applied sciences

MDPI

Article

Temporal Logic Planning and Receding Horizon Control for Signal Source Localization

Xingtong Chen, Qiang Lu *, Dilong Chen and Boyuan Geng

School of Automation, Hangzhou Dianzi University, Hangzhou 310018, China
* Correspondence: lvqiang@hdu.edu.cn; Tel.: +86-138-1913-9153

Abstract: This article copes with signal source localization by employing a receding horizon control approach with temporal logic planning in the light of a single mobile robot. First, a temporal logic planning approach is proposed such that the task requirements from the temporal logic specifications can be effectively dealt with based on the product automaton in an offline fashion. Second, in order to label the key nodes of the product automaton, a particle filter is utilized to predict the source positions as the key nodes. Third, on the basis of the product automaton, a receding horizon control approach with temporal logic planning is developed to produce the robot's trajectory that satisfies a given linear temporal logic specification. Finally, the effectiveness of the proposed control approach is illustrated for signal source localization.

Keywords: mobile robots; receding horizon control; linear temporal logic; obstacle avoidance

check for updates

Citation: Chen, X.; Lu, Q.; Chen, D.; Geng, B. Temporal Logic Planning and Receding Horizon Control for Signal Source Localization. *Appl. Sci.* **2022**, *12*, 10984. https://doi.org/10.3390/app122110984

Academic Editor: Andrea Prati

Received: 04 October 2022
Accepted: 27 October 2022
Published: 30 October 2022

1. Introduction

In the last decade, signal source localization has received much attention from researchers [1–8] since this problem is easily found in nature and society, such as searching for odor sources, rescuing victims, and detecting electromagnetic sources, to name a few. In order to deal with this problem, some control strategies have been proposed [6,8,9]. For example, in [8], Song et al. used a probability map to label the positions of signal sources and designed a controller to enable the robot to move toward the signal source. In [6], Lu et al. used particle filters to estimate the position of a signal source and designed movement for the behavior of the leader to search for signal clues.

It should be pointed out that these control strategies consist of a decision level and a control level. At the decision level, source positions are estimated by the probability method or the particle filter [10], such as the grid probability method [8] and the particle filter [9]. At the control level, in terms of the estimated source position, the corresponding controller controls the robot to move toward the possible source positions. However, there exist two drawbacks to the control approaches described above. One drawback is that some specifications for signal source localization are not considered. For example, a wide search area is usually divided into Region 1, Region 2, Region 3, Region 4, and so on. In each region, the positions of obstacles and signal sources are different. Correspondingly, a task specification can be described as "The mobile robot is required to orderly visit each region and search for signal sources while avoiding obstacles". These task specifications can be expressed by linear temporal logic (LTL) [11–15]. On the other hand, the LTL specifications are viewed as the limitations for the movement trajectories of the robots. Hence, from the viewpoint of path planning [16–18], one requires the finding of a path that satisfies the LTL specifications. The other drawback is that the dynamic events are not efficiently managed, and path constraints cannot be well integrated into the above approaches. Receding horizon control (RHC) [19,20] can enable efficient management of dynamic events and integration of the trajectories constraints. Based on the RHC approach, one can obtain a control sequence by solving a finite horizon optimization problem and then apply the first control input in this sequence in the current moment. In the next moment, a

new control sequence is recalculated by the next sampling information. However, how to give a control strategy that combines the receding horizon control with the LTL specifications for signal source localization is not understood. The main challenge is how to build the cost functions in terms of the LTL specifications. Therefore, by facing the above challenges, our aims are to design an RHC approach with temporal logic specifications.

The main contributions of this paper are summarized in the following.

