sensors

Article

Zonotopic Linear Parameter Varying SLAM Applied to Autonomous Vehicles

Marc Facerias [1], Vicenç Puig [2,*] and Eugenio Alcala [2]

[1] Autonomous Systems, Department of Electrical and Electronic Engineering, University of Manchester, Sackville Street Building, Manchester M1 3BB, UK; marc.faceriaspelegri@postgrad.manchester.ac.uk

[2] Institut de Robòtica i Informàtica Industrial (CSIC-UPC), Llorens i Artigas 4–6, 08028 Barcelona, Spain; eugenio.alcala@upc.edu

* Correspondence: vicenc.puig@upc.edu

Abstract: This article presents an approach to address the problem of localisation within the autonomous driving framework. In particular, this work takes advantage of the properties of polytopic Linear Parameter Varying (LPV) systems and set-based methodologies applied to Kalman filters to precisely locate both a set of landmarks and the vehicle itself. Using these techniques, we present an alternative approach to localisation algorithms that relies on the use of zonotopes to provide a guaranteed estimation of the states of the vehicle and its surroundings, which does not depend on any assumption of the noise nature other than its limits. LPV theory is used to model the dynamics of the vehicle and implement both an LPV-model predictive controller and a Zonotopic Kalman filter that allow localisation and navigation of the robot. The control and estimation scheme is validated in simulation using the Robotic Operating System (ROS) framework, where its effectiveness is demonstrated.

Keywords: autonomous driving; LPV modelling; optimal estimation; interval methods

check for updates

Citation: Facerías, M.; Puig, V.; Alcala, E. Zonotopic Linear Parameter Varying SLAM Applied to Autonomous Vehicles. *Sensors* **2022**, *22*, 3672. https://doi.org/10.3390/s22103672

Academic Editors: Michał R. Nowicki, Giorgio Grisetti and Marco Camurri

Received: 31 March 2022
Accepted: 5 May 2022
Published: 11 May 2022

1. Introduction

In the last few years, there has been a strong development in the automotive area towards making cars autonomous. A vast number of lines of research can be found, covering the perception of the environment to control strategies that drive the car through a given environment. In this work, the autonomous driving problem is narrowed, focusing on the localisation of the vehicle. From the control perspective, one of the most interesting techniques in autonomous driving is Model Predictive Control (MPC). We aim to use this technique and complement it with an appropriate localisation algorithm; thus, the work from Alcalá et al. [1], a state-of-the-art LPV-MPC, was used as the control technique. This decision was motivated by the promising results obtained in the past with the mentioned control technique, which are enhanced through the application of LPV models, allowing solving the optimisation problem in a linear manner.

The purpose of this paper is to enhance the reliability of localisation algorithms by means of introducing interval calculus in the estimation, motivating the usage of Zonotopic Kalman Filters (ZKF), as presented in Combastel [2], which were extended to the LPV framework in this work. Similar techniques have already been explored; for instance, Yu et al. [3] used ellipsoids instead of zonotopes, proving an enhanced performance with respect to classical Kalman filters. It can be seen that the main advantage of this line of work is that by dealing with both noise and disturbances in an interval manner, there is no need to make any assumption regarding their nature, which leads to systems that behave in a guaranteed manner as long as they are bounded, this being a more realistic restriction to meet in real applications. To our knowledge, this is the first study using LPV and dynamic models in a set-based manner within the autonomous driving localisation problem.

This paper presents a control estimation architecture for solving the autonomous driving problem in an unknown environment, taking advantage of optimal control theory. As the purpose of this research does not include the development of an exploration algorithm, the proposed scheme is integrated with a spline-based path planning module, presented in Alcalá et al. [1] and tested in a simulated scenario using ROS.

2. Related Work

The localisation problem has been widely studied within the robotics field, and it is still an ongoing problem. However, it is worth noting that the main focus of the current research is on solving the feature detection, while applying well-known Kalman filters, Monte Carlo methods, or some of their variations, as presented in Singandhupe and La [4]. These techniques can be used with a wide variety of sensor arrays; for instance, Ramesh et al. [5] relied on using point cloud data to perform Simultaneous Localisation And Mapping (SLAM), while Bhamidipati and Gao [6] merged the information from both the camera and GPS, using zonotopes to enhance the robustness of the estimation. In another line of work, we can see that other research works aimed to isolate the localisation problem and used previously computed maps to simplify the problem. For instance, Wan et al. [7] used Kalman filters to fuse the data from GNSS, LiDAR, and an inboard sensor rig to locate the vehicle in a known environment.

This research was performed from a control systems point of view; thus, all the considerations related to sensing and computer vision are considered solved, even though they are under development. As presented in Cadena et al. [8], most of the localisation techniques implemented in the state-of-the-art approaches heavily rely on probabilistic Kalman filters or their variations, some of the most-popular implementations being the Extended Kalman filter (EKF), as e.g., the one proposed in Paz et al. [9], or FastSLAM, as for instance, the work of Roh et al. [10], which relied on a combination between Monte Carlo sampling methods and the EKF. Another research line approaches the estimation problem not from the probabilistic point of view, but based on interval calculus, some examples being Mustafa et al. [11] and Fabrice et al. [12]. Moreover, an interval version of the particle filter was proposed by [13], exploring how set theory can be used to enhance the robustness of the algorithm through applying the q-satisfied intersection. More recently, we have seen how zonotopes can be applied in manipulators to reduce the inherent uncertainty of probabilistic models, leading to more consistent estimates in Li et al. [14].

In terms of addressing the inherent linearisation problem in most of the SLAM algorithms, we can see that there have been attempts at solving it with similar techniques to the LPV approach proposed in this work; for instance, Guerra et al. [15] applied a similar approach to the nonlinear kinematic model of a tricycle robot; in another line of work, Pathiranage et al. [16] used fuzzy logic to address the nonlinearity of the sensor models. On the other hand, we can see how model switching can also be used to enhance the performance of the system when facing variable noise conditions, as can be seen in [17].

3. Background Material

In this section, the preliminary knowledge required for the formulation of the set-based state estimation scheme proposed in this paper is introduced. Firstly, the basic zonotope operations are presented, and secondly, a reduction method to address dimensionality issues is explored.

