sensors

MDPI

Article

Distributed Joint Cooperative Self-Localization and Target Tracking Algorithm for Mobile Networks

Junjie Zhang [1], Jianhua Cui [1,*], Zhongyong Wang [2,*], Yingqiang Ding [2] and Yujie Xia [1]

[1] School of Physics and Electronic Information, Luoyang Normal University, Luoyang 471934, China
[2] School of Information Engineering, Zhengzhou University, Zhengzhou 450001, China
* Correspondence: cuijh@lynu.edu.cn (J.C.); iezywang@zzu.edu.cn (Z.W.)

Received: 9 July 2019; Accepted: 2 September 2019; Published: 4 September 2019

Abstract: Location information is a key issue for applications of the Internet of Things. In this paper, we focus on mobile wireless networks with moving agents and targets. The positioning process is divided into two phases based on the factor graph, i.e., a prediction phase and a joint self-location and tracking phase. In the prediction phase, we develop an adaptive prediction model by exploiting the correlation of trajectories within a short period to formulate the prediction message. In the joint positioning phase, agents calculate the cooperative messages according to variational message passing and locate themselves. Simultaneously, the average consensus algorithm is employed to realize distributed target tracking. The simulation results show that the proposed prediction model is adaptive to the random movement of nodes. The performance of the proposed joint self-location and tracking algorithm is better than the separate cooperative self-localization and tracking algorithms.

Keywords: mobile networks; distributed localization; variational message passing; average consensus; prediction model

1. Introduction

With the development of the Internet of Things (IoT) and related new information technology, wireless sensor networks (WSNs) are expected to gradually penetrate various industries and applications [1–4], influencing all aspects of people's lives. WSNs connect the material world with the human world. Embedded in various "things", sensors sense the states of the "things" and transmit collected information to processing terminals via the Internet to achieve monitoring and management in real-time [5]. The information collected by sensors is meaningful only when combined with location information, which helps the administrators know what, when, and where "things" happened. Therefore, cooperative self-localization and target tracking are two key issues in location-based applications of WSNs.

In WSNs, there are a small number of anchors and thousands of agents. Anchors are equipped with Global Position System (GPS) receivers or other devices with known coordinates, while the agents use a cooperative self-localization algorithm to estimate their own positions. In addition, when a target enters the WSNs, anchors and agents work cooperatively to track and locate it. In static WSNs, anchors and agents are fixed and so their coordinates do not change with time, meaning agents only need to locate themselves once. In mobile WSNs, anchors and agents move with time. Therefore, agents need to locate themselves in a sequential manner. In contrast with static WSNs, mobile WSNs alleviate several issues, such as coverage optimization and target tracking. A plethora of localization systems are proposed in the literature. In [6], the authors provided a general overview of localization in WSNs and showed the importance of different localization approaches in modern IoT applications. In [7], the authors focused on mobile WSNs and analyzed localization algorithms in mobile WSNs.

Cooperative self-localization and target tracking are closely related because target tracking can only be carried out when the locations of agents are clear. In recent years, some researches combined self-localization with target tracking and proposed a series of joint location algorithms. In [8], a simultaneous localization and tracking algorithm was introduced to solve the problem of tracking a non-cooperative target while simultaneously localizing and calibrating the static nodes in the network. In [9], a Bayesian method based on belief propagation (BP) was proposed for the distributed sequential localization of mobile networks composed of both cooperative agents and non-cooperative targets. This provided a consistent combination of cooperative self-localization and distributed tracking. However, the motion parameters (such as velocity) and the state-transition probability density function (pdf) of each node were assumed to be available. In [10], a distributed variational filtering was presented to simultaneously localize the detecting sensors and track the target by exploiting a series of measurements generated in the sensors when the target moved through the network field.

Target tracking is divided into two categories, i.e., centralized and distributed. In centralized target tracking, the nodes that detect the target transmit their measurements related to the target to a central processing node in a hop-by-hop communication mode. Then, the central node determines the positioning of the target by uniformly processing all of the measurements. However, the communication overhead of multi-hop data transmission is very large and energy consumption is too high. In distributed target tracking, each node locates the target according to all measurements related to the target. Therefore, the key problem is how to make each node obtain all of the measurements. In [11], a scheme was proposed in which agents broadcasted their own measurements, identification numbers, and additional information, such as the ID of the target. They also received these from other nodes and updated the information. All of the agents repeated the process until each agent obtained all of the measurement information related to the target. In this way, each agent was a fusion center and could perform tracing tasks. However, much redundant information existed in the network, and the required storage space was also very large. Therefore, this method is suitable for a fully connected network with a large communication radius. Consensus algorithm [12–14] provides another way for each agent to communicate observations with the remaining agents engaged in the tracking. Consensus is a distributed iterative algorithm relying on communication links between neighboring nodes and can achieve entire network polymerization (such as sum, average, or maximum). This does not need a routing algorithm and is robust to changes in network topology and unreliable network environments. Therefore, consensus algorithm is widely used in distributed target tracking [15,16]. In [17], each node calculated the mean and variance of the local posterior by weighted sampling. Then, the average consensus algorithm was employed to calculate the approximation of the global posterior to complete distributed target tracking. In [18], the global likelihood function was approximated to a Gaussian distribution, and each node obtained the approximation through consensus iteration. Furthermore, a consensus algorithm, which is applicable to any exponential likelihood function, was proposed in [19].

As an implementation of approximate Bayesian inference, messaging passing algorithm is particularly well-suited to distributed calculation and has been applied to cooperative self-localization and tracking of WSNs [9,20,21]. In previous work [22], we proposed a distributed cooperative self-localization algorithm by employing variational message passing (VMP) on factor graphs for static WSNs. With regard to the non-Gaussian messages caused by the nonlinear ranging model, we approximated them to Gaussian messages by exploiting second-order Taylor expansion, which significantly reduced the computational complexity. Then, we further combined the VMP approach in [22] with BP and proposed a cooperative self-localization algorithm for mobile WSNs in [23]. In this paper, we focus on mobile networks with anchors, moving agents, and targets deployed in a two-dimensional plane. In particular, the anchors have perfect location information at all times, while the cooperative agents and the non-cooperative targets are at unknown positions and can move independently. We divide the positioning process into two phases, i.e., a prediction phase and a joint self-location and tracking phase. Firstly, an adaptive prediction model is proposed to calculate the

prediction message from the previous time slot. Then, in the joint self-location and tracking phase, the agents locate themselves using the measurements from neighboring nodes (agents and anchors) and the detectable targets, and the average consensus algorithm is employed to achieve distributed positioning of the targets. The simulation results show that the proposed prediction model is more adaptive and accurate than instant prediction, linear prediction, or square prediction, and the proposed joint self-location and tracking algorithm is better than the separate cooperative self-localization and target tracking algorithms.

This paper is organized as follows. In Section 2, we first describe the system model of cooperative self-localization, and tracking. Then, the corresponding probabilistic model and factor graph are analyzed. In Section 3, the proposed distributed joint self-location and tracking algorithm are described in detail, including a prediction phase and a joint self-location and tracking phase. The performance of the proposed prediction model and joint localization algorithm are evaluated in Section 4. Finally, conclusions are drawn in Section 5.

Notations: Boldface lowercase and uppercase letters denote vectors and matrices, respectively. Superscript $(\,)^{\mathrm{T}}$ and symbol $\|\cdot\|$ stand for transposition and Euclidian norm, respectively. The probability density function (pdf) of a 1D Gaussian distribution with mean μ and variance σ^2 is represented by $\mathcal{N}\left(x;\mu,\sigma^2\right)$ while the pdf of a 2D Gaussian distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{V}$ is represented by $\mathcal{N}(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{V})$.

2. System Model and Factor Graph

We consider a network with anchors, moving agents, and targets deployed in a two-dimensional plane, which can be defined by a vertices set $\mathcal{V}$ and an edges set $\mathcal{E}$, i.e., $\mathcal{G}=(\mathcal{V},\mathcal{E})$. Here, each vertex $i\in\mathcal{V}$ represents a node in the plane and each edge $(i,j)\in\mathcal{E}$ indicates a neighborhood structure between node i and node j. We divide set $\mathcal{V}$ into three parts: $\mathcal{V}=\mathcal{A}\cup\mathcal{M}\cup\mathcal{S}$, where $\mathcal{A}$ designates all anchors with perfect location information, $\mathcal{M}$ includes all agents at unknown positions, and $\mathcal{S}$ represents all non-cooperative targets. Moreover, the number of anchors, agents, and targets are denoted by N_A, N_M, and N_O, respectively.