(1) We design a temporal logic planning approach where the task of signal source localization is transformed into a path planning task based on the product automaton. One main characteristic is that the product automaton can be built in an offline fashion, while the other characteristic is that the product automaton can be real-time trimmed based on the estimated position of the signal source according to particle filters.
(2) On the basis of the product automaton, an RHC approach with temporal logic planning is proposed such that the robot can locate signal sources while avoiding obstacles. Moreover, the generated movement trajectories meet the given LTL formula.
(3) We validate the proposed RHC approach with temporal logic planning through the simulation results and experimental results. The results reveal that our approach controls the robot to orderly find the signal sources.

The rest of the paper is organized as follows. In Section 2, we introduce the kinematics of the robot and the signal strength model. Then, the main problem of the paper is formulated. In Section 3, we propose a temporal logic planning approach by designing a product automaton where some key nodes are labeled by particle filters. In Section 4, we propose an RHC with temporal logic planning, where the corresponding movement trajectory meets a given LTL formula and guides the robot to locate all signal sources while avoiding obstacles. In Sections 5 and 6, we use Matlab simulation and real experiments to illustrate the effectiveness of the proposed control method. Finally, we give the conclusions in Section 7.

2. Preliminaries

2.1. Kinematics of Mobile Robots

The turtlebot robot, shown in Figure 1, is used to locate the signal source, and the corresponding kinematics model is described by

$$\begin{cases} \theta(k+1) = \theta(k) + \omega(k)\Delta t \\ x(k+1) = x(k) + \frac{\mu(k)}{\omega(k)}(\sin(\theta(k+1)) - \sin(\theta(k))) \\ y(k+1) = y(k) + \frac{\mu(k)}{\omega(k)}(\cos(\theta(k)) - \cos(\theta(k+1))) \end{cases} \tag{1}$$

where $[x(k), y(k)]$ are the positions of mobile robots; $\theta(k)$ is the orientation angle; $\omega(k)$ is the angular velocity at time k; $\mu(k)$ is linear velocity at time k; Δt is the sampling period; and $u_k = [\mu(k), \omega(k)]$.

Figure 1. The Turtlebot robot.

2.2. Signal Strength Model

The CC2530 module is utilized as a signal source in Figure 2. The corresponding received signal strength model is given by

$$\begin{cases} Z_k = \lfloor 10(\log_{10} c - \alpha \log_{10} d_k + \log_{10} \Phi(\phi_k)) \rfloor \\ d_k = \sqrt{(x(k)-p)^2 + (y(k)-q)^2} \\ \phi_k = \text{atan}(y(k)-q, x(k)-p) - \theta(k) \end{cases} \tag{2}$$

where Z_k is the expected signal strength; d_k is the distance between the robot and the signal source at time k; (p, q) is the source position; ϕ_k is the bearing angle; and c and β are parameters whose specific values are based on the used hardware. $\Phi(\phi_k)$ is calculated by

$$\Phi(\phi_k) = \begin{cases} 1 & d_k \leqslant d_a \\ \cos^2(\lambda \phi_k) & \phi_k \in [\pm\varphi^\circ], d_k > d_a \\ \cos^2(\lambda \varphi^\circ) & otherwise \end{cases} \tag{3}$$

where d_a is the antenna length. The specific values of parameters φ and λ are based on the used directional antenna.

Figure 2. CC2530 radio frequency module.

2.3. Problem Formulation

It should be noted that signal sources may be distributed in a wide area such that one usually requires the robot to search for signal sources from one region to another region. For example, the localization task can be described as "The whole area is partitioned into four regions. The robot should orderly search for the signal sources from one region to the other region infinitely often". In order to model the above localization task with temporal requirements, the linear temporal logic (LTL) formula can be used [14].

