the robot is analyzed and its path has been optimized to minimize the error between the actual and predicted paths. The process of path optimization is shown in Figure 4.

Table 2. Comparison of cost involved.

Component Used [35]	Approximately Cost	Components Used in Our Design	Exact Cost
2D Lidar Sensor	INR 10,000/-	Ez 430 Chronos Watch	INR 5000/-
RGB-D Camera	INR 8000/-	IMU, SONAR,IR	INR 1300/-
Nvidia Jetson Nano Board	INR 15,000/-	Arduino Board with overhead	INR 3000/-
WiFi Module	INR 400/-	Xigbee kit with Xbee base and 1 device module	INR 3200/-
Encoder Gearmotor	INR 2000/-	DC Motor with module	INR 510/-
McNam Wheel	INR 6000/- (100 mm)	Castor wheel with 2 rubber wheels	INR 600/-
Total Cost (Approximately)	INR 41,400	Total cost	INR 13,610/-

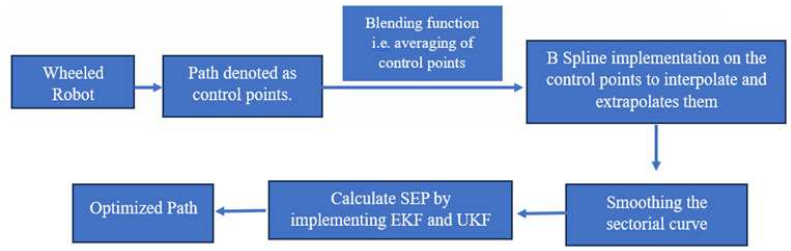

Figure 4. Workflow for path optimization.

The objective of this paper is to design a robot that shows minimum deviation from the actual path. Therefore, it is not significant to evaluate the impact of eight modelling parameters such as A_x, A_y, A_z, distance (ϕ) of an object from the robot, and the change in the state of a moving robot per unit time (θ) individually. Rather, it is important to demonstrate their cumulative impact on the path of the robot. The impact of these parameters is illustrated in Equations (3), (5a), (6) and (13). Here, Equation (3) determines the impact of distance (ϕ), and change in state (θ). Equation (5a) is employed to predict the state of a robot. Equation (6) illustrates the impact of dynamic acceleration on the path of a robot with sin and cosine components. Equation (13) shows the impact of orientation on the path of a robot. The cumulative impact of these parameters in terms of actual path predicted path, and optimized path is demonstrated in Figures 5 and 6.

Further, a comparison between the actual, predicted, and optimized path of the robot using the EKF and UKF algorithms is performed. The comparative analysis is demonstrated in Figures 4 and 5.

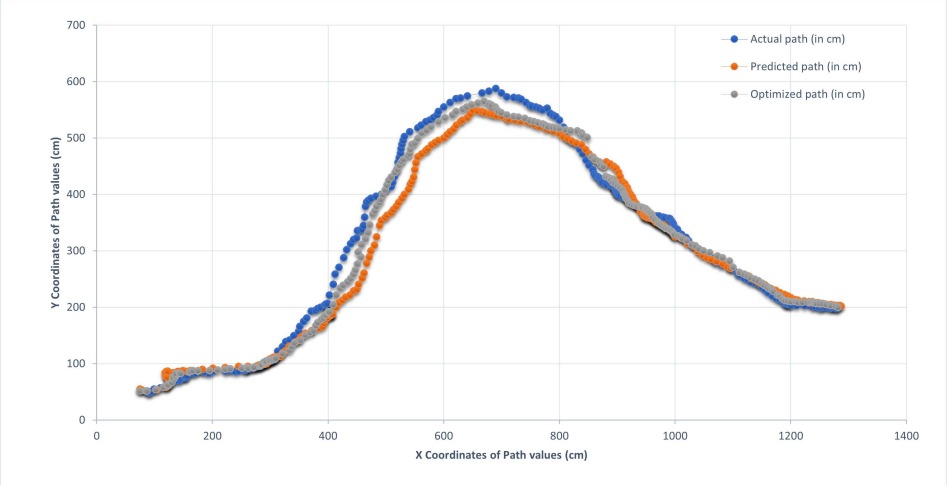

Figure 5. Path optimization of the robot using EKF.

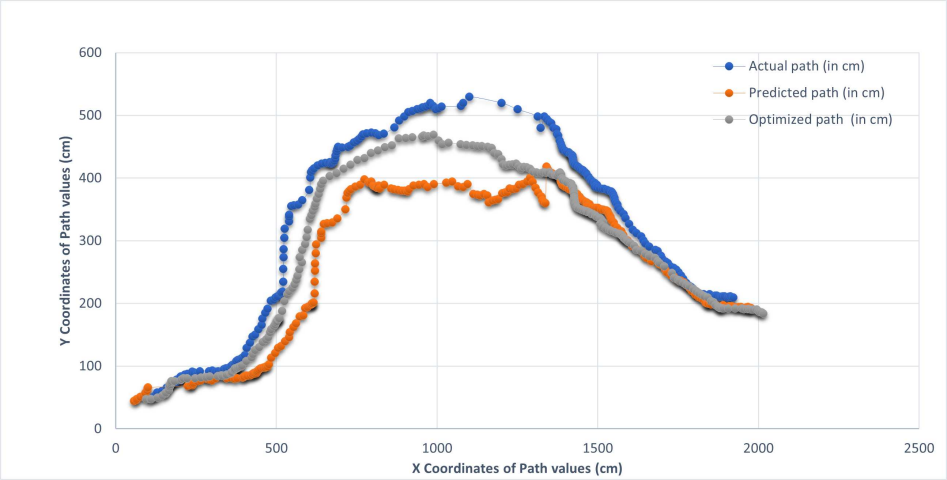

Figure 6. Path optimization of the robot using UKF.