Time is slotted and we assume that all agents and targets move independently. At the kth time slot, the position variable of node $i\in\mathcal{V}$ is denoted by $\boldsymbol{x}_i^k\triangleq\left[x_i^k,y_i^k\right]^{\mathrm{T}}$. For simplicity of illustration, the communication link between cooperative neighbors is assumed to be symmetric, which means $(i,j)\in\mathcal{E}$ and $(j,i)\in\mathcal{E}$ if $|\boldsymbol{x}_i^k-\boldsymbol{x}_j^k|\leq R$, where R is the communication radius. At the kth time slot, if $(i,j)\in\mathcal{E}$, nodes i can obtain the current position of node j and a ranging measurement $d_{i\leftarrow j}^k$ with Gaussian noise $\omega_{ij}^k\sim\mathcal{N}\left(\omega_{ij}^k;0,\left(\sigma_{ij}^k\right)^2\right)$. Consequently, the ranging measurement $d_{i\leftarrow j}^k$ can be formulated as

$$d_{i\leftarrow j}^k=\|\boldsymbol{x}_i^k-\boldsymbol{x}_j^k\|+\omega_{ij}^k,$$

and the likelihood function of the position variables of $\boldsymbol{x}_i^k$ and $\boldsymbol{x}_j^k$ is a Gaussian pdf, i.e.,

$$p(d_{i\leftarrow j}^k\left|\boldsymbol{x}_i^k,\boldsymbol{x}_j^k\right.)\sim\mathcal{N}(d_{i\leftarrow j}^k;\|\boldsymbol{x}_i^k-\boldsymbol{x}_j^k\|,\,\sigma_{ij}^k).$$

For convenience of description, we define $\mathcal{V}_o^k\triangleq\mathcal{A}_o^k\cup\mathcal{M}_o^k$ ($\mathcal{A}_o^k$ for anchors and $\mathcal{M}_o^k$ for agents) to represent all nodes who can detect target at the kth time slot. Similarly, let $\mathcal{V}_m^k\triangleq\mathcal{A}_m^k\cup\mathcal{M}_m^k\cup\mathcal{S}_m^k$ denote all nodes from whom node $m\in\mathcal{M}$ can implement ranging measurement at the kth time slot. Typically, notation $\mathcal{S}_m^k$ represents all targets that node m can detect at the kth time slot. Moreover, we denote the position variables of all nodes by $\boldsymbol{X}^k\triangleq\left\{\boldsymbol{x}_i^k,\forall i\in\mathcal{A}\cup\mathcal{M}\cup\mathcal{S}\right\}$, the ranging measurements of node $m\in\mathcal{M}$ by $\boldsymbol{Z}_m^k\triangleq\left\{d_{m\leftarrow j}^k,\forall j\in\mathcal{A}_m^k\cup\mathcal{M}_m^k\cup\mathcal{S}_m^k\right\}$, and the ranging measurements of target $O\in\mathcal{S}$ by $\boldsymbol{Z}_o^k\triangleq\left\{d_{j\leftarrow o}^k,\forall j\in\mathcal{A}_o^k\cup\mathcal{M}_o^k\right\}$, respectively. Then, let $\boldsymbol{X}^k\triangleq\left\{\boldsymbol{x}_i^k,\forall i\in\mathcal{V}\right\}$ and $\boldsymbol{Z}^k\triangleq\left\{\boldsymbol{Z}_i^k,\forall i\in\mathcal{M}\cup\mathcal{S}\right\}$ represent the position variables of all nodes and ranging measurements at the kth time slot, respectively.

Furthermore, let $\boldsymbol{X}^{0:K} \triangleq \left\{\boldsymbol{X}^k, k = 0:K\right\}$ denote the position variables of all nodes from the 0th time slot to the kth time slot and $\boldsymbol{Z}^{1:K} \triangleq \left\{\boldsymbol{Z}^k, k = 1:K\right\}$ denote the ranging measurements from the 1st time slot to the Kth time slot, respectively.

According to Bayesian rules and the assumption that all nodes move independently, the joint a posteriori distribution of $\boldsymbol{X}^{0:K}$ with given observations $\boldsymbol{Z}^{0:K}$ can be formulated as

$$\begin{aligned} p(\boldsymbol{X}^{0:K}\big|\boldsymbol{Z}^{1:K}) &\propto p(\boldsymbol{X}^0)\prod_{k=1}^{K} p(\boldsymbol{Z}^k\big|\boldsymbol{X}^k)p(\boldsymbol{X}^k\big|\boldsymbol{X}^{k-1}) \\ &\propto \prod_{i\in\mathcal{V}} p(\boldsymbol{x}_i^0)\prod_{k=1}^{K}\prod_{m\in\mathcal{M}}\prod_{j\in\mathcal{V}_m^k} p(d_{m\leftarrow j}^k\big|\boldsymbol{x}_m^k,\boldsymbol{x}_j^k)\prod_{o\in\mathcal{S}}\prod_{j\in\mathcal{V}_o^k} p(d_{j\leftarrow o}^k\big|\boldsymbol{x}_o^k,\boldsymbol{x}_j^k)\prod_{i\in\mathcal{V}} p(\boldsymbol{x}_i^k\big|\boldsymbol{x}_i^{k-1}) \end{aligned}, \quad (1)$$

where $p\left(\boldsymbol{x}_i^0\right)$ is the a prior distribution of node I, which is assumed to be a Gaussian pdf with mean μ_i^0 and covariance matrix $\boldsymbol{V}_i^0 \triangleq \sigma_{i0}^2\boldsymbol{I}_{2\times 2}$, $p(d_{i\leftarrow j}^k\big|\boldsymbol{x}_i^k,\boldsymbol{x}_j^k)$ is the likelihood function of node i and node j, and $p\left(\boldsymbol{x}_i^k\big|\boldsymbol{x}_i^{k-1}\right)$ is the probabilistic state-transition function.

For simplicity, we define $f_i^{k|k-1} \triangleq p\left(\boldsymbol{x}_i^k\big|\boldsymbol{x}_i^{k-1}\right)$ and $f_{ij}^k \triangleq p(d_{i\leftarrow j}^k\big|\boldsymbol{x}_i^k,\boldsymbol{x}_j^k)$. Based on the factorization in (1), the joint a posteriori distribution of $p\left(\boldsymbol{X}^{0:K}\big|\boldsymbol{Z}^{0:K}\right)$ can be represented by a factor graph, as shown in Figure 1. In the factor graph, each variable node represents the position variable $\boldsymbol{x}_i^k$ of node i and is depicted by a circle, while each factor node represents a local function $f_i^{k|k-1}$ or f_{ij}^k and is drawn by a square. Moreover, if a variable is an argument of a local function, it is connected to the factor node through an edge.

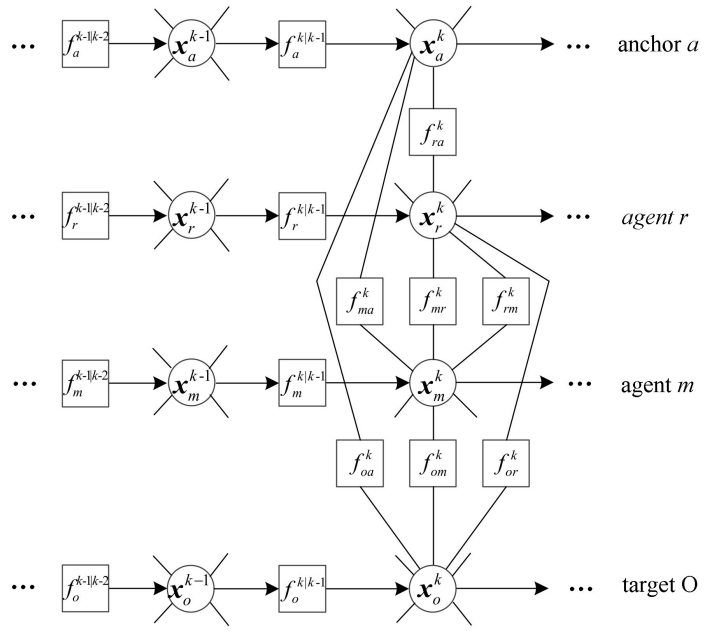

Figure 1. The factor graph corresponding to the factorization in (1).

Generally speaking, factor graphs are undirected digraphs, which means the edge between a factor node and a variable node is undirected. However, in Figure 1, messages only flow forward in time regarding the spatiotemporal constraints of the network. We do not calculate the messages from the present to the past because network connectivity may change and the states of the nodes may be outdated. Furthermore, the variable node of an anchor's position ignores the messages from its neighbors as anchors' positions are known.

3. Distributed Joint Self-Location and Tracking Algorithm

As shown in Figure 1, for the cooperative agent $m \in \mathcal{M}$ and the non-cooperative target $O \in \mathcal{S}$, the beliefs of $\boldsymbol{x}_m^k$ and $\boldsymbol{x}_O^k$, denoted by $b\left(\boldsymbol{x}_m^k\right)$ and $b\left(\boldsymbol{x}_o^k\right)$, consist of two kinds of message: Prediction messages from the ($k-1$)th time slot and cooperative messages from the neighbors. Based on message passing rules, for node $i \in \mathcal{M} \cup \mathcal{S}$, the belief $b\left(\boldsymbol{x}_i^k\right)$ can be described as

$$b\left(\boldsymbol{x}_i^k\right) = \frac{1}{Z} m_{f_i^{k|k-1} \to \boldsymbol{x}_i^k}\left(\boldsymbol{x}_i^k\right) \prod_{j \in \mathcal{V}_i^k} m_{f_{ij}^k \to \boldsymbol{x}_i^k}\left(\boldsymbol{x}_i^k\right), \tag{2}$$

where Z is the normalization constant and $m_{f_i^{k|k-1} \to \boldsymbol{x}_i^k}\left(\boldsymbol{x}_i^k\right)$ and $m_{f_{ij}^k \to \boldsymbol{x}_i^k}\left(\boldsymbol{x}_i^k\right)$ represent messages from factor node and factor nodes f_{ij}^k ($\forall j \in \mathcal{V}_i^k$) to variable node $\boldsymbol{x}_i^k$, respectively. Therefore, agent $m \in \mathcal{M}$ determines $b\left(\boldsymbol{x}_m^k\right)$ and $b\left(\boldsymbol{x}_o^k\right)(m \in \mathcal{V}_o^k)$ by two phases, i.e., a prediction phase and a joint self-location and tracking phase.