The LTL formula ϕ consists of a finite set of atomic propositions O, temporal operators, e.g., G (always), $\Rightarrow$ (implication), U (until), and the standard Boolean operators, e.g., $\wedge$ (conjunction), $\vee$ (disjunction), and $\neg$ (negation). Based on the LTL formula, the aforementioned search task can be described by

$$\phi = G((A \Rightarrow AUB) \wedge (B \Rightarrow BUC) \wedge (C \Rightarrow CUD) \wedge (D \Rightarrow DUA)) \tag{4}$$

where A, B, C, D denote four searching regions; AUB means that A has to hold at least until B is true; and $G\Psi$ means that Ψ is true at all positions of the infinite sequence. For example, there are two infinite sequences:

$$\begin{aligned} \mathbf{v}_1 &= AABBBBCCCDDDDA... \\ \mathbf{v}_2 &= AAABBBCBCCCDDAA... \end{aligned}$$

It is forward to see that sequence $\mathbf{v}_1$ meets formula ϕ. However, sequence $\mathbf{v}_2$ does not satisfy formula ϕ. Hence, this paper mainly deals with the following problem.

Problem 1. *For a mobile robot (1) and an LTL formula ϕ that are used to model the search task with temporal requirements, how to design an RHC controller $u(k)$ that can enable the robot to locate signal sources and avoid obstacles while ensuring that the generated trajectory meets ϕ.*

3. Temporal Logic Planning

3.1. Product Automaton

The search region is divided into $n \times n$ square cells of the same size. Let Γ be the index set of all cells. Each cell is denoted by C_q, where $q \in \Gamma$. Define a map o such that, for any position $p_k \in C_q$, we have $o(p_k) = q$. $C_{q_i} \cap C_{q_j} = \varnothing$ for all $q_i, q_j \in \Gamma$ and $q_i \neq q_j$ [12]. Then, we give the finite-state transition system for the robot as

Definition 1. *A finite-state transition system is a tuple $T = (Q_T, Q_0, \delta_T, \Pi_T, \zeta_T, \omega_T)$, where Q_T is a finite set of states representing the set of nodes in the transition system, Q_0 is the initial state set, $\delta_T \subseteq Q_T \times Q_T$ is the set of transition representing the transition relationships between states, Π_T is an observation set, $\zeta_T : Q_T \to 2^{\Pi_T}$ is the observation map, and $\omega_T : \delta_T \to \mathbb{R}^+$ is a positive weight function whose value represents the cost of transition between states in Q_T.*

It should be pointed out that if $q_i \to_T q_j$, $\omega_T(q_i, q_j) = 1$; otherwise, $\omega_T(q_i, q_j) = \infty$. An infinite sequence $\mathbf{q} = q_0 q_1 q_2 q_3...$ generated by the transition system T is accepted if and only if a given formula is satisfied. Moreover, one can use ltl2ba toolkit to convert the LTL formula to a Büchi automaton.

Definition 2. *A Büchi automaton is a tuple $B = (Q_B, Q_B^0, \Pi_B, \delta_B, F_B)$, where Q_B is a finite set of states, $Q_B^0 \subseteq Q_B$ is the set of initial states, Π_B is the input alphabet, F_B is a set with accepted states, and $\delta_B : Q_B \times \Pi_B \to 2^{Q_B}$ is the observation map.*

Further, one can construct the product automaton according to the finite-state transition system and the Büchi automaton.

Definition 3. *According to the system $T = (Q_T, Q_0, \delta_T, \Pi_T, \zeta_T, \omega_T)$ and a Büchi automaton $B = (Q_B, Q_B^0, \Pi_B, \delta_B, F_B)$, a product automaton $P = (Q_P, Q_P^0, \delta_P, \omega_P, F_P)$ can be constructed by $T \times B$, where $Q_P = Q_T \times Q_B$, $Q_P^0 = Q_0 \times Q_B^0$, $\delta_P \subseteq Q_P \times Q_P$ is the set of transitions defined by $[(q_i, s_i), (q_j, s_j)]$ where $q_i \to_T q_j$, $s_i \xrightarrow{\zeta(q_i)}_B s_j$ and $s_i, s_j \subseteq Q_B$, $\omega_P : \delta_P \to \mathbb{R}^+$ is the weight function defined by $\omega_P[(q_i, s_i), (q_j, s_j)] = \omega_T(q_i, q_j)$, and $F_P = Q_T \times F_B$ is the set of acceptable states.*