It is evident from the results shown in Figure 5, that there is a deviation of approximately 1.65 cm in the actual and predicted pose, and 1.125 cm between the actual and optimized paths when EKF was applied, whereas Figure 6 shows the higher deviations of 3.11 cm between the actual and predicted pose, and 3.51 cm between the actual and optimized pose when UKF was applied. This is due to the presence of continuous control points that interpolate the endpoints and approximate the remaining two points.

Sectorial-Based Analysis of Predicted, Optimized, and Actual Paths

Sectorial-based analysis involves examining the path of a robot inside a particular sector of the environment. It is carried out to evaluate the precision, efficacy, and efficiency of various path-planning and control systems. For example, initially, the environment is divided into sectors such as 0° to 15°; 15° to 30°; 30° to 45°; 45° to 90°; 90° to 120°; or 120° to 180°. The pathways of a robot are assessed inside these sectors. It gives more insights into how EKF and UKF algorithms work in various environments. Here, the predicted path is the path estimated by predictive models such as EKF and UKF using the robot's current state and its motion dynamics. The next state of the robot depends on both its current state and the current measurements. Here, the current state represents the state variables, i.e., A_x, A_y, A_z as the dynamic acceleration of the robot in the X, Y, and Z axes, respectively; G_x, G_y, and G_z to denote the orientation of the robot in motion in the X, Y, and Z axes, respectively; distance (ϕ) of object from the robot measured using SONAR; and motion (θ) of the robot detected using INFRARED. The current measurements include the data obtained from sensors, which provide information about the robot's external environment in which it is deployed. Current measurements collect data from the surroundings and motion of a robot. Measurement errors may be encountered due to environmental effects on the sensors. The impact of environment is calculated using Equations (17) and (18). So, it is crucial for the robot to make decisions solely based on the current state. Rather, it must consider diverse environmental factors as well as its current state before making a decision for navigating on a path. The path is expected to be followed by a robot in various environments. The optimized path is generated by interpolating and extrapolating the control points as demonstrated in Figure 7. The blending function is used to optimize the path on which the robot navigates. The actual path is the robot's trajectory when it moves in real life through its surroundings. This is determined by the environment, sensor inputs, and control actions.

Further, we calculated the sectorial turn of the robot and evaluated the impact of friction on the movement of the robot. From the experiments, it has been observed that frictional steering leads to the non-linear movement of the robot. Therefore, the EKF algorithm gives more promising results than UKF. UKF shows more deviation in the predicted and optimized paths with respect to the actual pose of the robot. Now, the robot is moved on six different

curve sectors. EKF and UKF are applied separately. The Euclidean distances for different sectorial curves such as 0° to 15°; 15° to 30°; 30° to 45°; 45° to 90°; 90° to 120°; or 120° to 180° are calculated. The obtained results are demonstrated in Figures 5, 6 and 8–13. The values reported by UKF and EKF at the above-mentioned angles are illustrated in Tables 3 and 4, respectively.

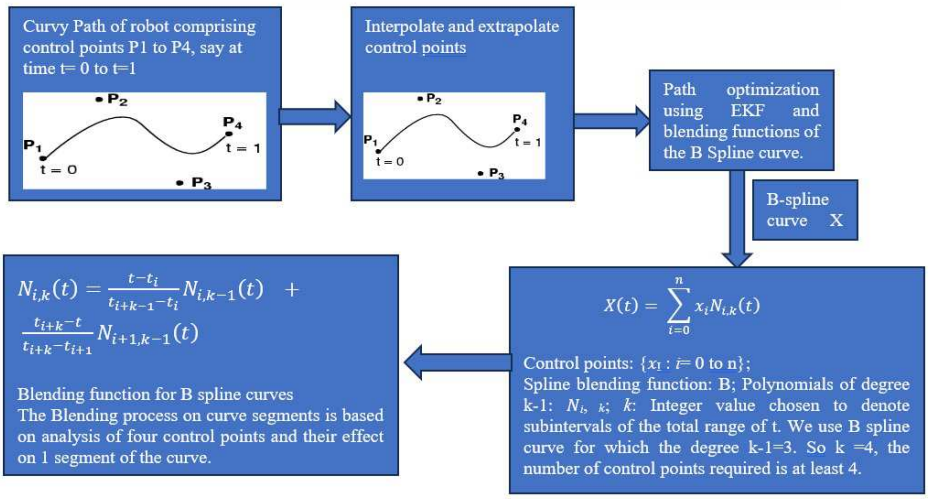

Figure 7. Optimized path by interpolating and extrapolating the control points.