3.1. Adaptive Prediction Model and Prediction Message Calculating

In (1), the probabilistic state-transition function $p(\boldsymbol{x}_i^k | \boldsymbol{x}_i^{k-1})$ of node i is related to the position prediction model. In this paper, according to the inertia of the motion, we propose an adaptive prediction model by exploiting the correlation of trajectories of node i within a short period.

We denote the trajectory of node i from the ($k-N$)th time slot to the ($k-1$)th time slot as $\boldsymbol{T}_i(k-1,N) \triangleq \left((\hat{\boldsymbol{x}}_i^{k-1})^{\mathrm{T}}, (\hat{\boldsymbol{x}}_i^{k-2})^{\mathrm{T}}, \cdots, (\hat{\boldsymbol{x}}_i^{k-N})^{\mathrm{T}}\right)^{\mathrm{T}}$, where $\hat{\boldsymbol{x}}_i^{k-N} \triangleq [\hat{x}_i^{k-N}, \hat{y}_i^{k-N}]^{\mathrm{T}}, \ldots, \hat{\boldsymbol{x}}_i^{k-1} \triangleq [\hat{x}_i^{k-1}, \hat{y}_i^{k-1}]^{\mathrm{T}}$ are the estimated positions of node i at the ($k-N$)th, ... , ($k-1$)th time slot. Similarly, the trajectory of node i from the ($k-N+1$)th time slot to the kth time slot can be described as $\boldsymbol{T}_i(k,N) \triangleq \left((\hat{\boldsymbol{x}}_i^k)^{\mathrm{T}}, (\hat{\boldsymbol{x}}_i^{k-1})^{\mathrm{T}}, \cdots, (\hat{\boldsymbol{x}}_i^{k-N+1})^{\mathrm{T}}\right)^{\mathrm{T}}$. Then, we formulate the prediction model of node i as

$$\boldsymbol{T}_i(k,N) = \boldsymbol{P}^i \cdot \boldsymbol{T}_i(k-1,N)\ , \tag{3}$$

where $\boldsymbol{P}^i$ is the prediction matrix of node i and its dimension is $2N \times 2N$. Obviously, there are only two different elements between $\boldsymbol{T}_i(k-1,N)$ and $\boldsymbol{T}_i(k,N)$, i.e., $\hat{\boldsymbol{x}}_i^k$ and $\hat{\boldsymbol{x}}_i^{k-N}$. Therefore, the elements from the 3rd to the $2N$th rows are

$$p_{r,s}^i = \begin{cases} 0, & r \geq 3 \text{ and } r \neq s+2 \\ 1, & r \geq 3 \text{ and } r = s+2 \end{cases}\ . \tag{4}$$

According to the inertia of the motion of node i, we assume that the prediction matrix $\boldsymbol{P}^i$ is unchanged in a short period. Based on the prediction model in (4), we have

$$\boldsymbol{T}_i(k-1,N) = \boldsymbol{P}^i \cdot \boldsymbol{T}_i(k-2,N)\ . \tag{5}$$

Consequently, it can be formulated as

$$\boldsymbol{T}_i(k,N) = \left(\boldsymbol{P}^i\right)^n \cdot \boldsymbol{T}_i(k-n,N). \tag{6}$$

Based on Equations (3) and (5), we have

$$\boldsymbol{T}_i(k,N) - \boldsymbol{T}_i(k-1,N) = \boldsymbol{P}^i \cdot [\boldsymbol{T}_i(k-1,N) - \boldsymbol{T}_i(k-2,N)]\ . \tag{7}$$

Obviously, translational motion will not change the prediction matrix $\boldsymbol{P}^i$. Therefore, it can be expressed as

$$\boldsymbol{T}_i(k,N) - \boldsymbol{H} = \boldsymbol{P}^i \cdot [\boldsymbol{T}_i(k-1,N) - \boldsymbol{H}] \,, \tag{8}$$

and $\boldsymbol{H}$ satisfies the constraint as follows

$$\boldsymbol{H} = (\boldsymbol{P}^i)^n \cdot \boldsymbol{H}, \tag{9}$$

where $\boldsymbol{H} \triangleq [a, b, \ldots, a, b]^{\mathrm{T}}_{2N\times 1}$ and its two base vectors are $\boldsymbol{h}_1 \triangleq [1\,0\,1\,0\ldots 1\,0]^{\mathrm{T}}$ and $\boldsymbol{h}_2 \triangleq [0\,1\,0\,1\ldots 0\,1]^{\mathrm{T}}$. Then we have $\boldsymbol{h}_1 = \boldsymbol{P}^i \cdot \boldsymbol{h}_1$ and $\boldsymbol{h}_2 = \boldsymbol{P}^i \cdot \boldsymbol{h}_2$.

Assume that each node stores M ($M > N$) estimated positions before the kth time slot, which constitute $M - N + 1$ trajectories, i.e., $\boldsymbol{T}_i(k-1,N)$, $\boldsymbol{T}_i(k-2,N)$, ... , $\boldsymbol{T}_i(k-M+N-1,N)$. Based on the prediction model in Equation (3), we obtain an equation set which can be expressed as

$$\begin{cases} \boldsymbol{T}_i(k-1,N) = \boldsymbol{P}^i \times \boldsymbol{T}_i(k-2,N) \\ \boldsymbol{T}_i(k-2,N) = \boldsymbol{P}^i \times \boldsymbol{T}_i(k-3,N) \\ \vdots \\ \boldsymbol{T}_i(k-M+N,N) = \boldsymbol{P}^i \times \boldsymbol{T}_i(k-M+N-1,N) \end{cases}. \tag{10}$$

According to Equations (4) and (10), we have

$$\begin{cases} \begin{pmatrix} 1 & 0 \end{pmatrix} = h_1^{\mathrm{T}} \times \begin{pmatrix} (\boldsymbol{p}^i_{1,*})^{\mathrm{T}} & (\boldsymbol{p}^i_{2,*})^{\mathrm{T}} \end{pmatrix} \\ \begin{pmatrix} 0 & 1 \end{pmatrix} = h_2^{\mathrm{T}} \times \begin{pmatrix} (\boldsymbol{p}^i_{1,*})^{\mathrm{T}} & (\boldsymbol{p}^i_{2,*})^{\mathrm{T}} \end{pmatrix} \end{cases}, \tag{11}$$

where $\boldsymbol{p}^i_{1,*}$ and $\boldsymbol{p}^i_{2,*}$ are the first and second rows of $\boldsymbol{P}^i$. For convenience of description, we define

$$\boldsymbol{B}_{i,M-N} \triangleq \begin{bmatrix} \hat{x}_i^{k-1} & \hat{y}_i^{k-1} \\ \hat{x}_i^{k-2} & \hat{y}_i^{k-2} \\ \vdots & \vdots \\ \hat{x}_i^{k-M+N} & \hat{y}_i^{k-M+N} \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \; \boldsymbol{T}_{i,M-N} \triangleq \begin{bmatrix} [\boldsymbol{T}_i(k-2,N)]^{\mathrm{T}} \\ [\boldsymbol{T}_i(k-3,N)]^{\mathrm{T}} \\ \vdots \\ [\boldsymbol{T}_i(k-M+N-1,N)]^{\mathrm{T}} \\ h_1^{\mathrm{T}} \\ h_2^{\mathrm{T}} \end{bmatrix}.$$

Then, according to Equations (4), (10), and (11), we have

$$\boldsymbol{B}_{i,M-N} = \boldsymbol{T}_{i,M-N} \cdot \begin{pmatrix} (\boldsymbol{p}^{\mathrm{i}}_{1,*})^{\mathrm{T}} & (\boldsymbol{p}^{\mathrm{i}}_{2,*})^{\mathrm{T}} \end{pmatrix}. \tag{12}$$

Consequently, $\boldsymbol{p}^i_{1,*}$ and $\boldsymbol{p}^i_{2,*}$ can be determined based on the least square method, i.e.,

$$\begin{pmatrix} \boldsymbol{p}^i_{1,*} \\ \boldsymbol{p}^i_{2,*} \end{pmatrix} = \boldsymbol{B}^{\mathrm{T}}_{i,M-N} \cdot \boldsymbol{T}_{i,M-N} \cdot \left[(\boldsymbol{T}^{\mathrm{T}}_{i,M-N} \cdot \boldsymbol{T}_{i,M-N})^{-1} \right]^{\mathrm{T}}. \tag{13}$$

Finally, the prediction matrix $\boldsymbol{P}^i$ is given by Equations (4) and (13). Let $\widetilde{\boldsymbol{x}}_i^k \triangleq \left[\widetilde{x}_i^k, \widetilde{y}_i^k\right]^{\mathrm{T}}$ denote the predicted position of node i. According to the prediction model in Equation (3), $\widetilde{\boldsymbol{x}}_i^k$ can be given by

$$\widetilde{\boldsymbol{x}}_i^k = \begin{pmatrix} \boldsymbol{p}^{\mathrm{i}}_{1,*} \\ \boldsymbol{p}^{\mathrm{i}}_{2,*} \end{pmatrix} \cdot \boldsymbol{T}_i(k-1,N) \tag{14}$$

Then, we use the predicted position $\widetilde{\boldsymbol{x}}_i^k$ to formulate the prediction message $m_{f_i^{k|k-1}\to\boldsymbol{x}_i^k}(\boldsymbol{x}_i^k)$ as

$$m_{f_i^{k|k-1}\to\boldsymbol{x}_i^k}\left(\boldsymbol{x}_i^k\right)=\mathcal{N}\left(\boldsymbol{x}_i^k;\widetilde{\boldsymbol{x}}_i^k,\widetilde{\boldsymbol{V}}_i^k\right), \tag{15}$$

where the covariance matrixes are $\widetilde{\boldsymbol{V}}_i^k=\widetilde{\sigma}_{i,k}^2\boldsymbol{I}_{2\times2}$ and $\widetilde{\sigma}_{i,k}^2=\|\hat{\boldsymbol{x}}_i^{k-1}-\widetilde{\boldsymbol{x}}_i^{k-1}\|^2$.