An infinite sequence $\boldsymbol{p} = (q_0, s_0)(q_1, s_1)(q_2, s_2)...$, where $(q_h, s_h) \in Q_P$, $(q_h, s_h) \to_P (q_{h+1}, s_{h+1})$ for all $h \geq 0$, is an accepted run of the product automaton P if and only if it starts from the initial states and intersects F_p infinitely many times. Let the set $\mathcal{D}(\boldsymbol{p}_0, \boldsymbol{p}_F)$ consist of all finite trajectories from state $\boldsymbol{p}_0$ to state $\boldsymbol{p}_F$. Then, define a set $\Omega = \{\boldsymbol{p}_F | \mathcal{D}(\boldsymbol{p}_0, \boldsymbol{p}_F), \boldsymbol{p}_0 \in Q_P^0, \boldsymbol{p}_F \in F_P\}$. Since the searching region is supposed to be connected, Ω is not an empty set. In addition, we define $\Delta \subseteq \Omega$ as an acceptable node-set and $\mathcal{D}(\boldsymbol{p}_{F1}, \boldsymbol{p}_{F2}) \neq \varnothing$ for any $\boldsymbol{p}_{F1} \in \Delta$ and $\boldsymbol{p}_{F2} \in \Delta$.

In order to effectively search for signal sources, any cell in the search area can be regarded as the initial state in the finite-state transition system T. Based the acceptable node set Δ, the corresponding product automaton can be further simplified. It should be pointed out that the above work can be performed offiline, which means that the process of generating a product automaton cannot influence the calculation performance of robots.

3.2. Particle Filter

It should be noted that all trajectories that satisfy the LTL formula can be obtained based on the product automaton P in an offline fashion for any initial cells. However, the robot is not only required to orderly visit each region but also is required to locate the corresponding signal sources in each region. Hence, the possible positions of signal sources need to be labeled in the product automaton P. In order to estimate the positions of signal sources, the particle filter can be used [21].

Since signal sources are unknown, particles are distributed evenly in the whole searching space at the beginning. The number of particles is n and the position of the ith particle at k time is p_k^i. Through the above signal strength model (2), the corresponding signal strength value of each particle can be calculated. The specific formula is as follows:

$$y_k^i = \lfloor Z_k^i - \sqrt{R} * randn \rfloor \tag{5}$$

where Z_k^i is the signal strength received by the robot at time k when the ith particle is regarded as the signal source and can be calculated by (2); R is the observed noise; and $randn$ is a random number that satisfies the normal distribution.

After obtaining the real observation value Z_k, we calculate the weight of each particle as follows:

$$\omega_k^i = \frac{1}{\sqrt{2\pi R}} \exp\left(-\frac{(y_k^i - Z_k^i)^2}{2R}\right) \tag{6}$$

Then, we normalize the particle weight as:

$$\widehat{\omega}_k^i = \omega_k^i \Big/ \sum_{i=1}^{n} \omega_k^i \tag{7}$$

According to the calculated weight, the new particle swarm is obtained by resampling. Low weight particles are sampled with low probability, and high weight particles are sampled with high probability. According to the new particle population, the estimated location of the signal source is given by

$$\overline{p}_k = \sum_{i=1}^{n} p_k^i \Big/ n \tag{8}$$

Repeat (5)–(8) after receiving a new signal strength observation. Finally, the estimation position of a signal source will converge to the corresponding real position.

If the searching area is divided into four regions in terms of the LTL formula (4), Δ_A, Δ_B, Δ_C, and Δ_D are employed to represent the node set of each region where each element in the set is also a set consisting of the states in the automaton T with the states in the automaton B and is described by $\Delta_{Aj}, j = 1, \ldots$. It is worth mentioning that these elements in sets Δ_A, Δ_B, Δ_C, or Δ_D, may change over time. Therefore, one can use Δ_A, Δ_B, Δ_C, and Δ_D, and the product automaton as a path monitor to guarantee that the generated path meets (4) and passes through the acceptable nodes.