3.1. Zonotopes

Zonotopes are a class of convex polytopes defined as p-dimensional hypercubes in an R^n space, formed by a centre, c_z, and a radius matrix, R_z:

$$[Z] = c_z + R_z \tag{1}$$

In the following, the most relevant properties used can be found, as presented in paper [18]:

1. The sum of two zonotopes is denoted as the Minkowski sum, and for z_1 and z_2:

$$[Z_3] = [Z_1] + [Z_2] = c_{z_3} + R_{z_3} \tag{2}$$

where

$$c_z = c_{z_1} + c_{z_2} \qquad\qquad R_{z_3} = [R_{z_1} R_{z_2}]$$

2. A unitary zonotope is defined by $R = I$ with appropriate dimensions.
3. The convex hull of a zonotope is defined as the smallest centred interval vector containing R^{nxp}.

$$R = rs(R) \tag{3}$$

where $rs()$ denotes a row sum operator, which generates a diagonal matrix, whose elements are defined by:

$$rs(R)_{ii} = \sum_{j=1}^{p} |R_{ij}| \tag{4}$$

The motivation behind using these types of sets is that their basic operations can be easily handled as simple matrix manipulation, which ensures low computational costs when operating with them.

A visual representation of a zonotope can be found in Figure 1, obtained from [19]. It can be seen that a set in the space is approximated by a polytope, represented by using both c_z and R_z, which are the central point of the polytope and its shape. In this way, we can apply the numerical tools presented in this section to propagate a given initial set under given conditions, being the robot inputs and the noise limits of the system. In this particular representation, we would expand the shape over time, generating a bounded set for each time instant k that encloses all the possible states of the system.

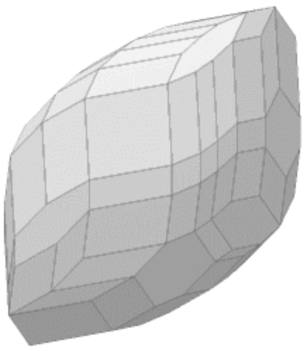
Figure 1. Graphical representation of a zonotope.

3.2. Dimensional Reduction of a Generation Matrix

As can be seen in the previous section, consecutive operations with zonotopes might lead to arbitrarily large matrices, which makes the implementation of a reduction strategy necessary. The algorithm used relies on a heuristic presented in [18] and is formalised in Algorithm 1.

Algorithm 1 Dimensional reduction of a generation matrix.

```
1: Define $d$, stating the maximal zonotope complexity
2: Sort the column vectors $R$ in decreasing order based on their euclidean norm, leading to:
       $R = [Q_1 \dots Q_{d-2} Q_{d-1} Q_d]$
3: Replace each set $[Q_{d-1} Q_d]$ by its interval hull.
```

4. Modelling

The vehicle used to implement the proposed algorithms is a 1:10 RC Car designed by the University of Berlin. It has been designed to allow an autonomous navigation and mapping by having a LiDAR, depth cameras, encoders, an IMU, and a GPS module. Furthermore, the vehicle can be actuated through a steering servo drive and a DC motor.

Due to the nature of the algorithms, a mathematical formulation of the behaviour of the system is required. In this section, the estimation model is presented. Firstly, the equations of the model are introduced, and secondly, those expressions are reformulated in an LPV manner. Note that both the vehicle and landmark behaviour are treated independently and ultimately are merged into a unique LPV model.

4.1. Nonlinear Vehicle Model

In this work, a dynamic bicycle model based on the approximation of a four-wheeled vehicle into a two-wheeled one is used, as proposed in [1]. This approach analyses the forces applied to the vehicle and derives a set of equations describing the dynamic behaviour of the system. The resulting equations are reformulated as a continuous time nonlinear model:

$$\dot{x} = f(x,u) \tag{5}$$

where the state and control vectors, respectively, are defined as

$$x = \begin{bmatrix} v_x \\ v_y \\ \omega \\ x \\ y \\ \theta \end{bmatrix}, \; u = \begin{bmatrix} \delta \\ a \end{bmatrix} \tag{6}$$

Applying the bicycle model formulation leads to:

$$\begin{aligned} \dot{x} &= v_x \cos\theta - v_y \sin\theta \\ \dot{y} &= v_x \sin\theta + v_y \cos\theta \\ \dot{\theta} &= \omega \\ \dot{v}_x &= a_{motor} - F_{df} + \frac{-F_{yf}\sin\delta}{m} + \omega v_y \\ \dot{v}_y &= \frac{F_{yf}\cos\delta + F_{yr}}{m} - \omega v_x \\ \dot{\omega} &= \frac{F_{yf} a \cos\delta - F_{yr} b}{I} \end{aligned} \tag{7}$$

where

$$\begin{aligned} F_{yf} &= C_f \alpha_f \quad \alpha_f = \delta - atan(\frac{v_y + a\omega}{v_x}) \\ F_{yr} &= C_r \alpha_r \quad \alpha_r = atan(\frac{b\omega - v_y}{v_x}) \end{aligned} \tag{8}$$

Additionally, a friction term F_{df} is introduced to model the influence of the static friction force and drag force that act to oppose the movement of the vehicle. μ, ρ, and g are the static friction coefficient, the air density at 25 °C, and the gravity, respectively. C_d is the product of the drag coefficient and vehicle cross-sectional area.

$$F_{df} = \frac{\frac{1}{2} C_d \rho_{air} A_f (v_x)^2 + \mu m g}{m} \tag{9}$$

State variables v_x, v_y, and ω represent the body frame velocities, i.e., linear in x, linear in y, and angular velocities, respectively. States x, y, and θ represent the world frame position coordinates as translations in both the x and y axis and a rotation with respect to the z axis. The control variables δ and a are the steering angle at the front wheels and the longitudinal acceleration vector on the rear wheels, respectively. F_{yf} and F_{yr} are the lateral forces produced in the front and rear tires, respectively. Front and rear slip angles are represented as α_f and α_r, respectively, and C_f and C_r are the front and rear tire stiffness coefficients. m and I represent the vehicle mass and inertia, and l_f and l_r are the distances from the vehicle centre of mass to the front and rear wheel axes, respectively. All the dynamic vehicle parameters are properly defined in Table 1.

Table 1. Dynamic model parameters of the vehicle.

Parameter	Value	Parameter	Value
l_f	0.125 m	l_r	0.125 m
m	1.98 kg	I	0.03 kg m^2
C_f	68	C_r	71
μ	0.05	ρ	1.225 kg m^3
C_{dA}	0.03 m^2	g	9.8 $\frac{\text{m}}{\text{s}^2}$

4.2. LPV Modelling of the Vehicle

An LPV model relies on redefining the expression presented in (5) by means of embedding the nonlinear nature of the equations into matrices that depend on a set of scheduling variables ϕ according to [20]:

$$\begin{aligned} x_k &= A_{robot}(\phi)x_{k-1} + B_{robot}(\phi)u_{k-1} + w_k \\ y_k &= C_{robot}(\phi)x_k + v_k \end{aligned} \tag{10}$$

ϕ being a set of variables known as scheduling variables, which modify the value of the A, B, and C matrices to adapt them to the current vehicle operating point.