3.2. Joint Self-Location and Tracking Phase

After calculating the prediction message based on the adaptive prediction model, agent $m\in\mathcal{M}$ locates itself using ranging measurements, not only from its neighbor anchors and agents $j\in\mathcal{A}_i^k\cup\mathcal{M}_i^k$, but also from targets $O\in\mathcal{S}_i^k$ that it can detect. Meanwhile, agents and anchors that can detect target O share information with each other based on consensus iteration and determine the position of the target O. Compared with the separate cooperative self-localization and algorithms, agents use the ranging measurements from targets to improve the self-location accuracy, which promotes tracking performance in turn.

3.2.1. Self-Location of Agents Based on Variational Message Passing

At the kth time slot, agent $m\in\mathcal{M}$ obtains its prediction message $m_{f_m^{k|k-1}\to\boldsymbol{x}_m^k}\left(\boldsymbol{x}_m^k\right)=\mathcal{N}\left(\boldsymbol{x}_m^k;\widetilde{\boldsymbol{x}}_m^k,\widetilde{\boldsymbol{V}}_m^k\right)$ based on the adaptive prediction model described in Section 3. Then, according to VMP update rules, cooperative messages from its neighbors are expressed as follows:

$$\begin{aligned} m_{f_{ma}^k\to\boldsymbol{x}_m^k}\left(\boldsymbol{x}_m^k\right) &\triangleq \exp\left\{\int m_{\boldsymbol{x}_a^k\to f_{ma}^k}\left(\boldsymbol{x}_a^k\right)\ln f_{ma}^k\left(\boldsymbol{x}_a^k,\boldsymbol{x}_m^k\right)\mathrm{d}\boldsymbol{x}_a^k\right\} \\ &= \mathcal{N}\left(d_{m\leftarrow a}^k;\|\,\boldsymbol{x}_m^k-\mu_a^k\,\|,\left(\sigma_{ma}^k\right)^2\right), \end{aligned} \tag{16}$$

$$\begin{aligned} m_{f_{mj}^k\to\boldsymbol{x}_m^k}\left(\boldsymbol{x}_m^k\right) &= \exp\left\{\int m_{\boldsymbol{x}_j^k\to f_{mj}^k}\ln p\left(d_{m\leftarrow j}^k\middle|\boldsymbol{x}_m^k,\boldsymbol{x}_j^k\right)\mathrm{d}\boldsymbol{x}_j^k\right\} \\ &= \exp\left\{\int b\left(\boldsymbol{x}_j^k\right)\ln\mathcal{N}\left(d_{m\leftarrow j}^k;\|\,\boldsymbol{x}_m^k-\boldsymbol{x}_j^k\,\|,\left(\sigma_{mj}^k\right)^2\right)\mathrm{d}\boldsymbol{x}_j^k\right\}. \end{aligned} \tag{17}$$

In (16), $\boldsymbol{\mu}_a^k$ is the position of anchor $a\in\mathcal{A}$ at the kth time slot. Consequently, the belief $b(\boldsymbol{x}_m^k)$ in Equation (2) can be rewritten as

$$\begin{aligned} b\left(\boldsymbol{x}_m^k\right) &= \tfrac{1}{Z}\mathcal{N}\left(\boldsymbol{x}_m^k;\widetilde{\boldsymbol{x}}_m^k,\widetilde{\boldsymbol{V}}_m^k\right)\times\prod_{a\in\mathcal{A}_m^k}\mathcal{N}\left(d_{m\leftarrow a}^k;\|\,\boldsymbol{x}_m^k-\mu_a^k\,\|,\left(\sigma_{ma}^k\right)^2\right) \\ &\quad\times\prod_{j\in\mathcal{M}_m^k\cup\mathcal{S}_m^k}\exp\left\{\int b\left(\boldsymbol{x}_j^k\right)\ln\mathcal{N}\left(d_{m\leftarrow j}^k;\|\,\boldsymbol{x}_m^k-\boldsymbol{x}_j^k\,\|,\left(\sigma_{mj}^k\right)^2\right)\mathrm{d}\boldsymbol{x}_j^k\right\}. \end{aligned} \tag{18}$$

Considering the communication overhead, the messages passed on in Figure 1 are restricted to Gaussian distribution. However, $b(\boldsymbol{x}_m^k)$, given by Equation (18), is non-Gaussian because of the nonlinear ranging model. In [20], we propose a low-complexity approximation method based on second-order Taylor expansion. In this way, the belief $b(\boldsymbol{x}_m^k)$ is approximated into a Gaussian function denoted by $\hat{b}(\boldsymbol{x}_m^k)\sim\mathcal{N}\left(\boldsymbol{x}_m^k;\hat{\boldsymbol{\mu}}_m^k,\hat{\boldsymbol{V}}_m^k\right)$, where $\boldsymbol{\mu}_i^k$ and $\boldsymbol{V}_i^k$ are as follows:

$$\boldsymbol{V}_m^k\triangleq\left\{\left(\widetilde{\boldsymbol{V}}_m^k\right)^{-1}+\sum_{a\in\mathcal{A}_m^k}\frac{\boldsymbol{I}_{2\times2}-d_{m\leftarrow a}^k\nabla_{F_{ma}}^2}{\left(\sigma_{ma}^k\right)^2}+\sum_{j\in\mathcal{M}_i^k}\frac{1}{\left(\sigma_{mj}^k\right)^2}[\boldsymbol{I}_{2\times2}-d_{m\leftarrow j}^k\frac{\partial^2F_{mj}}{\partial\left(\boldsymbol{x}_m^k\right)^2}]\right\}^{-1}, \tag{19}$$

$$\boldsymbol{\mu}_m^k \triangleq \hat{\boldsymbol{V}}_m^k \left\{ \left(\widetilde{\boldsymbol{V}}_m^k\right)^{-1} \widetilde{\boldsymbol{x}}_m^k + \sum_{a \in \mathcal{A}_m^k} \frac{\boldsymbol{\mu}_a^k - d_{i \leftarrow a}^k \left(\nabla_{F_{ma}} - \nabla_{F_{ma}}^2 \boldsymbol{\mu}_m^{k*}\right)}{\left(\sigma_{ma}^k\right)^2} + \sum_{j \in \mathcal{M}_m^k \cup \mathcal{S}_m^k} \frac{1}{\left(\sigma_{mj}^k\right)^2} \left[\boldsymbol{\mu}_j^{k*} + d_{m \leftarrow j}^k \left(\frac{\partial F_{mj}}{\partial \boldsymbol{x}_m^k} - \frac{\partial^2 F_{mj}}{\partial \left(\boldsymbol{x}_m^k\right)^2} \boldsymbol{\mu}_m^{k*}\right)\right] \right\}. \tag{20}$$

In Equations (19) and (20), the notations $\boldsymbol{\mu}_m^{k*}$ and $\boldsymbol{\mu}_j^{k*}$ represent the estimated positions of nodes m and j in the latest iteration, respectively, notations $\nabla_{F_{ma}}$ and $\nabla_{F_{ma}}^2$ are the first-order gradient and the Hessian matrix of $F_{ma}^k \triangleq \|\boldsymbol{x}_m^k - \mu_a^k\|$ at $\boldsymbol{\mu}_m^{k*}$, respectively, and the notations $\frac{\partial F_{mj}}{\partial \boldsymbol{x}_m^k}$ and $\frac{\partial^2 F_{mj}}{\partial \left(\boldsymbol{x}_m^k\right)^2}$ are the first-order partial derivatives of $F_{mj}^k \triangleq \|\boldsymbol{x}_m^k - \boldsymbol{x}_j^k\|$ and $\frac{1}{\partial \boldsymbol{x}_m^k}\left(\frac{\partial F_{mj}}{\partial \boldsymbol{x}_m^k}\right)$ at $(\boldsymbol{\mu}_m^{k*}, \boldsymbol{\mu}_j^{k*})$, respectively (please see [22] for detailed derivation). Consequently, agent $i \in \mathcal{M}$ obtains its location based on maximum a posteriori (MAP), i.e., $\hat{\boldsymbol{x}}_i^k = \boldsymbol{\mu}_i^k$.