4. Receding Horizon Control with Temporal Logic Planning

4.1. Obstacle Avoidance

It is necessary to consider how to control the robot to locate signal sources while avoiding obstacles. If there exist obstacles in the movement path of the robot, in order to

avoid obstacles, we can assume that there is a virtual robot that moves along the boundary of the rectangle obstacles. According to Lemma 4 of the reference [22], the velocity and position of the virtual robot are described by

$$\hat{p}_k^j = Zp_k + (I - Z)s_k \tag{9}$$

$$\hat{v}_k^j = Zv_k \tag{10}$$

where $p_k = [x(k),\ y(k)]$; $\hat{p}_k^j$ is viewed as the position of the jth virtual robot at time k; $\hat{v}_k^j$ can be regarded as the velocity of the jth virtual robot at time k; $v_k = [\mu(k)\cos(\theta(k)),\ \mu(k)\sin(\theta(k))]$; $Z = I - a_k a_k^T$ where a_k is the unit vector perpendicular to the boundary of the obstacle; and I is an identity matrix.

By establishing a cost function, the actual robot and the virtual robot can maintain a safe distance. Hence, the cost function is given by

$$\mathcal{F} = \sum_{j=1}^{M} \left(\alpha | \|p_k - \hat{p}_k^j\| - d| + \beta \|v_k - \hat{v}_k^j\| \right) \tag{11}$$

where α and β are the weight coefficients; there are M virtual robots; $\|\cdot\|$ denotes 2-norm; and d is the safety distance. From (11), one can see that the cost function consists of two items. The first item $|\|p_k - \hat{p}_k^j\| - d|$ enables the robots to keep a safe distance from the jth obstacle. By using the second item $\|v_k - \hat{v}_k^j\|$, the robot can move along the boundary of the jth obstacle.

4.2. Receding Horizon Control with Temporal Logic Planning

According to the above temporal logic planning, we give the basic steps of the proposed control approach.

(1) At $k = 0$, we initialize the node sets Δ_A, Δ_B, Δ_C, and Δ_D where $\Delta_{A1}, \Delta_{A2}, \ldots, \in \Delta_A$, $\Delta_{B1}, \Delta_{B2}, \ldots, \in \Delta_B$, $\Delta_{C1}, \Delta_{C2}, \ldots, \in \Delta_C$, $\Delta_{D1}, \Delta_{D2}, \ldots, \in \Delta_D$. After that, we can obtain $u^*(0|0), ..., u^*(N-1|0)$. Correspondingly, an initial position sequence $p_0^*(1), ..., p_0^*(N)$ is obtained such that the distance between $o(p_0^*(N))$ and a labeled node in Δ_{A1} is minimized.

(2) At $k > 0$, according to the position sequence $p_k(1), ..., p_k(N)$, we can obtain a sequence $q = q_k(1), ..., q_k(N)$ in T where $o(p_k(j)) = q_k(j)$ for $j = 1, \ldots, N$. Based on the product automaton, we can obtain the corresponding sequence $\boldsymbol{p} = \boldsymbol{p}_{1|k}, ..., \boldsymbol{p}_{N|k}$. Then we define $\mathcal{L}(\boldsymbol{p}_k, \boldsymbol{p}_n) = \sum_{l=k}^{n-1} \omega_P(\boldsymbol{p}_l, \boldsymbol{p}_{l+1})$. Then, we consider two cases as follows.