Table 3. Applicability of UKF algorithm on curve sector observation points (all distances (actual, predicted, and optimized) are measured in cm).

Observation Points	Actual Distance	Predicted Distance	Optimized Distance	Actual-Predicted	Actual-Optimized	Actual-Predict/ Actual (%)	Actual-Optimized/ Actual (%)
1	482.8	731	580	−249	−97	−51.4872	−20.1326
2	600	643	666	−42	−66	−7.05814	−11.0245
3	179	205	258	−26	−79	−14.2586	−44.2212
4	480	256	239	224	241	46.68958	50.20833
5	532	644	586	−112	−54	−21.0526	−10.1504
6	130	153	150	−23	−20	−9.44104	−3.14521
Sum	2404.29	2631.28	2479.91			−9.43467	−6.41092

Table 4. Applicability of EKF algorithm on curve sector observation points (all distances (actual, predicted, and optimized) are measured in cm).

Observation Points	Actual Distance	Predicted Distance	Optimized Distance	Actual-Predicted	Actual-Optimized	Actual-Predict/ Actual (%)	Actual-Optimized/ Actual (%)
1	380	623	470	−243	−90	−64.0361	−23.658
2	562	504	512	59	50	10.41694	8.929528
3	149	106	145	43	4	28.67205	2.670958
4	205	201	168	3	37	1.696848	18.09336
5	434	457	484	−23	−50	−5.26495	−11.4537
6	101	121	99	−20	3	−19.3412	2.653122
Sum	1935	1936	1827			−797606	−0.46079

It is evident from the results demonstrated in Figure 8 that the deviation in path between the actual and predicted paths is reduced by applying UKF rather than EKF. The deviation observed between the actual and predicted path is observed as 51.48%, whereas

the value is reduced to 20.13% between the actual and optimized paths. This shows that the UKF is more effective in path optimization when robots are navigated from 0° to 15°.

It is apparent from the results shown in Figure 9 that the predicted path by applying UKF is optimized with continuous control points and directed at an approximately linear path. UKF reports a deviation of 7.05% between the actual and predicted path. Whereas the value is 11.02% between the actual and optimized paths when a robot is navigated from 15° to 30°.

It is apparent from the results demonstrated in Figure 10 that EKF is more effective in path optimization when a robot is navigated from 30° to 45°. The high acute angle promotes more blending of control points at the ends than control points in the center. Here, the EKF reports a deviation of 2.67% between actual and optimized paths.

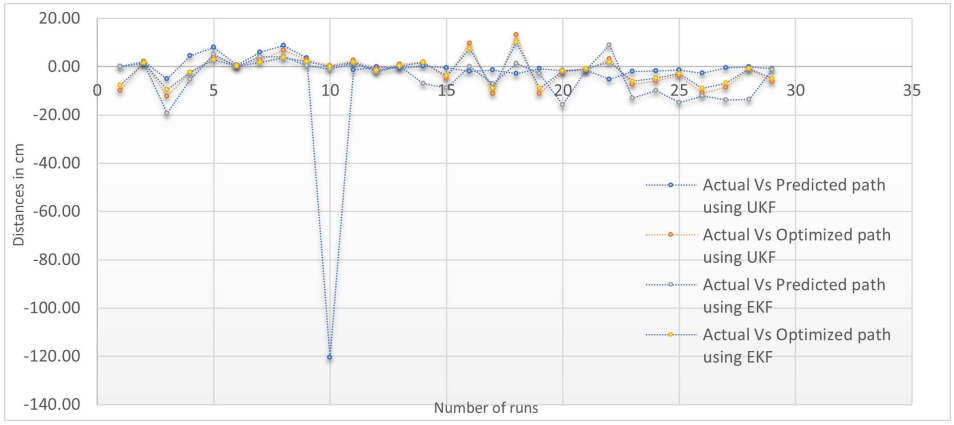

Figure 8. Euclidean difference between two consecutive values of curve sector-based observation points Group I (0° to 15°).

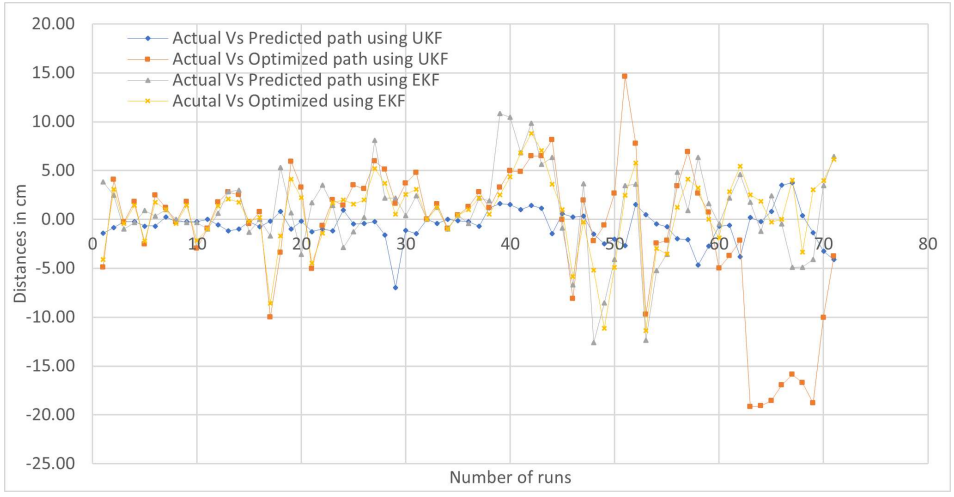

Figure 9. Euclidean difference between two consecutive values of curve sector-based observation points Group II (15° to 30°).