3.2.2. Target Tracking Based on Average Consensus

At kth time slot, node $i \in \mathcal{S}_o^k$, which can detect target $O \in \mathcal{S}$, computes the prediction message $m_{f_o^{k|k-1} \to \boldsymbol{x}_o^k}\left(\boldsymbol{x}_o^k\right) = \mathcal{N}\left(\boldsymbol{x}_o^k; \widetilde{\boldsymbol{x}}_o^k, \widetilde{\boldsymbol{V}}_o^k\right)$ and the local message $m_{f_{oi}^k \to \boldsymbol{x}_o^k}\left(\boldsymbol{x}_o^k\right)$. Then, $m_{f_{oi}^k \to \boldsymbol{x}_o^k}\left(\boldsymbol{x}_o^k\right)$ is approximated into Gaussian function $\mathcal{N}\left(\boldsymbol{x}_o^k; \mu_{oi}^k, \boldsymbol{V}_{oi}^k\right)$ using the approximation method in [22]. Therefore, the Gaussian approximation of $b\left(\boldsymbol{x}_o^k\right)$ can be expressed as

$$\hat{b}\left(\boldsymbol{x}_o^k\right) = \frac{1}{Z} \mathcal{N}\left(\boldsymbol{x}_o^k; \widetilde{\boldsymbol{x}}_o^k, \widetilde{\boldsymbol{V}}_o^k\right) \times \prod_{i \in \mathcal{V}_o^k} \mathcal{N}\left(\boldsymbol{x}_o^k; \mu_{oi}^k, \boldsymbol{V}_{oi}^k\right). \tag{21}$$

The mean vector $\boldsymbol{\mu}_o^k$ and the covariance matrix $\boldsymbol{V}_o^k$ are given by $\left(\boldsymbol{V}_o^k\right)^{-1} = \left(\widetilde{\boldsymbol{V}}_o^k\right)^{-1} + \sum_{i \in \mathcal{V}_o^k} \left(\boldsymbol{V}_{oi}^k\right)^{-1}$ and $\mu_o^k = \boldsymbol{V}_o^k \left(\left(\widetilde{\boldsymbol{V}}_i^k\right)^{-1} \mu_i^k + \sum_{i \in \mathcal{V}_o^k} \left(\boldsymbol{V}_{oi}^k\right)^{-1} \mu_{oi}^k \right)$, respectively. Each node $i \in \mathcal{S}_o^k$ can calculate $\widetilde{\boldsymbol{V}}_o^k$ and $\boldsymbol{V}_{oi}^k$ locally. However, node $i \in \mathcal{S}_o^k$ cannot determine $\hat{b}(\boldsymbol{x}_o^k)$ unless all other neighbors of target O share their local information with it.

For convenience of discussion, let $\boldsymbol{W}_o^k \triangleq \left(\boldsymbol{V}_o^k\right)^{-1}$, $\widetilde{\boldsymbol{W}}_o^k \triangleq \left(\widetilde{\boldsymbol{V}}_o^k\right)^{-1}$ and $\boldsymbol{W}_{oi}^k \triangleq \left(\boldsymbol{V}_{oi}^k\right)^{-1}$; therefore, we have

$$\boldsymbol{W}_o^k = \widetilde{\boldsymbol{W}}_o^k + \sum_{i \in \mathcal{V}_o^k} \boldsymbol{W}_{oi}^k, \tag{22}$$

$$\boldsymbol{W}_o^k \mu_o^k = \widetilde{\boldsymbol{W}}_o^k \widetilde{\boldsymbol{x}}_o^k + \sum_{i \in \mathcal{V}_o^k} \boldsymbol{W}_{oi}^k \mu_{oi}^k. \tag{23}$$

In order to locate the target in a distributed manner, we use the average consensus based on Metropolis weights proposed in [13] to diffuse information across the network. During the process, each node updates its data with a weighted average of its neighbors' data, which converges to the global average. Let t denote the iteration index and N_i^k represent the number of neighbors of node i, respectively. The consensus iteration process is as follows.

Step 1: Each node $i \in \mathcal{V}_o^k$ uses local information to initialize the global average $\boldsymbol{W}_i^{k,(0)}$ and $\boldsymbol{W}_i^{k,(0)} \mu_i^{k,(0)}$ as

$$\boldsymbol{W}_i^{k,(0)} = \boldsymbol{W}_{oi}^k, \boldsymbol{W}_i^{k,(0)} \mu_i^{k,(0)} = \boldsymbol{W}_{oi}^k \mu_{oi}^k,$$

while each node $i \notin \mathcal{V}_o^k$ initializes $\boldsymbol{W}_i^{k,(0)}$ and $\boldsymbol{W}_i^{k,(0)}\mu_i^{k,(0)}$ as

$$\boldsymbol{W}_i^{k,(0)} = 0, \boldsymbol{W}_i^{k,(0)}\mu_i^{k,(0)} = 0.$$

Step 2: Each node computes the Metropolis weight matrix, which is defined as

$$\xi_{i,j}^{(t)} = \begin{cases} \frac{1}{1+\max\left\{N_i^k, N_j^k\right\}} & \text{if } j \in \mathcal{V}_i^k, \\ 1 - \sum\limits_{l \in \mathcal{V}_i^k} \xi_{i,l}^{(t)} & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{24}$$

Step 3: Each node broadcasts its global average computed in the latest iteration and collects the information broadcasted by its neighbors. Then, each node updates the global average $\boldsymbol{W}_i^{k,(t+1)}$ and $\boldsymbol{W}_i^{k,(t+1)}\boldsymbol{\mu}_i^{k,(t+1)}$ by

$$\boldsymbol{W}_i^{k,(t+1)} = \xi_{i,i}^{(t)}\boldsymbol{W}_i^{k,(t)} + \sum_{j \in \mathcal{V}_i^k} \xi_{i,j}^{(t)}\boldsymbol{W}_j^{k,(t)}, \tag{25}$$

$$\boldsymbol{W}_i^{k,(t+1)}\mu_i^{k,(t+1)} = \xi_{i,i}^{(t)}\boldsymbol{W}_i^{k,(t)}\mu_i^{k,(t)} + \sum_{j \in \mathcal{V}_i^k} \xi_{i,j}^{(t)}\boldsymbol{W}_j^{k,(t)}\mu_j^{k,(t)}. \tag{26}$$

Step 4: If $t = t_{\max}$ (where $t_{\max}$ is the maximum number of iteration), the consensus process ends. Otherwise, go to Step 2.

After sufficient consensus iterations, all nodes obtain the global average of $\sum\limits_{i \in \mathcal{V}_o^k} \boldsymbol{W}_{oi}^k$ and $\sum\limits_{i \in \mathcal{V}_o^k} \boldsymbol{W}_{oi}^k\mu_{oi}^k$. Therefore, each node $i \in \mathcal{V}_o^k$ can compute $\sum\limits_{i \in \mathcal{V}_o^k} \boldsymbol{W}_{oi}^k$ and $\sum\limits_{i \in \mathcal{V}_o^k} \boldsymbol{W}_{oi}^k\,\boldsymbol{\mu}_{oi}^k$ by

$$\sum_{i \in \mathcal{V}_o^k} \boldsymbol{W}_{oi}^k = N_o^k \times \boldsymbol{W}_i^{k,(t_{\max})}, \tag{27}$$

$$\sum_{i \in \mathcal{V}_o^k} \boldsymbol{W}_{oi}^k\mu_{oi}^k = N_o^k \times \boldsymbol{W}_i^{k,(t_{\max})}\mu_i^{k,(t_{\max})}. \tag{28}$$

Submitting Equations (27) and (28) to Equations (22) and (23), each node $i \in \mathcal{V}_o^k$ has

$$\boldsymbol{W}_o^k = \widetilde{\boldsymbol{W}}_o^k + N_o^k \times \boldsymbol{W}_i^{k,(t_{\max})}, \tag{29}$$

$$\boldsymbol{W}_o^k\boldsymbol{\mu}_o^k = \widetilde{\boldsymbol{W}}_o^k\widetilde{\boldsymbol{x}}_o^k + N_o^k \times \boldsymbol{W}_i^{k,(t_{\max})}\boldsymbol{\mu}_i^{k,(t_{\max})}. \tag{30}$$

Therefore, each node $i \in \mathcal{V}_o^k$ can determine the position of target O based on MAP, i.e., $\hat{\boldsymbol{x}}_o^k = \boldsymbol{\mu}_o^k$.

3.3. Implementation Scheme of the Joint Self-Location and Tracking Algorithm

The proposed distributed joint self-location and tracking algorithm based on VMP and average consensus is stated as follows.

Algorithm 1: Distributed joint self-location and tracking algorithm

for $k = 1 : k_{\max}$(k is the time slot index)

Each agent $m \in \mathcal{M}$ updates M historical position information of itself and the targets before the kth time slot.

Each agent $m \in \mathcal{M}$ calculates its prediction matrix $\boldsymbol{P}^m$ using Equation (4) and Equation (13), then predicts its position $\widetilde{\boldsymbol{x}}_m^k$ using Equations (14) and obtains its prediction message $m_{f_m^{k|k-1} \to \boldsymbol{x}_m^k}(\boldsymbol{x}_m^k)$ using Equation (15).

Each node $i \in \mathcal{S}_o^k$ predicts the position of target O using Equation (4), Equation (13), and Equation (14), i.e., $\widetilde{\boldsymbol{x}}_o^k$. Then, the prediction message $m_{f_o^{k|k-1} \to \boldsymbol{x}_o^k}(\boldsymbol{x}_o^k)$ is calculated using Equation (15).