Case 1: $\mathcal{L}(\boldsymbol{p}_k, \boldsymbol{p}_n) > 0$, and $\mathcal{L}(\boldsymbol{p}_{l|k-1}^*, \boldsymbol{p}_n) \neq 0, l \in \{1, ..., N\}, \boldsymbol{p}_n \in \Delta_{A1}$.

$$\mathbf{u}_k^* = \arg\min_{\mathbf{u}(k)} \sum_{l=1}^{N} \sum_{j=1}^{M} \left(\alpha | \|p_k(l) - \hat{p}_k^j(l)\| - d| + \beta \|v_k(l) - \hat{v}_k^j(l)\| \right) \tag{12}$$

s.t.

$$(1), \mathcal{L}(\boldsymbol{p}_{N|k}, \boldsymbol{p}_n) \leq \mathcal{L}(\boldsymbol{p}_{N|k-1}^*, \boldsymbol{p}_n)$$
$$p_k(l) \in \mathcal{X}, v_k(l) \in \mathcal{V}, u_k(l-1) \in \mathcal{U}$$

where N is the prediction horizon; $p_k(l)$ and $\hat{p}_k^j(l)$is the lth predicted position and the jth virtual robot's position, respectively; $v_k(l)$ and $\hat{v}_k^j(l)$ are the lth predicted velocity and the jth virtual robot's velocity, respectively; and $\mathbf{u}^*(k) = u^*(0|k), \ldots, u^*(N-1|k)$.

Case 2: $\mathcal{L}(\boldsymbol{p}_k, \boldsymbol{p}_n) > 0$, and exist a $z (z \in \{1, ..., N\})$ such that $\mathcal{L}(\boldsymbol{p}^*_{z|k-1}, \boldsymbol{p}_n) = 0, \boldsymbol{p}_n \in \Delta_{A1}$.

$$\mathbf{u}^*_k = \arg \min_{\mathbf{u}(k)} \sum_{l=1}^{N} \sum_{j=1}^{M} \left(\alpha | \| p_k(l) - \hat{p}^j_k(l) \| - d | + \beta \| v_k(l) - \hat{v}^j_k(l) \| \right) \tag{13}$$

s.t.

$$(1), \mathcal{L}(\boldsymbol{p}_{z-1|k}, \boldsymbol{p}_n) = 0$$
$$p_k(l) \in \mathcal{X}, v_k(l) \in \mathcal{V}, u_k(l-1) \in \mathcal{U}$$

(3) Repeat step (2) until all elements in Δ_A have arrived. Then we use Δ_B, Δ_C, Δ_D, and Δ to replace Δ_A.

(4) Repeat steps (2) and (3).

In the following, the proposed control approach is summarized in Algorithm 1, and Theorem 1 describes the corresponding characteristics.

Theorem 1. *Consider the kinematics of robot (1). Algorithm 1 is practicable for any initial position in the application regions under an LTL formula ϕ such that there exists a movement trajectory along which the robot can avoid obstacles and find the signal sources.*

Proof. At $k = 0$, an initial trajectory can be easily generated. By induction, a sequence $\boldsymbol{x}^* = x^*(1|k-1), \ldots, x^*(N|k-1)$ at time $k-1$ can be attained. Hence, the corresponding sequence in T is $\boldsymbol{q}^* = q^*_{1|k-1}, \ldots, q^*_{N|k-1}$. According to Definitions 2 and 3, one can obtain $\boldsymbol{p}^* = \boldsymbol{p}^*_{1|k-1}, \ldots, \boldsymbol{p}^*_{N|k-1}$ in the automaton P. At time k, based on $\widehat{x}(l|k) = x^*(l+1|k-1)$, $l \in \{1, \ldots, N-1\}$, a sequence $\widehat{\boldsymbol{x}} = \widehat{x}(1|k), \ldots, \widehat{x}(N-1|k)$ can be directly constructed. When conditions satisfy Case 1, we have $\widehat{x}(N|k)$ such that $\mathcal{L}(\boldsymbol{p}_{N|k}, \boldsymbol{p}_n) \leq \mathcal{L}(\boldsymbol{p}_{N|k-1}, \boldsymbol{p}_n)$. When the conditions meet Case 2, we have a sequence $\widehat{\boldsymbol{x}} = \widehat{x}(1|k), \ldots, \widehat{x}(N|k)$ where $\widehat{x}(1|k) = x^*(2|k-1), \ldots, \widehat{x}(z-1|k) = x^*(z|k-1)$ and $\widehat{x}(N|k) = x^*(z|k-1)$. Therefore, we can conclude that there exist solutions for the proposed control at $k = 0, 1, \ldots$.