This technique allows expressing the system as linear with respect to both states and control actions by embedding the nonlinearities in the system matrices. In general, the system keeps being nonlinear; however, by instantiating ϕ at a given time instant, it is possible to extend classical control techniques (as, e.g., LMIs) designed for linear systems:

$$A_{robot}(\phi) = \begin{bmatrix} A_{11} & A_{12} & A_{13} & 0 & 0 & 0 \\ 0 & A_{22} & A_{23} & 0 & 0 & 0 \\ 0 & A_{32} & A_{33} & 0 & 0 & 0 \\ \cos(\theta) & -\sin(\theta) & 0 & 0 & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \tag{11a}$$

$$B_{robot}(\phi) = \begin{bmatrix} -\frac{1}{m}\sin\delta C_f & 1 \\ \frac{1}{m}\cos\delta C_f & 0 \\ \frac{1}{I}\cos\delta C_f a & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \tag{11b}$$

being

$$A_{11} = -\mu \qquad A_{12} = \frac{C_f \sin\delta}{m v_x} \tag{11c}$$

$$A_{13} = \frac{C_f a \sin\delta}{m v_x} + v_y \qquad A_{22} = -\frac{C_r + C_f \cos\delta}{m v_x} \tag{11d}$$

$$A_{23} = -\frac{C_f a \cos\delta - C_r b}{m v_x} - v_x \qquad A_{32} = -\frac{C_f a \cos\delta - b C_r}{I v_x} \tag{11e}$$

$$A_{33} = -\frac{C_f a^2 \cos\delta + b^2 C_r}{I v_x} \tag{11f}$$

where the vector of scheduling variables ϕ is defined by a combination of states and control inputs.

$$\phi = \begin{bmatrix} v_x & v_y & cos(\theta) & sin(\theta) & \delta \end{bmatrix}^T \tag{12}$$

This formulation needs to be slightly modified to be used in further sections of this paper, as a discrete model is required. The discretisation procedure is trivial by approximating the derivative terms by their finite differences.

4.3. Landmark Modelling

In this section, the observation model of the system is derived considering that the vehicle provides us with the following measurement, m, which can be unequivocally related to the landmark i:

$$m_i = [r_i(k), \alpha_i^1(k)]^T \tag{13}$$

$r_i(k)$ being the distance between the centre of the vehicle and the landmark and α_i^j a bearing measurement. In order to simplify the definition of an observation model, these measurements are expressed as Cartesian coordinates related to the COG of the robot. This approach leads to

$$x^r_{lm_i}(k) = -x_r cos(\theta) - y_r sin(\theta) + x^w_{lm_i} cos(\theta) + y^w_{lm} sin(\theta) \tag{14a}$$

$$y^r_{lm_i}(k) = x_r sin(\theta) - y_r cos(\theta) - x^w_{lm_i} sin(\theta) + y^w_{lm} cos(\theta) \tag{14b}$$

In order to model the behaviour of the landmarks, we need to express the landmarks, $x^w_{lm_i}(k)$ and $y^w_{lm_i}(k)$, as part of a differential model, which, due to their static nature, have a zero derivative:

$$\dot{x}^w_{lm_i} = 0 \tag{15}$$

$$\dot{y}^w_{lm_i} = 0 \tag{16}$$

Finally, this formulation needs to be rewritten as an LPV model, the matrices A_{lm_i}, B_{lm_i}, and C_{lm_i} being defined considering the set of states presented in Equation (17) and N being the number of landmarks considered by the model:

$$x = [v_x, v_y, \omega, x, y, \theta, x^w_{lm_1}, y^w_{lm_1}, \dots, x^w_{lm_N}, y^w_{lm_N}] \tag{17}$$

$$A_{lm_i} = \begin{bmatrix} 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix} \tag{18}$$

$$B_{lm_i} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \tag{19}$$

$$C_{lm_i} = \begin{bmatrix} 0 & 0 & 0 & -cos(\theta) & -sin(\theta) & 0 & cos(\theta) & sin(\theta) \\ 0 & 0 & 0 & sin(\theta) & -cos(\theta) & 0 & -sin(\theta) & cos(\theta) \end{bmatrix} \tag{20}$$

4.4. LPV Modelling of the System

It is straightforward to generate a differential model, as presented in Equation (10) by considering Equations (11) and (16). The resulting model can be reformulated in an LPV manner:

$$A(\phi) = \begin{bmatrix} A_{robot} & 0 & \dots & 0 \\ 0 & A_{lm_1} & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & A_{lm_N} \end{bmatrix} \tag{21a}$$

$$B(\phi) = \begin{bmatrix} B_{robot} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{21b}$$

$$C(\phi) = \begin{bmatrix} C_{robot} & 0 & 0 & \dots & 0 \\ 0_{1,6} & C_{lm_1} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0_{1,6} & 0 & 0 & \dots & C_{lm_N} \end{bmatrix} \tag{21c}$$

ϕ being a new set of scheduling variables,

$$\phi = \begin{bmatrix} v_x & v_y & cos(\theta) & sin(\theta) & \delta \end{bmatrix}^T \tag{22}$$

Both terms w_k and v_k are related to the disturbances and sensors, respectively, and their covariances are defined by Q and R.

$$Q = \begin{bmatrix} Q_{robot} & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix} \tag{23a}$$

$$R = \begin{bmatrix} R_{robot} & 0 & \dots & 0 \\ 0 & R_{y_1} & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & R_{y_N} \end{bmatrix} \tag{23b}$$

where Q_{robot} and R_{robot} are the noise covariance matrices associated with the vehicle.

Finally, the state vector is formed by the six original states of the model and two extra states for each landmark, while the output vector has the five original states and two additional ones for each landmark:

$$x = [v_x, v_y, \omega, x, y, \theta, x^w_{lm_1}, y^w_{lm_1} \dots, x^w_{lm_N}, y^w_{lm_N}] \tag{24a}$$

$$y = [v_x, \omega, x, y, \theta, x^r_{lm_1}, y^r_{lm_1}, \dots, x^r_{lm_N}, y^r_{lm_N}] \tag{24b}$$

This formulation is dimensionally varying, as in an unknown environment, the number of observer landmarks is not fixed, which makes it unsuitable with LMI design techniques. A solution to this issue is fixing the number of landmark processes each time to one and then replacing the information depending on which landmark is observed. This assumption holds as long as we consider the vehicle position independent of the landmark measurements.