Similarly, it is clear from the results displayed in Figure 11 that EKF improves the learning behavior of the robot while navigating on the sectorial turn from 45° to 90°. EKF reports a lower deviation than UKF in the actual and predicted paths. The deviation observed in the actual and predicted paths is 1.69%, whereas the value is 18.09% when observed between the actual and optimized paths. This shows that the EKF is more effective in path optimization when the robot is navigated from 45° to 90°.

Further, it is observed from the results given in Figure 12 that the minimum deviation of 5.26% is observed in the actual and predicted path when EKF is applied. A higher value of 10.15% is reported between the actual and optimized paths when UKF is applied. This

shows that the EKF is more effective in path optimization when the robot is navigated from 90° to 120°.

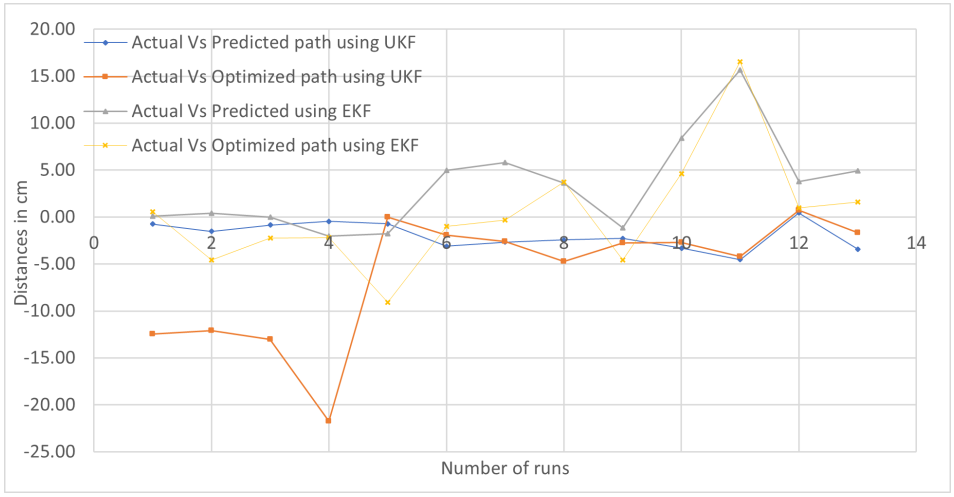

Figure 10. Euclidean difference between 2 consecutive values of curve sector-based observation points Group III (30° to 45°).

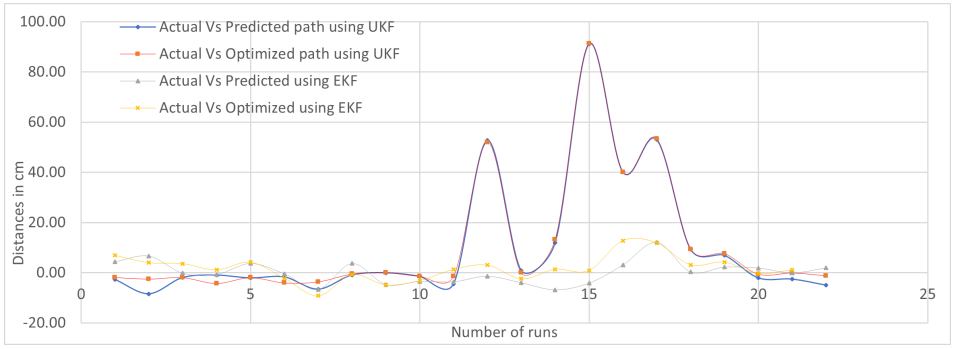

Figure 11. Euclidean difference between 2 consecutive values of curve sector-based observation points Group IV (45° to 90°).

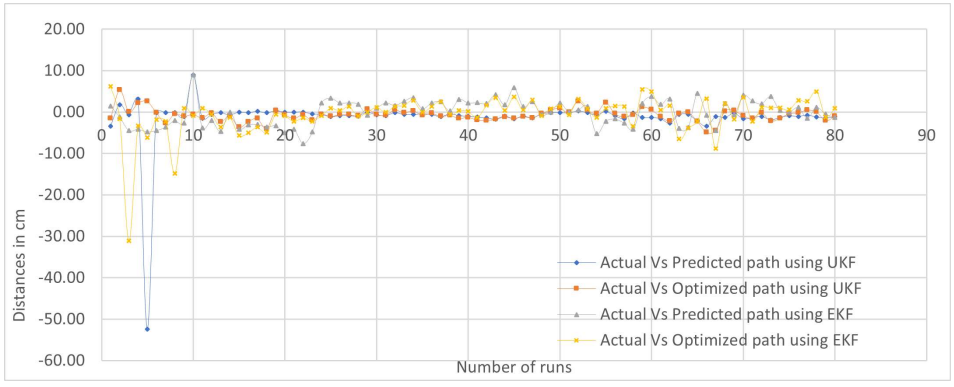

Figure 12. Euclidean difference between 2 consecutive values of curve sector-based observation points Group V (90° to 120°).