Initialization: Each agent $m \in \mathcal{M}$ initializes its position as $\boldsymbol{x}_m^k = \widetilde{\boldsymbol{x}}_m^k$ and initializes the position of the target that it detects as $\boldsymbol{x}_o^k = \widetilde{\boldsymbol{x}}_o^k$.

for $q = 1 : q_{\max}$(where q is the iteration index)

(1) Each agent $m \in \mathcal{M}$ collects ranging measurements from its neighbors and calculates cooperative messages $m^{(q)}_{f_{ma}^k \to \boldsymbol{x}_m^k}\left(\boldsymbol{x}_m^k\right)(\forall a \in \mathcal{A}_m^k)$ from neighbor anchors using Equation (16) and $m^{(q)}_{f_{mj}^k \to \boldsymbol{x}_m^k}\left(\boldsymbol{x}_m^k\right)(\forall j \in \mathcal{M}_m^k \cup \mathcal{S}_m^k)$ from neighbor agents and targets using Equation (17), respectively. Then, each agent $m \in \mathcal{M}$ calculates $\boldsymbol{\mu}_m^{k,(q)}$ and $\boldsymbol{V}_m^{k,(q)}$ of $\hat{b}^{(p)}\left(\boldsymbol{x}_m^k\right)$ using Equations (19) and (20), respectively.

(2) Each node $i \in \mathcal{S}_o^k$ receives the local message $m^{(q)}_{f_{oi}^k \to \boldsymbol{x}_o^k}(\boldsymbol{x}_o^k)$ and approximates it into Gaussian function $\mathcal{N}(\boldsymbol{x}_o^k; \boldsymbol{\mu}_{oi}^{k,(q)}, \boldsymbol{V}_{oi}^{k,(q)})$ based on the Taylor expansion approximation method in [20]. Then the global average is calculated based on average consensus iteration from Step 1 to Step 4, which are depicted in Section 3.2.2. Then, each node $i \in \mathcal{S}_o^k$ computes $\boldsymbol{W}_o^{k,(q)}$ and $\boldsymbol{W}_o^{k,(q)}\boldsymbol{\mu}_o^{k,(q)}$ using Equations (27)–(30).

(3) Each agent $m \in \mathcal{M}$ broadcasts the mean $\boldsymbol{\mu}_m^{k,(q)}$ and the covariance matrix $\boldsymbol{V}_m^{k,(q)}$ of $\hat{b}^{(p)}(\boldsymbol{x}_m^k)$. Each node $i \in \mathcal{S}_o^k$ broadcasts $\boldsymbol{W}_o^{k,(q)}\boldsymbol{\mu}_o^{k,(q)}$ and $\boldsymbol{W}_o^{k,(q)}$. At the same time, they collect the information broadcasted by its neighbors.

end for q

Each agent $m \in \mathcal{M}$ determines its position based on MAP and broadcasts $\boldsymbol{\mu}_m^k$ and $\boldsymbol{V}_m^k$. Each node $i \in \mathcal{S}_o^k$ determines the position of target O based on based on MAP and broadcast $\boldsymbol{W}_o^k$ and $\boldsymbol{W}_o^k\boldsymbol{\mu}_o^k$.

end for k

4. Stimulation Results and Analysis

In this section, we evaluate the performance of the proposed adaptive prediction model and the distributed joint self-location and tracking algorithm (denoted by JSLT).

4.1. Stimulation Results and Performance Analysis of the Proposed Adaptive Prediction Model

In order to assess the performance of the proposed adaptive prediction model, we compare it with linear prediction and square prediction. We consider a 50 m × 50 m plane with 10 moving nodes. The prediction error is defined as the distance between the true position and the prediction position. The average prediction errors in Figures 2–4 are obtained by averaging over 10,000 realizations, with each realization continuing 20 time slots.

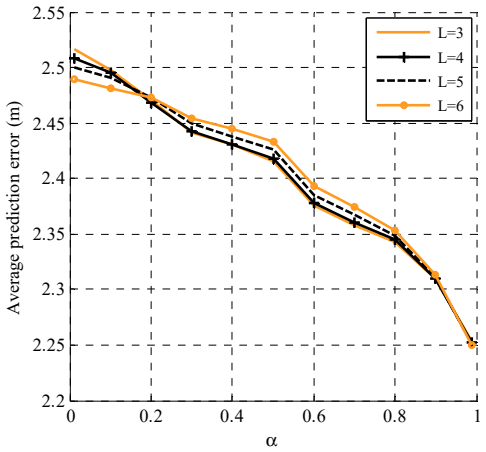

Figure 2. The performance of the proposed adaptive prediction model with different numbers of historical positions and an adjustment parameter α.

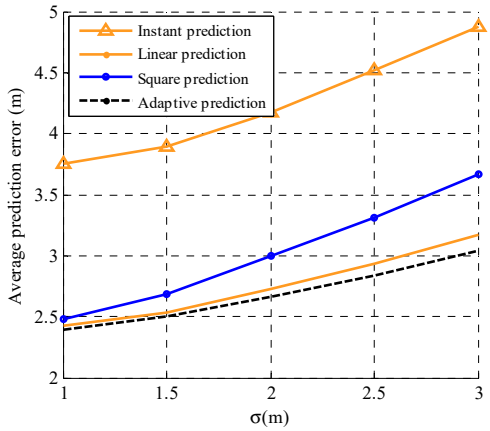

Figure 3. The performance comparison of the four prediction models with different standard deviations of the random variables.

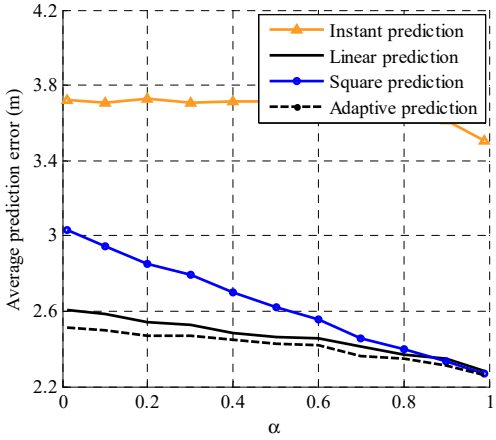

Figure 4. The performance comparison of the four prediction models with different adjustment parameter α values.

We employ the Gauss–Markov (GM) mobility model [24] to simulate the motion of nodes. In the GM model, the velocity and direction of nodes are time-dependent Gauss–Markov processes, which means the velocity and direction at the kth time slot is related to that at the (k−1)th time slot. Let d_i^k and θ_i^k denote the velocity and direction of node i from the kth time slot to the (k−1)th time slot, respectively, then we have

$$\begin{cases} d_i^k = \alpha d_i^{k-1} + (1-\alpha)\overline{d}_i + \sqrt{1-\alpha^2}\omega_d \\ \theta_i^k = \alpha \theta_i^{k-1} + (1-\alpha)\overline{\theta}_i + \sqrt{1-\alpha^2}\omega_\theta \end{cases}, \tag{31}$$

where constants $\overline{d}_i$ and $\overline{\theta}_i$ are the velocity and direction of node i when $k \to \infty$, ω_d and ω_θ are Gaussian random variables independent of $\overline{d}_i$ and $\overline{\theta}_i$, and $\alpha \in [0, 1]$ is an adjustment parameter to control the randomness of motion. When α increases, the impact of d_i^{k-1} and θ_i^{k-1} rises, while the impact of ω_d and ω_θ declines. As discussed above, the Gauss–Markov mobility model not only reflects the relationship between the current moment and a historical moment, but also reflects the randomness of motion. Therefore, it is very suitable for the simulation of high random motion, such as the travel of birds and the diffusion of gas.

The performance of the proposed adaptive prediction model with different numbers of historical estimated positions and α is illustrated in Figure 2. Simulation configurations are as follows. For the GM mobility model shown in Equation (31), we set $\overline{d}_i = 5$ m/s and $\overline{\theta}_i = 60^\circ$. For the prediction using Equation (13), the parameters are $N = 1$ and $M = 3/4/5/6$. As we know, the prediction algorithm is based on the historical estimated positions. In initialization, we use the true positions along with Gaussian noise (mean = 0 and variance = 4 m^2) as the estimated positions. As shown in Figure 2, there is little difference in the average prediction error using different historical positions with different α. Comparatively speaking, when the value of α is smaller, more historical positions are used and the more accurate the obtained prediction is; when the value of α is bigger, more historical positions are used and a more accurate prediction is achieved. This is because a smaller α value indicates a greater randomness of motion, therefore the impact of the (k−1)th time slot on the kth time slot is less. More historical positions can reflect the trend of the movement. However, a bigger α value leads to a stronger correlation between the kth time slot and the (k−1)th time slot. In this situation, the contribution of the earlier time slots to the current moment is small, and using too much historical information even influences the accuracy of prediction. Moreover, greater the number of historical positions used, the more information the nodes store and the greater the energy consumption. In subsequent simulations, we use three historical positions, meaning each node stores three historical positions.