5. Localisation Algorithm

In this section, the formulation of a zonotopic observer of an uncertain discrete dynamic system is presented, including the algorithm used and its implementation in a real system.

5.1. Algorithm

The aim of using zonotopic observers is the possibility of expressing the prediction as a region in an R^{n_x} space defined by a zonotope. This region embeds all the possible states reachable by the robot given certain bounds on both measurement and system noise. In this approach, the model can be written as follows:

$$x_k = Ax_{k-1} + Bu_{k-1} + E_w w_k, \tag{25a}$$
$$y_k = Cx_k + E_v v_k \tag{25b}$$

As this is a set-based approach, both noise sources are defined as bounded by a unitary hypercube centred at the origin:

$$w = \langle 0, I_{n_w} \rangle, \tag{26a}$$
$$v = \langle 0, I_{n_v} \rangle \tag{26b}$$

Similarly, the set of states is represented using zonotopes.

$$X = \langle c_x^{io}, R_x^{io} \rangle \tag{27}$$

The equations of this observer can be derived by defining a Kalman filter where the Gaussian pdfs are replaced by zonotopic sets.

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + E_w w_k + L(y_k - \hat{y}_k) \tag{28a}$$
$$\hat{y}_k = C\hat{x}_k + E_v v_k \tag{28b}$$

Then, by substituting Equation (28a) in Equation (28b), the expression of $\hat{x}$ can be easily generated:

$$\hat{x}_{k+1} = (A - LC)\hat{x}_k + Bu_k + E_w w_k + -LE_v v_k + Ly_k \tag{29}$$

Finally, by applying the properties of zonotopic operators, the expressions of both the centre and generator matrix are defined as follows:

$$c_x^{io} = c_p^{io} + L(y_{k-1} - Cc_p^{io}), \tag{30a}$$
$$R_x^{io} = [(I - LC){R_p}^{io\downarrow} - LE_v] \tag{30b}$$

where:

$$c_p^{io} = Ac_x^{io} + Bu_{k-1}, R_p^{io} = [A{R_x}^{io\downarrow} E_w] \tag{31}$$

In this set of equations, the operator ↓ is used to symbolise a dimensional reduction of the zonotope. The presented structure requires the computation of an observer gain, which can be obtained by minimizing the F_W radius of $\langle {c_x^{io}}_p, {R_x^{io}}_p \rangle$. This was covered in depth in [2].

5.2. Design Technique

The gain of the observer is computed by exploiting the LPV formulation of the system. One of the advantages of this type of formulation is that it allows designing the optimal Kalman gain at each of the vertices defined by extreme values of the scheduling variables (see Table 2).

Table 2. Scheduling variables' limits.

Scheduling Variable	Minimum	Maximum
V_x	0.1	3.5
V_y	-2	2
$cos(\theta)$	-1	1
$sin(\theta)$	-1	1
δ	-0.3	0.3

Defining those limits leads to the computation of 2^5 steady-state Kalman gains, which are derived by means of the LMI in Equation (32), presented in [21]. It is remarkable that the purpose of this computation is to derive optimal estimation gains for the estimator and, furthermore, ensure the stability of the algorithm by means of embedding stability conditions inside within the following LMI:

$$\begin{bmatrix} -Y & YA_i - W_i^T C_i & YQ^T & W_i \\ A_i^T Y - C^T W & -Y & 0 & 0 \\ QY & 0 & -I & 0 \\ W^T & 0 & 0 & -R^{-1} \end{bmatrix} < 0, \tag{32a}$$

$$\begin{bmatrix} \gamma I & I \\ I & Y \end{bmatrix} > 0 \tag{32b}$$

The solution of this LMI is obtained by finding Y and W_i for each vertex i. Finally, each of those gains can be computed as $L_i = (W_i Y^{-1})^T$. Then, a Kalman gain can be derived at each operational point by applying a weighted interpolation:

$$\mu_i(\phi) = \prod_{j=1}^{N} \xi(\alpha_j, \beta_j) \tag{33a}$$

$$\alpha_j = \frac{\overline{\phi_j} - \phi_j(k)}{\overline{\phi_j} - \underline{\phi_j}} \tag{33b}$$

$$\beta_j = 1 - \alpha_j \tag{33c}$$

ξ being the function computing all possible combinations and N the number of scheduling variables in ϕ. This allows the definition of L as

$$L(\phi) = \sum_{i=1}^{2^N} \mu_i(\phi) L_i \tag{34}$$

5.3. Designing towards a Practical Implementation

It is clear that due to the nature of the system, there are two subsystems clearly differentiated, which represent both kinematic and dynamic behaviours. During preliminary experiments, it was found that the rate at which this state evolves is dramatically different, and in order to maintain a proper estimation of the kinematic states, the algorithm had to run at a frequency of around 200 Hz. This could lead to performance problems in certain robots; thus, it was decided to explore a cascade architecture, which allows considering both the dynamics and kinematics of the vehicle independently.

In order to do so, firstly, the state vector defined in Equation (24) needs to be split in two parts, the dynamic system being defined by

$$x_d = [v_x, v_y, \omega] \tag{35a}$$

$$y_d = [v_x, \omega] \tag{35b}$$

and the kinematic dynamic system being defined by

$$x_k = [x, y, \theta, x^w_{lm_1}, y^w_{lm_1}, \ldots, x^w_{lm_N}, y^w_{lm_N}] \tag{36a}$$

$$y_k = [x, y, \theta, x^r_{lm_1}, y^r_{lm_1}, \ldots, x^r_{lm_N}, y^r_{lm_N}] \tag{36b}$$

This implies a redefinition of the matrices A, B, and C for each subsystem considering as the inputs of the dynamic system the outputs of the kinematic one. Those matrices are redefined as follows:

$$A_d = A[1:3, 1:3] \qquad B_d = B[1:3, 1:2] \qquad C_d = C[1:3, 1:3] \tag{37}$$

$$A_k = A[3:m, 3:m] \qquad C_k = C[1:3, 1:3] \tag{38}$$

$$B_k = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \tag{39}$$

m being the dimension of the A matrix.

Once both subsystems have been uncoupled, it is trivial to apply the design methodology presented in Section 5.2. Then, they are implemented in a cascade manner, taking into account the need to have a temporal correspondence between both of them, ensuring that the estimation of the dynamic states is aligned with the measurements of the kinematic ones.