Similarly, it is obvious from the results shown in Figure 13 that the deviation observed between the actual and predicted paths is 9.44%, whereas the value is reduced to 2.65% between the actual and optimized paths when EKF is applied. This shows that the EKF is more effective at path optimization when the robot is navigated from 120° to 180°. The differences between the actual, predicted, and optimized paths were estimated on the tangent of SEP observation points viz. I, II, III, IV, V, and VI when the robot moves on

a sectorial curve motion. It is observed from Table 5 that the average deviation of the predicted path and the optimized path from the actual path of a robot is 1.6520 cm and 1.125 cm, respectively when EKF was employed. However, the deviation of the predicted and optimized paths from the actual path increases to 3.1109 cm and 3.5193 cm respectively, when UKF was employed.

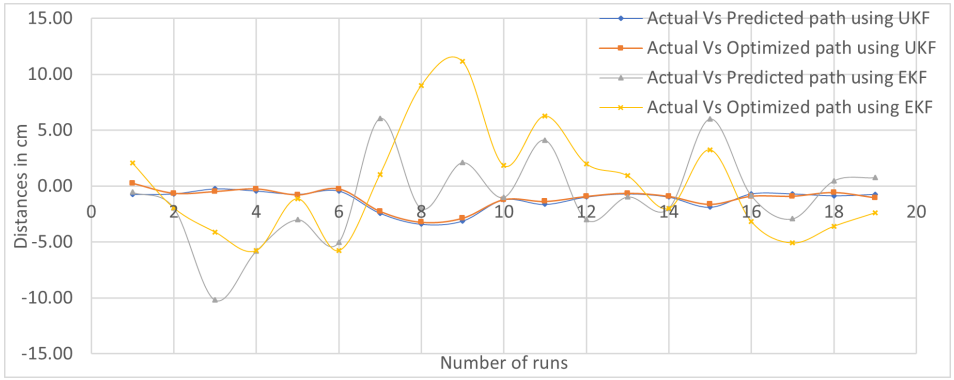

Figure 13. Euclidean difference between two consecutive values of curve Sector-based observation points Group VI (120° to 180°).

Table 5. Deviation in actual pose, predicted and optimized path using EKF and UKF at SEP observation points.

S.No.	Deviation in Paths	Key I (cm)	Key II (cm)	Key III (cm)	Key IV (cm)	Key V (cm)	Key VI (cm)	Average SEP Observation Points
1.	EKF (Actual vs. Predicted)	−4.19	0.83	3.28	0.16	−0.12	1.33	1.65
2.	EKF (Actual vs. Optimized)	−1.55	0.71	0.31	1.69	−0.62	1.88	1.13
3.	UKF (Actual vs. Predicted)	−4.12	−0.60	−1.97	10.22	−1.28	0.48	3.11
4.	UKF (Actual vs. Optimized)	−1.68	−0.93	−6.10	10.98	−0.68	0.75	3.52

The extended Kalman filter is a powerful tool for estimating the state of moderately non-linear systems that use matrix operations in real-time. Hence, it can be easily implemented on our designed platform. We are working with a low-power Arduino Uno microcontroller (ATmega328 (Atmel, Jaipur, Rajasthan) and limited computational resources. EKF can work efficiently even with low computational resources. EKF reports a better performance than UKF on the non-linear data generated due to frictional steering. Also, EKF is computationally less expensive than the UKF due to the smaller number of parameters in its robotic system and simple non-linearities. As our system supports the Exo-GPU architecture with limited computational resources, EKF is a more practical choice. The EKF-based SLAM mathematical model linearizes the path on two control points. Moreover, the coefficient employed in EKF shows better fusion due to the higher compatibility with the Kalman filter when the robot moves in various environments.

5. Discussion

This research focuses on improving the movement of a robot on both linear and curvy paths. To achieve this, a three-wheeled robot was designed, with two traditional wheels and a caster wheel. The caster wheel allows the robot to move easily in any direction and maneuver heavy objects. The robot includes sensors such as IMU, Infrared, SONAR,

and Xbee module for wireless transmission. These sensors are more cost-effective than visual sensors. The use of an ez430 Chronos watch as an accelerometer mode extends the robot's applicability for covering long distances. With the optimum use of sensors and mathematical modeling of EKF and UKF algorithms, the robot achieved a small deviation of 1.125 cm from the predicted optimized path. The use of frictional steering instead of rack-and-pinion steering is important for moving the robot on a sectorial turn and measuring SEP. In contrast to the four-wheeled robots with complex structures and high steering control requirements, we propose a low-cost, efficient two-wheeled robot design with a caster wheel. For example, the works in [11] applied SLAM with the Improved Schmidt Orthogonal UKF algorithm on Cark Park dataset to validate the efficacy of their design. The authors in [36] employed trajectory tracking control for wheeled mobile robots with kinematic parameter uncertainty. Similarly, the authors in [37] applied UKF for estimating the precise position of a robot.