At time k, assume that the current conditions satisfy Case 1. We have $\mathcal{L}(\boldsymbol{p}^*_{N|k-\xi}, \boldsymbol{p}_n) \geq \mathcal{L}(\boldsymbol{p}^*_{N|k}, \boldsymbol{p}_n) \geq \ldots$ where $k \geq \xi \geq 0$. Since the number of nodes in P is finite, there is a time step z such that $\mathcal{L}(\boldsymbol{p}^*_{N|k+z}, \boldsymbol{p}_n) = 0$. Then, the current conditions meet Case 2. We have $\mathcal{L}(\boldsymbol{p}^*_{k+z}, \boldsymbol{p}_n) = 0$. In terms of Δ_A, Δ_B, Δ_C, Δ_D, and Δ, along with the steps to repeat, the robot can locate all the signal sources, and the trajectory satisfies the given LTL formula. Moreover, according to (12) and (13), the robot keeps a safe distance to avoid obstacles. □

Algorithm 1 Receding Horizon Control with Temporal Logic Planning (RHC-TLP).

/*Initialization*/
Initialize the particle filter and the parameters of received signal strength model (2) such as c, α, d_a, φ, and λ;
Initialize the robot' parameters and give the LTL formula ϕ;
Build a state transition system and obtain an acceptable set of nodes Δ based on the results of the division;
Initialize Δ_{A1}, $\Delta_{A2}\ldots$ in Δ_A, Δ_{B1}, $\Delta_{B2}\ldots$ in Δ_B, Δ_{C1}, $\Delta_{C2}\ldots$ in Δ_C, and Δ_{D1}, $\Delta_{D2}\ldots$ in Δ_D.
/*Main Body*/
repeat
  Receive the signal strength Z_k at the position p_k;
  Update Δ_{A1}, $\Delta_{A2}\ldots$ in Δ_A, Δ_{B1}, $\Delta_{B2}\ldots$ in Δ_B, Δ_{C1}, $\Delta_{C2}\ldots$ in Δ_C, and Δ_{D1}, $\Delta_{D2}\ldots$ in Δ_D according to the estimated positions of signal sources;
  if Δ_{Aj}, Δ_{Bj}, Δ_{Cj}, or Δ_{Dj} $(j = 1,\ldots)$ are updated **then**
    Select a node in Δ_{Aj}, Δ_{Bj}, Δ_{Cj}, or Δ_{Dj};
  else
    The given node is not changed in Δ_{Aj}, Δ_{Bj}, Δ_{Cj}, or Δ_{Dj}.
  end if
  if Case 1 is satisfied. **then**
    Solve (12) to obtain $u_k^*(0),\ldots,u_k^*(N-1)$.
  end if
  if Case 2 is satisfied. **then**

    Solve (13) to obtain $u_k^*(0),\ldots,u_k^*(N-1)$.
  end if
  if The given node in Δ_{A1} is arrived **then**
    Δ_{A1} is replaced by Δ_{A2}, $\Delta_{A3}\ldots$ in order.
  end if

  if All the given nodes in Δ_A are arrived **then**
    we orderly replace Δ_A with Δ_B, Δ_C, Δ_D, Δ.
  end if
  $u_k^*(0)$ is used as control input at time k;
until $\boldsymbol{p}_k \in \Delta$

4.3. Complexity

From Algorithm 1, one can see that the state number of the production automaton has an impact on the algorithm complexity. Let $|\phi|$ be the length of the formula ϕ where all the symbols and operators are included. Based on the LTL formula, the Büchi automaton includes, at most, $|\phi| \times 2^{|\phi|}$ states [11]. Hence, the size of the production automaton is bounded by $|Q_P| \times |\phi| \times 2^{|\phi|}$. Moreover, F_P is the number of acceptable states, and the overall complexity of the offline portion of Algorithm 1 is $O(|F_P|^3 + |F_P|^2 + (|\phi| \times 2^{|\phi|})^2 \times |F_P|)$ according to [11]. It should be pointed out that, due to constructing the product automaton offline, the computational complexity does not influence the real-time computation of Algorithm 1.