6. Implementation

In this section, the implementation of the proposed navigation algorithm within the autonomous driving framework is proposed. Due to the nature of the problem that we wanted to address, the whole system was implemented in ROS. This ensured a proper simulation and the scalability of the results into a real platform, as it is trivial to port the implementation of the codes into the physical RC Car. Furthermore, it provides a realistic temporal behaviour of the detection hardware. A scheme of the proposed platform is represented in Figure 2, which represents the layout of the experiments performed in this work.

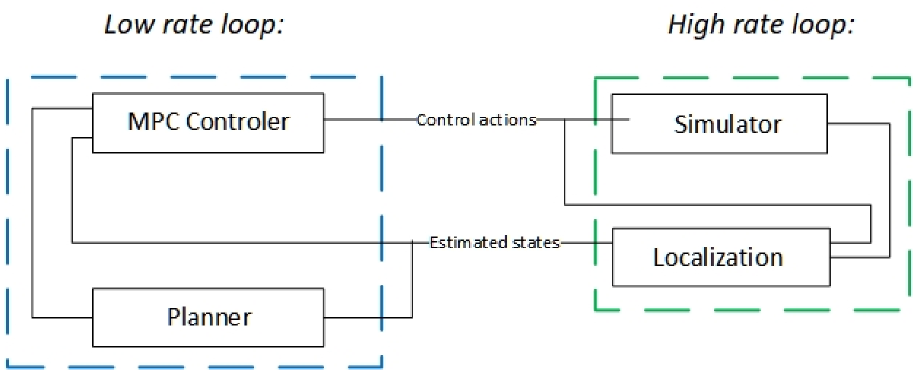

Figure 2. Proposed solution outline.

It can be seen that the system is divided into three main parts: control, simulation, and localisation. Each of these parts is described in the following.

6.1. Experimentation Environment and Simulation

The simulation platform relies on a numeric model of an RC Car, which simulates the dynamic behaviour of the robot, allowing the navigation through the scenario presented in Figure 3, with a representation of both the path to be followed and the landmark. In the typical SLAM problem, where an exploration algorithm is used to map the unknown environment, it is decided to use a known track and then locate the landmarks along the path.

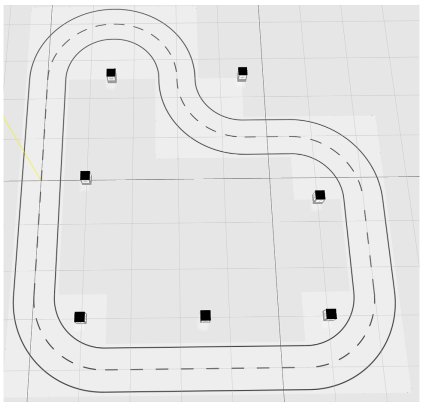

Figure 3. Simulation environment (small dark squares represent the landmarks).

The resulting system relies on a numerical simulation of the dynamical system using the model presented in Section 4 with noise added to both the control actions and the model states. This computation updates the position of the robot in the Gazebo environment. On top of that, we relied on gazebo to simulate landmark detection using a virtual camera and fiducial markers. The motivation behind this dual scheme is that the numeric model used to simulate the vehicle was extensively tested and tuned in [1]. Furthermore, we have perfect knowledge and control over the noise applied to the system, which is an important requirement due to the nature of the algorithms presented. The camera noise defined for this experiment was sampled from a uniform distribution bounded to ± 0.1 m. Noise involved in the system can be found in Equation (40), E_v being associated with the measurements and E_w associated with the model.

$$E_v = \begin{bmatrix} 0.1 & 0.16 & 0.06 & 0.06 & 0.17 \end{bmatrix} \tag{40a}$$

$$E_w = 10^{-3} \begin{bmatrix} 0.2 & 0.18 & 1.40 & 0.13 & 0.16 & 0.068 \end{bmatrix} \tag{40b}$$

6.2. Control and Planning

The controller used to drive the car through the world presented in Section 6.1 is an MPC controller that uses an error-based dynamic model, as presented in [1], along with a spline-based planner, which exploits the fact that the car can be placed within the known track while detecting and estimating the position of each landmarks. This planning approach was covered deeply in [1].

6.3. Localisation

In this section, the implementation of the estimator presented in Section 5.1 is expanded into a localisation approach considering three different scenarios. On the one hand, we implemented an estimation that deals with the dynamic states of the system, which operates by solving Equations (29) and (30) with the subsystem associated with the velocity of the system.

On the other hand, we considered the kinematic estimation connected in a cascade framework, which deals with both robot and landmark position. Firstly, if no landmark is detected, the localisation algorithm behaves as a Kalman filter, using the information available from the on-board sensors, correcting the information provided by the system model without considering the terms that involve the landmark detection.

Secondly, if a non-registered landmark is detected, its position is set by considering the measurement as the real position of the system and instantiating it by applying Equation (14). Finally, if a registered landmark is detected, the model is updated accordingly, and then, the position of both the landmark and vehicle is updated by merging both the camera info and the rest of the on-board sensors. This strategy is stated in Algorithm 2.

Algorithm 2 Landmark estimation algorithm.

```
Initialisation
while Robot is moving do
    $data_{obs} \leftarrow Observation$
    $data_{mov} \leftarrow Odometry$
    if Landmark detected then
        while Landmark list $\neq$ $empty$ do
            if New landmark detected then
                Add to the map the new location
            end if
            if Old landmark detected then
                Load into the state vector the old location
                Update system LPV matrices (22) )
                LPV-KF $\leftarrow x_k, u_k$
                LPV-KF $\rightarrow x_{k+1}$
                Store new estimation
            end if
        end while
        if No landmark detected then
            $x_k = x_k[1:6]$
            LPV-KF $\leftarrow x_k, u_k$
            LPV-KF $\rightarrow x_{k+1}$
        end if
    end if
    Update robot position, and wait until next movement
end while
```

7. Experiments and Results

This section is devoted to the assessment of the proposed approach. The experiment consisted of two complete laps along the circuit proposed in Figure 3. The role of the control is to complete the laps tracking a certain cruising speed. On the other hand, the localisation algorithm provides an estimation of both the robot and landmarks detected along the path along with a region where the position of the robot is considered to be guaranteed.