The above-stated works in literature have applied sensor fusion algorithms for precise robot positioning, but they lack in measuring SEP for real-time datasets. The Mobile robot localization via EKF and UKF applied on a real-life dataset precisely estimates the path of a robot on a linear path. The comparative analysis of the 8-DOF, 6-DOF, and 4-DOF robot designs is elaborated in Table 6. It is evident from Table 6 that robots designed with 4-DOF and 6-DOF are similar in surface type used for movement, grip on surface, and remote navigation, but differ from robots designed with 8-DOF on all these parameters. Moreover, the robots designed with 4-DOF, 6-DOF, and 8-DOF differ from each other in path deviation. The robot with 4-DOF shows a deviation of 3.965 cm from the path is observed when it is moved on a U-shaped pathway and the Kalman filter is applied, whereas the robot with 6-DOF shows a deviation of 3.642 cm from the path. The robot with 8-DOF presents a minimum deviation of 1.125 cm between the actual and optimized position when EKF is applied.

Table 6. Comparative analysis of robot designs with different degree of freedom.

Parameters	4-DoF and 6-DoF	8-DoF
Diverse Surfaces	The wheel profile of the Turtlebot has independent spring suspension. Thus, the height of two wheels varies when it is moved on pebbled surface. This may lead to misbalancing and falling of Turtlebot.	The wheel profile of the designed robot makes it effective to move on pebbled surface [7]
Grip on surface	The presence of a single cut in the center leads to reduced grip (μ) on the surface, making it challenging to control the speed of both wheels while moving on that surface. μ = F_friction/ F_normal Here, F_friction represents the friction between a wheel and a surface, while F_normal denotes the force that acts perpendicular to the surface on which the wheel is positioned.	The wheels have more than four nearby thin cuts that improve grip on various surfaces. The wheel profile is designed without suspension, making it rigid and fixed on a round plate. Additionally, each wheel generates different power, enabling the robot to maneuver effectively on diverse surfaces. The wheels individually can generate power. So, we can control their speeds independently.
Remote navigation	Turtlebot does not have built in remote area network (RAN). Its' external integration may lead to battery and power loss issues.	Our wheeled robot supports APIs-based communication with the hardware using built in RAN.

It is apparent from the comparison that the robot design with 8-DOF has a higher grip, is effective in moving on diverse surface types, and shows low deviation from the planned path. It also provides flexibility to control from a remote location without any external attachment. Further, the algorithms proposed in [20] differ in number of control points and SEP reported. The comparison of these parameters is demonstrated in Table 7. It is clear

from the comparison that the proposed robot design reports the lowest value of SEP and has a low number of control points.

Thus, it can be used in medicine dispensing, agriculture, healthcare, technological assistance for elders, smart warehouses, and luggage management. Its applications can be further extended by using edge computing devices. Integrating edge computing devices near the source data may also prove useful in optimizing data transmission, real-time data analysis, and path optimization. Also, the use of robots for optimized path planning has numerous applications in different domains. For example, optimized path planning can help to reduce waste and increase efficiency by finding the most efficient path for the robot while harvesting, spraying, and planting in agriculture. Optimized path planning can improve productivity by finding the most efficient path for the robot to follow while navigating through the warehouse or distribution center. Optimized path planning also finds the most efficient path for the robot to follow during surgery, rehabilitation, and patient care. Hence, it reduces the risk of human error. Moreover, optimized path planning is helpful in reducing the risk of damage to the robot and improves the efficiency of data collection by finding the most efficient path for the robot to follow while navigating in challenging environments.

Table 7. Comparison in control points and SEP of state-of-the art algorithms and proposed method.

Parameters	Curve SLAM	Open Key Frame-Based Visual Inertial SLAM (OKVIS)	Stereo Version of the Parallel Tracking and Mapping (SPTAM)	Proposed Method
Control points	220	345,340	138,403	240
SEP	2.57 m	4.68 m	2.58 m	1.652 cm Actual and predicted position of robot 1.125 cm Actual and optimized position of robot

6. Conclusions

In this manuscript, we designed a low-cost robot with low hardware requirements. We successfully optimized its 8-DOF path by interpolating and extrapolating the control points. Our experiments show that SLAM-based EKF and UKF can estimate and reduce path prediction errors in sectorial curve motion. The accuracy of a robot's path depends on the actual, predicted, and optimized paths. With the EKF applied, we observed an average deviation of 1.6520 cm and 1.125 cm for the predicted and optimized paths, respectively, from the actual path. However, when UKF was employed, the deviation of the predicted and optimized paths from the actual path increased to 3.1109 cm and 3.5193 cm, respectively. Therefore, EKF gives more promising results than UKF in path optimization. Based on our experiments on a curvy path, we concluded that the UKF shows 31.35% less deviation than the EKF when a robot moves from 0° to 15°. Additionally, UKF provides a more optimized path when a robot moves from 15 degrees to 30° on a sectorial path. When the robot moves at various angles, such as 30° to 180°with a gap of 15°, EKF reports a minimum deviation from the actual path. The proposed design and algorithms are efficient in moving the terrain on various surfaces such as boulder tile and pebbled surfaces, with auto-correction of the predicted path.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/computers12120250/s1.