To further evaluate the performance of the proposed adaptive prediction model, we compare the proposed model with linear prediction, square prediction, and instant prediction, which use the movements in x axis and y axis at the (k−1)th time slot to predict the position at the kth time slot. Simulation configurations are $\alpha = 0.7$, $\overline{d}_i = 3$ m/s, $\overline{\theta}_i = 60^\circ$, $N = 1$, and $M = 3$. The performance comparison of the four prediction models with different standard deviations $\sigma_{\omega_d} = \sigma_{\omega_\theta} = \sigma$ of the random variables ω_d and ω_θ is shown in Figure 3. It is observed that the average prediction error of the four prediction models increases with an increase in σ. Specifically, the performance of instant prediction is the worst and the performance of the proposed adaptive prediction is the best. Moreover, the advantage of the proposed adaptive prediction model is more obvious when σ is bigger. Figure 4 illustrates the performance of the four prediction models with different adjustment parameter α values. Simulation configurations are $\alpha = 0.7$, $\overline{d}_i = 3$ m/s, $\overline{\theta}_i = 60^\circ$, $\sigma = 1$ m and $M = 3$. It can be observed that the average prediction error of the proposed adaptive prediction model is smaller than the other three prediction models and the difference is bigger with increasing α. In fact, from the Gauss–Markov mobility model in Equation (31), we see that the randomness of motion rises with the decline of α and the increase in σ. Therefore, the proposed adaptive prediction model better adapts to different motion patterns than the other three prediction models.

4.2. Stimulation Results and Performance Analysis of the Proposed Distributed Joint Self-Location and Tracking Algorithm

The proposed distributed joint self-location and tracking (JSLT) algorithm is evaluated by comparison with the distributed cooperative self-localization algorithm and the centralized target tracking algorithm in this section. We consider a 50 m × 50 m plane with $N_A = 5$ anchors, N_M agents, and N_O targets. The communication radius and ranging radius of anchors and agents are $R_c = 25$ m, the detection radius is $R_d = 25$ m, and the standard variance of ranging measurement noise is 1 m. At the 0th time slot, the variance matrix of the prior of agents and targets are set to be $25\boldsymbol{I}_{2\times2}$. The iterations of the average consensus algorithm and the joint location algorithm are 20 and 10, respectively. The results in Figures 5–7 are obtained by performing 10,000 realizations over 20 time slots.

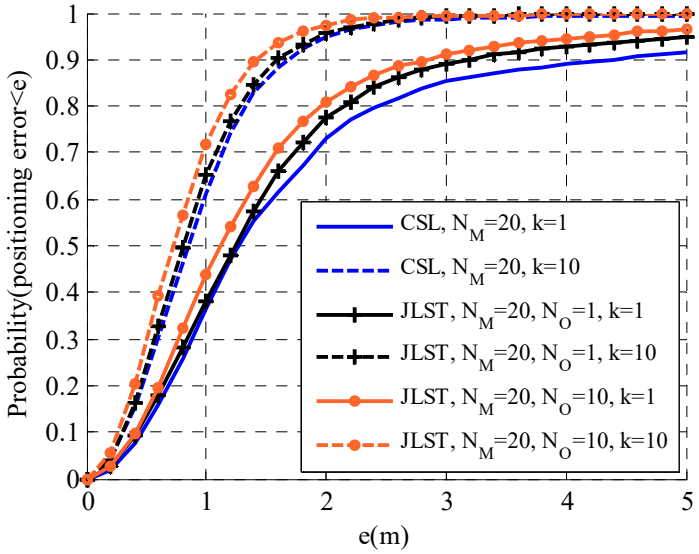

Figure 5. Self-localization performance of the proposed joint self-location tracking JSLT algorithm and the cooperative self-localization (CSL) algorithm (N_M= 20).

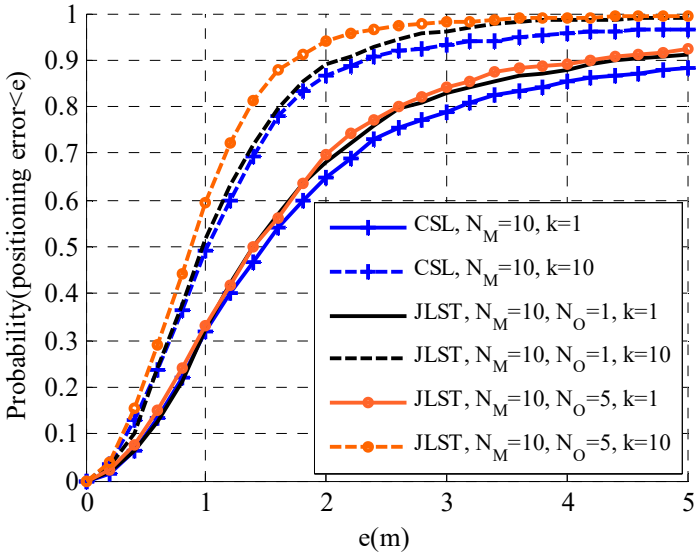

Figure 6. Self-localization performance of the proposed JSLT algorithm and the CSL algorithm (N_M= 10).

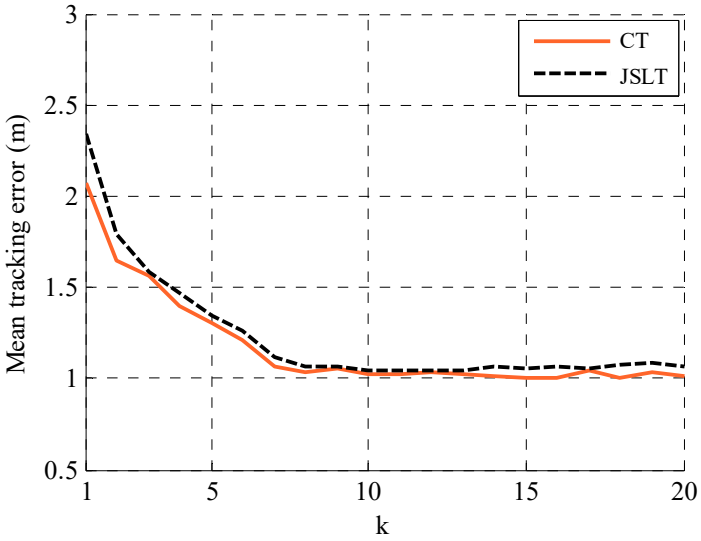

Figure 7. Tracking performance of the proposed JSLT algorithm and the CT algorithm.

Self-localization performance of the proposed JSLT algorithm and the cooperative self-localization (CSL) algorithm in [23] with different numbers of agents and targets are shown in Figures 5 and 6. In Figure 5, N_M= 20, N_A = 1 or 10; in Figure 6, N_M= 10, N_A = 1 or 5. It can be seen that the localization performance of both algorithms improves as time increases. This is because the agents' and their neighbors' positions become more accurate with the increase in time. However, a higher k value does not always correspond to improved positioning accuracy. The estimated accuracy becomes stable as time goes on. Furthermore, in the two scenarios, the proposed JSLT algorithm has a higher positioning accuracy than the CSL algorithm. This is because, in the JSLT algorithm, agents not only use the distance observation from the neighboring anchors and agents as in the CSL algorithm, but they also make use of the ranging measurements from the targets that they can detect, which improves the self-localization accuracy. Moreover, the performance of the two algorithms is quite different with fewer agents and more targets. When the density of agents is high, agents have enough neighbors. Therefore, they can localize themselves well by cooperating with neighboring anchors and agents. However, when the density of agents is low, many agents have few neighboring anchors and agents to localize themselves. In this case, the ranging measurements from targets play an important role in the self-localization of agents. The computational complexity of both algorithms depends on the number of neighbors. For agent $m \in \mathcal{M}$, we define N_m^A, N_m^M, and N_m^O to denote the number of neighboring agents, anchors, and targets that can detected at a certain time. The computational complexity of the CSL algorithm and the proposed JSLT algorithm are $O(N_i^A + N_i^M)$ and $O(N_i^A + N_i^M + N_i^O)$, respectively. With regard to communication overhead, as all messages are approximated to Gaussian function, each agent broadcasts the mean and the covariance to its neighbors and collects the information broadcasted by its neighbors. The information collected by each agent is $O(N_i^A + N_i^M)$ and $O(N_i^A + N_i^M + N_i^O)$ for the CSL algorithm and the proposed JSLT algorithm, respectively. Since $N_i^O \ll N_i^A + N_i^M$, the complexity and the communication overhead of the two algorithms are similar.

Figure 7 shows the tracking performance of the proposed JSLT algorithm and the centralized tracking (CT) algorithm is illustrated with N_M= 20 and N_A = 1. In the CT algorithm, agents first localize themselves using the CSL algorithm in [23], then anchors and agents track the targets by sending information to a control center. As shown in Figure 7, the tracking performance of the proposed distributed JSLT algorithm is very close to that of CT algorithm, but the communication overhead of JSLT algorithm is much less than that of CT algorithm. In the CT algorithm, the information collected and broadcasted by each node (anchor or agent) are both proportional to the number of neighbors. In the proposed JLST algorithm, each node broadcasts its global average to its neighbors and collects global averages broadcasted by its neighbors, which is proportional to the number of neighbors.

Therefore, the communication overhead sharply reduces by half. Moreover, in the CT algorithm, anchors and agents transmit information to the control center in a multi-hop manner, which leads to a fair amount of redundant information in the networks and information delay.

In order to visualize the tracking performance, a single trial performance of target tracking is shown in Figure 8. We can see that the trajectory recognized by the proposed algorithm is close to the actual trajectory of the target, except for at the initial time and at the moment of a sharp turn.