During this experiment, both the control and estimation were decoupled in order to ensure that they did not interfere each other. It is worth noting that perfect data association was assumed; thus, we considered that the only source of uncertainties was the different noises present in each sensor and the non-perfect modelling, which are both defined as bounded without any prior knowledge of any distribution. Finally, a comparison between the proposed algorithm and a widely used localisation algorithm, the Extended Kalman Filter (EKF), is provided. It is worth noting that in order to keep a proper relation between both strategies, the EKF implementation was performed using LPV techniques in order to avoid Euler discretisation.

The EKF for this comparison was implemented with the same structure presented in Algorithm 2, the only difference being the state estimation, which is generated by applying Equation (41), where A and C are the model matrices, Q and R are the tuning parameters, and L is the gain matrix to be applied in the estimation. Predict:

$$\hat{x}_k^- = A\hat{x}_{k-1}^- + Bu_{k-1} \tag{41a}$$

$$P_k^- = AP_{k-1}A^T + Q \tag{41b}$$

Update:

$$L_k = P_k^- C^T (CP_k^- C^T + R)^{-1} \tag{41c}$$

$$\hat{x}_k = \hat{x}_k^- + L_k(y_k - C\hat{x}_k^-) \tag{41d}$$

$$P_k = (I - L_k C)P_k^- \tag{41e}$$

Firstly, in Figures 4 and 5, the behaviour of the kinematic and dynamic variables of the system can be seen, the performance of both implementations being very similar, as in terms of the RMSE, the EKF and its set-membership version both present similar values, as can be seen in Table 3. This phenomenon was expected, as according to *Combastel*, both estimators are equivalent as long as certain conditions are met. However, as the noise distributions are not assumed when applying intervals, the resulting region will bound the state of the system under any circumstance, which does not apply to an LPV EKF.

Table 3. Error comparison.

	v_x	v_y	w	x	y	θ
LPV EKF	0.0066	0.0747	0.0065	0.0019	0.0019	0.0010
ZKF	0.0088	0.0799	0.0066	0.0020	0.0021	0.0011

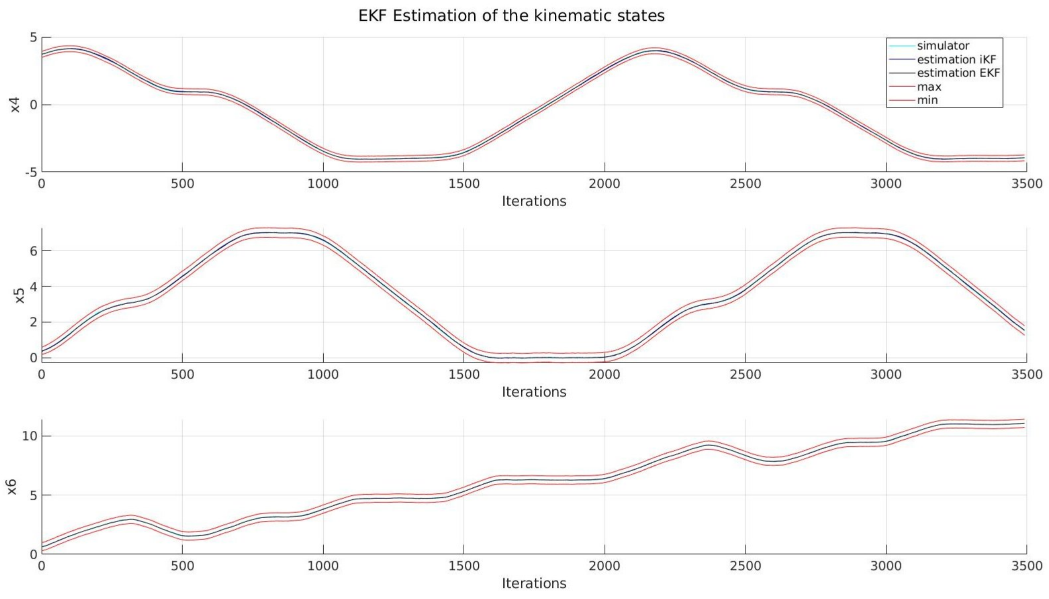

Figure 4. Kinematic states.

Secondly, we can see the resulting estimation of all the landmarks detected during the path; it can be seen that the discussion presented before holds for the rest of the system, and in terms of accuracy, it presents the same behaviour. It is worth noting that due to the similarities between each figure, only one landmark was included, which can be found in Figure 6, while the general behaviour can be seen in Figure 7.

When comparing both implementations, it can be said that the most remarkable difference between both approaches is that the region that restricts the position does not depend on any assumed property of the noise other than its bounds, overcoming in this way one of the limitations of probabilistic implementations of the EKF.

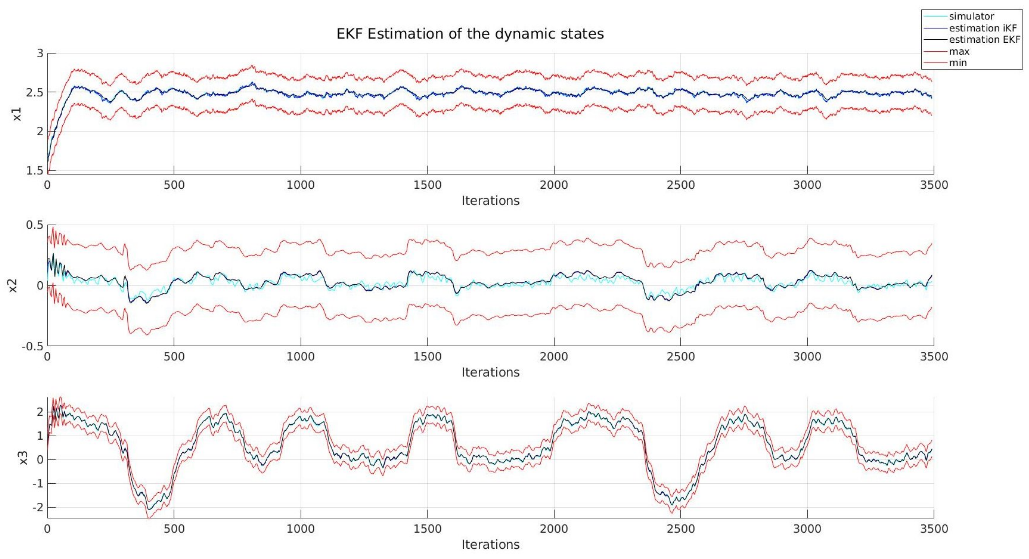

Figure 5. Dynamic states.