Author Contributions: Conceptualization, R.M., G.R., V.P., V.S.D., E.V. and E.Z.; Methodology, R.M., G.R., S.C., E.V. and E.Z.; Software, R.M., G.R., S.C., E.V. and E.Z.; Validation, R.M., G.R., V.P., V.S.D., E.V. and E.Z.; Formal analysis, R.M., G.R., V.S.D., E.V. and E.Z.; Investigation, R.M., G.R., E.V. and E.Z.; Resources, G.R.; Data curation, R.M., G.R. and S.C.; Writing—original draft, R.M.; Writing—review & editing, R.M., G.R., V.P., V.S.D., E.V. and E.Z.; Supervision, G.R., V.S.D., E.V. and E.Z.; Project administration, G.R., V.P., V.S.D., E.V. and E.Z.; Funding acquisition, E.Z. All authors contributed

equally to the study, conception, design and preparation of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Department of Informatics, Modeling, Electronics and Systems (DIMES), University of Calabria Grant/Award Number: SIMPATICO ZUMPANO.

Data Availability Statement: Data are contained within the article.

Acknowledgments: The authors would like to acknowledge the Manipal University Jaipur for providing the Computer lab with a high-end GPU to complete this research.

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

References

1. Matharu, P.S.; Ghadge, A.A.; Almubarak, Y.; Tadesse, Y. Jelly-Z: Twisted and coiled polymer muscle actuated jellyfish robot for environmental monitoring. *Acta IMEKO* **2022**, *11*, 1–7. [CrossRef]
2. De Fazio, R.; Katamba, D.M.; Ekuakille, A.L.; Ferreira, M.J.; Kidiamboko, S.; Giannoccaro, N.I.; Velazquez, R.; Visconti, P. Sensors-based mobile robot for harsh environments: Functionalities, energy consumption analysis and characterization. *Acta IMEKO* **2021**, *10*, 29. [CrossRef]
3. Goll, S.; Borisov, A. Interactive model of magnetic field reconstruction stand for mobile robot navigation algorithms debugging which use magnetometer data. *Acta IMEKO* **2019**, *8*, 47–53. [CrossRef]
4. Goll, S.; Zakharova, E. An active beacon-based leader vehicle tracking system. *Acta IMEKO* **2019**, *8*, 33–40. [CrossRef]
5. Li, G.; Qin, D.; Ju, H. Localization of wheeled mobile robot based on extended Kalman filtering. In Proceedings of the MATEC Web of Conferences, Singapore, 15–16 September 2015; p. 01061.
6. Mittal, R.; Pathak, V.; Mishra, N.; Mithal, A. Localization and Impulse Analysis of Experimental Bot Using eZ430-Chronos. In *MARC 2018, Proceedings of the Applications of Computing, Automation and Wireless Systems in Electrical Engineering, Delhi, India, 19–20 July 2018*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 711–722.
7. Mittal, R.; Mishra, N.; Pathak, V. Exploration of Designed Robot in Non Linear Movement Using EKF SLAM and Its Stability on Loose Concrete Surface. *Int. J. Eng. Adv. Technol.* **2019**, *9*, 875–878. [CrossRef]
8. Apriaskar, E.; Nugraha, Y.P.; Trilaksono, B.R. Simulation of simultaneous localization and mapping using hexacopter and RGBD camera. In Proceedings of the 2017 2nd International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT), Jakarta, Indonesia, 23–24 October 2017; pp. 48–53.
9. Valade, A.; Acco, P.; Grabolosa, P.; Fourniols, J.Y. A study about Kalman filters applied to embedded sensors. *Sensors* **2017**, *17*, 2810. [CrossRef]
10. Wan, E.A.; Van Der Merwe, R. The unscented Kalman filter for nonlinear estimation. In Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373), Lake Louise, AB, Canada, 4 October 2000; pp. 153–158.
11. Tang, M.; Chen, Z.; Yin, F. SLAM with Improved Schmidt Orthogonal Unscented Kalman Filter. *Int. J. Control Autom. Syst.* **2022**, *20*, 1327–1335. [CrossRef]
12. Chen, S.; Zhou, W.; Yang, A.-S.; Chen, H.; Li, B.; Wen, C.-Y. An end-to-end UAV simulation platform for visual SLAM and navigation. *Aerospace* **2022**, *9*, 48. [CrossRef]
13. Mittal, R.; Pathak, V.; Gandhi, G.C.; Mithal, A.; Lakhwani, K. Application of machine learning in SLAM algorithms. *Mach. Learn. Sustain. Dev.* **2021**, *9*, 147.
14. Nüchter, A.; Lingemann, K.; Hertzberg, J.; Surmann, H. 6D SLAM—3D mapping outdoor environments. *J. Field Robot.* **2007**, *24*, 699–722. [CrossRef]
15. Mittal, R.; Pathak, V.; Gandhi, G.C.; Mishra, N. Effect of Curve Algorithms on Standard SLAM algorithm for Prediction of Path. *Test Eng. Manag.* **2020**, *83*, 1196–1201.
16. Liu, M.; Huang, S.; Dissanayake, G. Feature based SLAM using laser sensor data with maximized information usage. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1811–1816.
17. Kysar, S.; Young, P.; Kurup, A.; Bos, J. C-SLAM: Six degrees of freedom point cloud mapping for any environment. In Proceeding of the Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure, Online, 27 April–9 May 2020; pp. 98–112. [CrossRef]
18. Dhiman, N.K.; Deodhare, D.; Khemani, D. Where am I? Creating spatial awareness in unmanned ground robots using SLAM: A survey. *Sadhana* **2015**, *40*, 1385–1433. [CrossRef]
19. Dryanovski, I.; Morris, W.; Kaushik, R.; Xiao, J. Real-time pose estimation with RGB-D camera. In Proceedings of the 2012 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), Hamburg, Germany, 13–15 September 2012; pp. 13–20.
20. Meier, K.; Chung, S.J.; Hutchinson, S. Visual-inertial curve simultaneous localization and mapping: Creating a sparse structured world without feature points. *J. Field Robot.* **2018**, *35*, 516–544. [CrossRef]
21. Molina Martel, F.; Sidorenko, J.; Bodensteiner, C.; Arens, M.; Hugentobler, U. Unique 4-DOF relative pose estimation with six distances for UWB/V-SLAM-based devices. *Sensors* **2019**, *19*, 4366. [CrossRef]