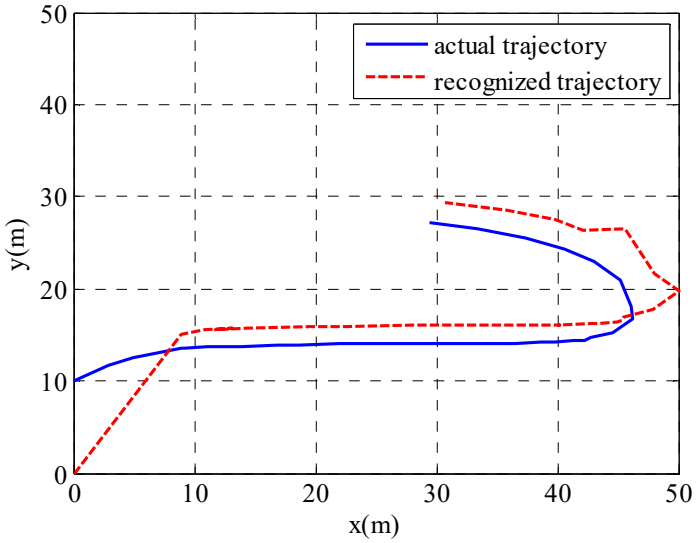

Figure 8. A single trial performance of target tracking.

5. Conclusions

In this paper, we built an adaptive prediction model by exploiting the correlation of trajectories and then proposed a joint self-location and tracking algorithm based on VMP and average consensus. Simulation results and analysis illustrated that, compared with instant prediction, linear prediction, and square prediction, the proposed prediction model was more adaptive to the movement of the nodes. The self-location performance of the proposed algorithm is better than the separated cooperative self-localization, and the tracking performance of the proposed algorithm is close to the centralized tracking algorithm with less communication overhead.

The proposed algorithm is suitable for tracking bird migration patterns, monitoring gas diffusion, etc. In these applications, the motion of the targets is random. The following research aimed to apply the proposed approach to practices. In practical networks, delays in data transmission and communication errors are two important issues which affect positioning accuracy and real-time tracking. Future research will focus on the issue of how to deal with the impact of these two issues.

Author Contributions: Conceptualization, J.C. and Z.W.; Methodology, J.C. and Y.D.; Software, J.C. and Y.D.; Validation, J.C. and Z.W., and Y.D.; Formal analysis, J.C.; Investigation, J.C.; Resources, J.C.; Data curation, J.C.; Writing—Original draft preparation, J.Z.; Writing—Review and editing, J.Z. and Y.X.; Visualization, J.Z.; Supervision, Z.W.; Project administration, Z.W.; Funding acquisition, J.Z., Z.W., and Y.X.

Funding: This research was funded by the National Natural Science Foundation of China (NSFC) (grant number 61571402), the Key Scientific Research Project of Colleges and Universities of Henan Province, China (grant number 19A510019) and the Foundation for Young Backbone Teachers in Higher Education Institutions of Henan Province, China (grant number 2018GGJS126).

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

References

1. Antolín, D.; Medrano, N.; Calvo, B. Reliable Lifespan Evaluation of a Remote Environment Monitoring System Based on Wireless Sensor Networks and Global System for Mobile Communications. *J. Sens.* **2016**, *2016*, 4248230. [CrossRef]
2. Fakhrulddin, S.S.; Gharghan, S.K.; Al-Naji, A.; Chahl, J. An Advanced First Aid System Based on an Unmanned Aerial Vehicles and a Wireless Body Area Sensor Network for Elderly Persons in Outdoor Environments. *Sensors* **2019**, *19*, 2955. [CrossRef] [PubMed]
3. Han, W.; Tian, Z.; Shi, W.; Huang, Z.; Li, S. Low-Power Distributed Data Flow Anomaly-Monitoring Technology for Industrial Internet of Things. *Sensors* **2019**, *19*, 2804. [CrossRef] [PubMed]
4. Suryadevara, N.K.; Mukhopadhyay, S.C.; Kelly, S.D.T.; Satinder, P.S.G. WSN-Based Smart Sensors and Actuator for Power Management in Intelligent Buildings. *IEEE/ASME Trans. Mechatron.* **2015**, *20*, 564–571. [CrossRef]
5. Atzori, L.; Lera, A.; Morabito, G. The Internet of Things: A Survey. *Comput. Netw.* **2010**, *54*, 2787–2805. [CrossRef]
6. Fekher, K.; Abbas, B.; Abderrahim, B.; Priyanka, R.; Mohamed, A. A Survey of Localization Systems in Internet of Things. *Mob. Netw. Appl.* **2019**, *24*, 761–785.
7. Chelouah, L.; Semchedine, F.; Bouallouche-Medjkoune, L. Localization Protocols for Mobile Wireless Sensor Networks: A survey. *Comput. Electr. Eng.* **2018**, *71*, 733–751. [CrossRef]
8. Taylor, C.; Rahimi, A.; Bachrach, J.; Shrobe, H.; Grue, A. Simultaneous Localization, Calibration, and Tracking in an Ad Hoc Sensor Network. In Proceedings of the International Conference on Information Processing in Sensor Networks, Nashville, TN, USA, 19–21 April 2006; pp. 27–33.
9. Meyer, F.; Hlinka, O.; Wymeersch, H.; Riegler, E.; Hlawatsch, F. Distributed Localization and Tracking of Mobile Networks Including Noncooperative Objects. *IEEE Trans. Signal Inf. Process. Netw.* **2016**, *2*, 57–71. [CrossRef]
10. Teng, J.; Snoussi, H.; Richard, C.; Zhou, R. Distributed Variational Filtering for Simultaneous Sensor Localization and Target Tracking in Wireless Sensor Networks. *IEEE Trans. Veh. Technol.* **2012**, *61*, 2305–2318. [CrossRef]
11. Djurić, P.M.; Beaudeau, J.; Bugallo, M.F. Non-Centralized Target Tracking with Mobile Agents. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Prague Congress Center, Prague, Czech Republic, 22–27 May 2011; pp. 5928–5931.
12. Xiao, L.; Boyd, S. Fast Linear Iterations for Distributed Averaging. In Proceedings of the 42nd IEEE International Conference on Decision and Control (IEEE CDC), Maui, HI, USA, 9–12 December 2003.
13. Xiao, L.; Boyd, S.; Lall, S. A Scheme for Robust Distributed Sensor Fusion Based on Average Consensus. In Proceedings of the International Symposium on Information Processing in Sensor Networks, Boise, ID, USA, 15 April 2005.
14. Olfati-Saber, R.; Fax, J.A.; Murray, R.M. Consensus and Cooperation in Networked Multi-Agent Systems. *Proc. IEEE* **2007**, *95*, 215–233. [CrossRef]
15. Farahmand, S.; Roumeliotis, S.I.; Giannakis, G.B. Set-Membership Constrained Particle Filter: Distributed Adaptation for Sensor Networks. *IEEE Trans. Signal Process.* **2011**, *59*, 4122–4138. [CrossRef]
16. Hlinka, O.; Hlawatsch, F.; Djuric, P.M. Consensus-based Distributed Particle Filtering with Distributed Proposal Adaptation. *IEEE Trans. Signal Process.* **2014**, *62*, 3029–3041.
17. Gu, D.; Sun, J.; Hu, Z.; Li, H. Consensus Based Distributed Particle Filter in Sensor Networks. In Proceedings of the International Conference on Information and Automation, Zhangjiajie, Hunan, China, 20–23 June 2008.
18. Ghirmai, T. Distributed Particle Filter Using Gaussian Approximated Likelihood Function. In Proceedings of the 48th Conference on Information Sciences and Systems, Princeton, NJ, USA, 19–21 March 2014.
19. Hlinka, O.; Sluciak, O.; Hlawatsch, F.; Djuric, P.M.; Rupp, M. Likelihood Consensus and Its Application to Distributed Particle Filtering. *IEEE Trans. Signal Process.* **2012**, *60*, 4334–4349. [CrossRef]
20. Wymeersch, H.; Lien, J.; Win, M.Z. Cooperative Localization in Wireless Networks. *Proc. IEEE* **2009**, *97*, 427–450. [CrossRef]
21. Yuan, W.J.; Wu, N.; Etzlinger, B.; Wang, H.; Kuang, J. Cooperative Joint Localization and Clock Synchronization Based on Gaussian Message Passing in Asynchronous Wireless Networks. *IEEE Trans. Veh. Technol.* **2016**, *65*, 7258–7273. [CrossRef]

22. Cui, J.; Wang, Z.; Zhang, C.; Sun, P.; Zhu, Z. Variational Message Passing-based Localisation Algorithm with Taylor Expansion for Wireless Sensor Networks. *IET Commun.* **2016**, *10*, 2396–2401. [CrossRef]
23. Cui, J.; Wang, Z.; Zhang, C.; Zhang, Y.; Zhu, Z. Message Passing Localisation Algorithm Combining BP with VMP for Mobile Wireless Sensor Networks. *IET Commun.* **2017**, *11*, 1106–1113. [CrossRef]
24. Camp, T.; Boleng, J.; Davies, V. A Survey of Mobility Models for Ad Hoc Network Research. *Wirel. Commun. Mob. Comput.* **2002**, *2*, 483–502. [CrossRef]