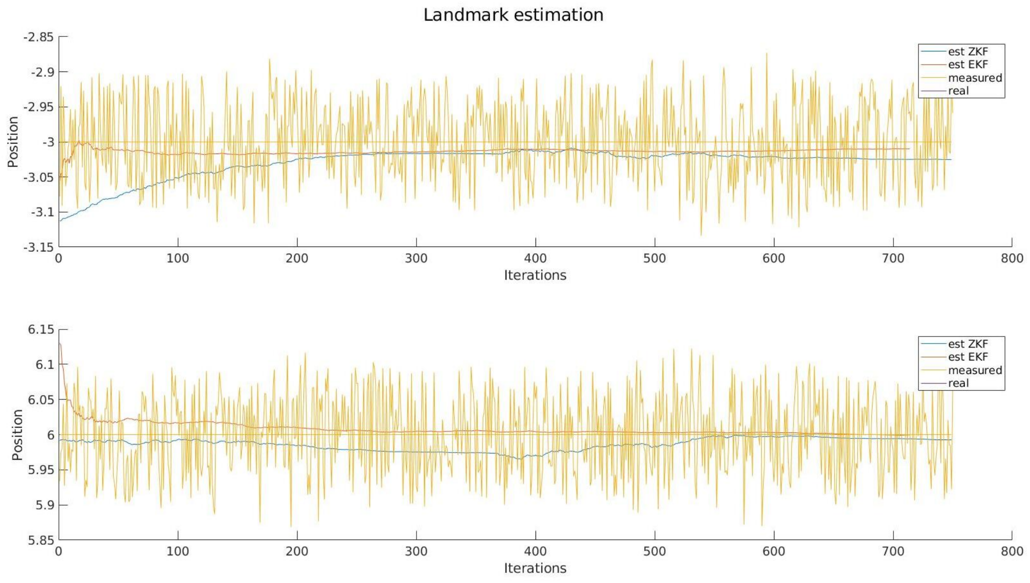

Figure 6. Landmark estimation.

In addition, tests in different conditions to ensure the viability of the algorithm were performed. In particular, we tested how the noise conditions may degrade the performance by doubling the noise levels in the landmark location and diminishing the number of landmarks in the path traversed from seven to four. As can be seen in Table 4, where the results previously shown have been added as the baseline case, for all tested scenarios, the performance was similar, showing the robustness of the algorithm in different conditions.

Table 4. Error in different scenarios.

	v_x	v_y	w	x	y	θ
Baseline case	0.0066	0.0747	0.0065	0.0019	0.0019	0.0010
Noise doubled	0.0071	0.1147	0.0074	0.0021	0.00264	0.0012
4 landmarks	0.0072	0.0676	0.0082	0.0041	0.0034	0.0016

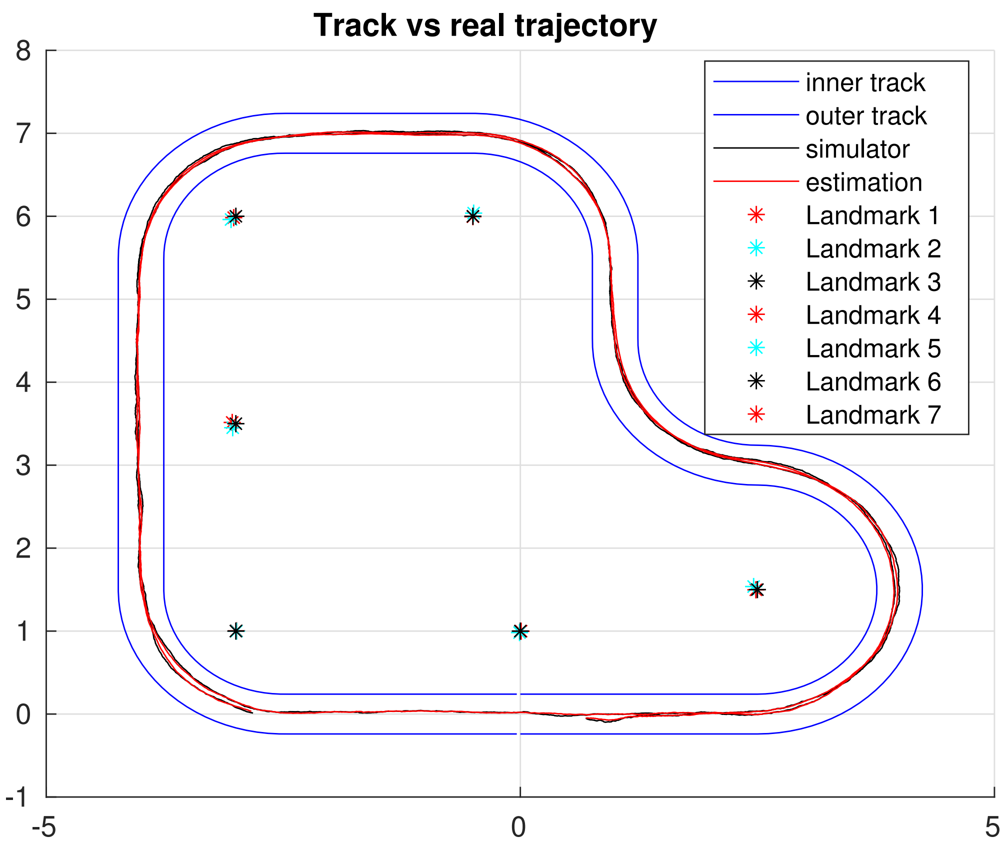

Figure 7. Path along the system.

8. Conclusions and Future Work

In this paper, a zonotopic LPV Kalman filter was proposed as an alternative to the classical EKF for SLAM applied to autonomous vehicles. The proposed approach was able to provide a robust estimation in scenarios where, by construction, probabilistic methods such as the EKF should find their performance trimmed while providing a certain bound to the states that can be used to enhance security in autonomous navigation. The results achieved motivate the usage of interval over probabilistic techniques within the framework studied, as being more flexible in terms of modelling, this ensures proper performance given any bounded noise.

The work presented in this paper opens the door towards enhancing the security of algorithms used within the autonomous driving field. As seen in the literature, most of the state-of-the-art techniques rely on assumptions and relaxations on the characterisation of both the vehicle and the noise, while we proposed a novel approach that is less constraining in this sense. On the one hand, applying LPV modelling allows having an exact linear representation of a nonlinear system. On the other hand, we were able to treat noise by only assuming known bounds. Furthermore, having guaranteed knowledge about the maximum and minimum state values at each time instant allows the design of navigation techniques that can traverse a given path while mathematically ensuring that no collisions will happen as long as the obstacle is not within the bounds of the estimation.