22. You, W.; Li, F.; Liao, L.; Huang, M. Data fusion of UWB and IMU based on unscented Kalman filter for indoor localization of quadrotor UAV. *IEEE Access* **2020**, *8*, 64971–64981. [CrossRef]
23. Zhang, P.; Li, R.; Shi, Y.; He, L. Indoor navigation for quadrotor using rgb-d camera. In Proceedings of the 2018 Chinese Intelligent Systems Conference, Wenzhou, China, 11 January 2019; Springer: Singapore, 2019; Volume 2, pp. 497–506.
24. Mittal, R.; Pathak, V.; Goyal, S.; Mithal, A. GP-GPU. In *Recent Trends in Communication and Intelligent Systems: Proceedings of ICRTCIS 2019, Jaipur, India, 22–23 October 2021*; Springer: Berlin/Heidelberg, Germany, 2020; p. 157.
25. Fink, A.; Lange, J.; Beikirch, H. Radio-based indoor localization using the eZ430-Chronos platform. In Proceedings of the 2012 IEEE 1st International Symposium on Wireless Systems (IDAACS-SWS), Offenburg, Germany, 20–21 September 2012; pp. 19–22.
26. Fuentes-Pacheco, J.; Ruiz-Ascencio, J.; Rendón-Mancha, J.M. Visual simultaneous localization and mapping: A survey. *Artif. Intell. Rev.* **2015**, *43*, 55–81. [CrossRef]
27. Sakenas, V.; Kosuchinas, O.; Pfingsthorn, M.; Birk, A. Extraction of semantic floor plans from 3D point cloud maps. In Proceedings of the 2007 IEEE International Workshop on Safety, Security and Rescue Robotics, Rome, Italy, 27–29 September 2007; pp. 1–6.
28. Martinez, A.M.; Kak, A.C. Pca versus lda. *IEEE Trans. Pattern Anal. Mach. Intell.* **2001**, *23*, 228–233. [CrossRef]
29. Viola, P.; Jones, M. Rapid object detection using a boosted cascade of simple features. In Proceeding of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Kauai, HI, USA, 8–14 December 2001.
30. Bonato, V.; Marques, E.; Constantinides, G.A. A floating-point extended kalman filter implementation for autonomous mobile robots. *J. Signal Process. Syst.* **2009**, *56*, 41–50. [CrossRef]
31. Mittal, R.; Mishra, N.; Pathak, V. A review of robotics through cloud computing. *J. Adv. Robot.* **2018**, *5*, 1–11.
32. Briand, T.; Davy, A. Optimization of image B-spline interpolation for GPU architectures. *Image Process. Line* **2019**, *9*, 183–204. [CrossRef]
33. Dorj, B.; Hossain, S.; Lee, D.J. Highly curved lane detection algorithms based on Kalman filter. *Appl. Sci.* **2020**, *10*, 2372. [CrossRef]
34. Yang, C.; Shi, W.; Chen, W. Comparison of unscented and extended Kalman filters with application in vehicle navigation. *J. Navig.* **2017**, *70*, 411–431. [CrossRef]
35. Tian, C.; Liu, H.; Liu, Z.; Li, H.; Wang, Y. Research on Multi-Sensor Fusion SLAM Algorithm Based on Improved Gmapping. *IEEE Access* **2023**, *11*, 13690–13703. [CrossRef]
36. Bai, J.; Du, J.; Li, T.; Chen, Y. Trajectory tracking control for wheeled mobile robots with kinematic parameter uncertainty. *Int. J. Control Autom. Syst.* **2022**, *20*, 1632–1639. [CrossRef]
37. Naab, C.; Zheng, Z. Application of the unscented Kalman filter in position estimation a case study on a robot for precise positioning. *Robot. Auton. Syst.* **2022**, *147*, 103904. [CrossRef]