Along the development of this research, different lines of investigation out of the scope of the initial hypothesis appeared, and we consider the following to be the most interesting ones:

- Design control techniques that consider the intervals generated by the localisation to enhance the application safety.
- Create a framework able to adapt itself towards certain sensor failures by exploring localisation within the fault detection field.

- Explore how data-based algorithms could be used to improve the modelling of both the robot and noise.
- Evaluate the performance of the localisation algorithms under extreme circumstances.

In conclusion, we believe that the viability of enhancing probabilistic techniques by applying interval calculus, in particular zonotopes, was assessed and proven to be more flexible in terms of noise definition than other techniques in the field. In addition, the capability of both modelling and designing control estimation algorithms by means of applying LPV techniques is a feasible solution to deal with nonlinear systems within the autonomous driving field.

Author Contributions: Conceptualisation, M.F., V.P. and E.A.; methodology, M.F., V.P. and E.A.; software, M.F.; writing—original draft preparation, M.F.; writing—review and editing, V.P. and E.A.; supervision, V.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been co-financed by the Spanish State Research Agency (AEI) and the European Regional Development Fund (ERFD) through the project SaCoAV (ref. MINECO PID2020-114244RB-I00), by the European Regional Development Fund of the European Union in the framework of the ERDF Operational Program of Catalonia 2014–2020 (ref. 001-P-001643 Looming Factory), and by the DGR of Generalitat de Catalunya (SAC group ref. 2017/SGR/482). The author is also supported by an Industrial PhD AGAUR grant (Ref. 2019 DI 040).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All the required data are included in the paper.

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

References

1. Alcalá, E.; Puig, V.; Quevedo, J.; Rosolia, U. Autonomous racing using linear parameter varying-model predictive control (lpv-mpc). *Control Eng. Pract.* **2020**, *95*, 104270. [CrossRef]
2. Combastel, C. Merging Kalman filtering and zonotopic state bounding for robust fault detection under noisy environment. *IFAC-PapersOnLine* **2015**, *48*, 289–295. [CrossRef]
3. Yu, W.; Zamora, E.; Soria, A. Ellipsoid SLAM: A novel set membership method for simultaneous localization and mapping. *Auton. Robot.* **2016**, *40*, 125–137. [CrossRef]
4. Singandhupe, A.; La, H.M. A Review of SLAM Techniques and Security in Autonomous Driving. In Proceedings of the 2019 Third IEEE International Conference on Robotic Computing (IRC), Naples, Italy, 25–27 February 2019; pp. 602–607. [CrossRef]
5. Ramesh, A.N.; León, C.M.; Zafra, J.C.; Brüggenwirth, S.; González-Huici, M.A. Landmark-based RADAR SLAM for Autonomous Driving. In Proceedings of the 2021 21st International Radar Symposium (IRS), Berlin, Germany, 21–22 June 2021; pp. 1–10. [CrossRef]
6. Bhamidipati, S.; Gao, G. Robust GPS-Vision Localization via Integrity-Driven Landmark Attention. *Navig. J. Inst. Navig.* **2022**, *69*, navi.501. [CrossRef]
7. Wan, G.; Yang, X.; Cai, R.; Li, H.; Zhou, Y.; Wang, H.; Song, S. Robust and precise vehicle localization based on multi-sensor fusion in diverse city scenes. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 4670–4677.
8. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [CrossRef]
9. Paz, L.M.; Tardós, J.D.; Neira, J. Divide and Conquer: EKF SLAM in $O(n)$. *IEEE Trans. Robot.* **2008**, *24*, 1107–1120. [CrossRef]
10. Roh, H.C.; Sung, C.H.; Kang, M.T.; Chung, M.J. Fast SLAM using polar scan matching and particle weight based occupancy grid map for mobile robot. In Proceedings of the 2011 8th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), Incheon, Korea, 23–26 November 2011; pp. 756–757.
11. Mustafa, M.; Stancu, A.; Delanoue, N.; Codres, E. Guaranteed SLAM—An interval approach. *Robot. Auton. Syst.* **2018**, *100*, 160–170. [CrossRef]
12. Fabrice, L.; Bertholom, A.; Sliwka, J.; Jaulin, L. Interval SLAM for underwater robots; a new experiment. *IFAC Proc. Vol.* **2010**, *43*, 42–47.
13. Wang, P.; Xu, P.; Bonnifait, P.; Jiang, J. Box Particle Filtering for SLAM with Bounded Errors. In Proceedings of the 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV), Singapore, 18–21 November 2018; pp. 1032–1038. [CrossRef]

14. Li, S.; Stouraitis, T.; Gienger, M.; Vijayakumar, S.; Shah, J.A. Set-Based State Estimation With Probabilistic Consistency Guarantee Under Epistemic Uncertainty. *IEEE Robot. Autom. Lett.* **2022**, *7*, 5958–5965. [CrossRef]
15. Guerra, E.; Bolea, Y.; Grau, A. Pseudo-measured LPV Kalman filter for SLAM. In Proceedings of the IEEE 10th International Conference on Industrial Informatics, Beijing, China, 25–27 July 2012; pp. 700–705. [CrossRef]
16. Pathiranage, C.D.; Udawatta, L.; Watanabe, K.; Izumi, K. A fuzzy logic based approach to the SLAM problem using pseudolinear models with two sensors data association. In Proceedings of the 2007 Third International Conference on Information and Automation for Sustainability, Melbourne, VIC, Australia, 4–6 December 2007; pp. 70–75. [CrossRef]
17. Xu, Q.; Li, X.; Chan, C.Y. A cost-effective vehicle localization solution using an interacting multiple model- unscented Kalman filters (IMM-UKF) algorithm and grey neural network. *Sensors* **2017**, *17*, 1431. [CrossRef] [PubMed]
18. Combastel, C. A state bounding observer based on zonotopes. In Proceedings of the 2003 European Control Conference (ECC), Cambridge, UK, 1–4 December 2003.
19. Puig, V. Fault diagnosis and fault tolerant control using set-membership approaches: Application to real case studies. *Int. J. Appl. Math. Comput. Sci.* **2010**, *20*, 619–635. [CrossRef]
20. Alcala, E.; Puig, V.; Quevedo, J.; Escobet, T. Gain-scheduling LPV control for autonomous vehicles including friction force estimation and compensation mechanism. *IET Control Theory Appl.* **2018**, *12*, 1683–1693. [CrossRef]
21. Ostertag, E. *Mono- and Multivariable Control and Estimation: Linear, Quadratic and LMI Methods*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011.