On the other hand, the in-line complexity of Algorithm 1 is highly dependent on the horizon N, the number of obstacles M and the maximal changing number of positions of signal sources δP. Let the maximal number of transitions at each state in the production automaton be δQ. Hence, the complexity of the in-line portion of Algorithm 1 is bounded by $\delta P \times (\delta Q)^{MN}$ and is determined by the used searching algorithm. Moreover, the computational latency can be relaxed by using a more efficient graph search algorithm such that the optimal trajectory can be found from exponential time to polynomial time. Further, by dividing the whole region, each region can consist of the lesser states so as to reduce the search time.

5. Simulation Results

Algorithm 1 (RHC-TLP) is implemented based on Matlab. The searching environment is 10 × 10 m and is divided into 0.25 × 0.25 m small areas. The robot detects the signal strength at a time interval of 1 second where the observed noise R is 5. Moreover, the simulation parameters are shown in Table 1. The initial position and velocity are (−4.5, 4.5 m) and (0, 0 m/s), respectively. In order to show the effectiveness of RHC-TLP, the RHC approach in [9] is used as a comparison algorithm. The main reasons are the following. The first reason is that the other signal source searching algorithms, including the RHC approach, do not deal with the case with complicated task specifications. Because of the no-planning part, the complex task is hard to finish. The other reason is that the RHC approach can deal with uncertain problems in the search process well and can be used as a typical search algorithm. Hence, the RHC approach is utilized as a comparison algorithm.

In the following, we consider two cases. In Case 1, we put a different number of signal sources in different regions. For Region A, the source number is 1 and the position is (−2.5, 2). For Region B, the source number is 1 and the position is (2, 3). For Region C, the source number is 2 and the corresponding positions are (4, −4) and (2, −2). For Region D, the source number is 1 and the position is (−2, −4). In Case 2, we place the same number of signal sources in different regions. The number of signal sources in each region is 2. For Region A, the positions are (−1, 3) and (−2.5, 2). For Region B, the positions are (4, 1) and (1, 4). For Region C, the positions are (4, −2) and (2, −4). For Region D, the positions are (−4, −1) and (−2, −4). For the two cases, the green five-pointed star indicates the real signal sources and the red dot indicates the predicted signal sources. The red circle is the starting position, and the red star is the ending position. The black blocks refer to the obstacles.

Table 1. The simulation parameters.

Parameter	Value
The control parameters α and β	2, 1
The safety distance d	0.3 m
The detection radius r	0.5 m
The prediction horizon N	6
The RSS model parameters c, α, d_a, φ and λ	0.001, 1.96, 0.2, 40°, 2

From Figure 3, the mobile robot can effectively locate all the signal sources and does not collide with the obstacles. Although there are some corner-like obstacles, the mobile robot can avoid the obstacles. If there is no signal source in a certain area, the mobile robot reaches the default node and then moves to the next area. For Case 2, shown in Figure 4, like Case 1, the mobile robot can effectively locate all the signal sources and does not collide with any obstacles. Table 2 gives the localization time and path length.

Table 2. Average (Standard error) results for localization time (s) and path length (m) in light of 30 runs for Case 1.

Algorithms	Localization Time	Path Length
RHC-TLP	117.74 (1.08)	29.53 (0.23)
RHC	259 (4.2)	37.39 (0.81)

From Tables 2 and 3, it can be seen that, with the increase in the number of signal sources, the localization time and path length obtained by the RHC-TLP approach also increase. In addition, from the standard error, the fluctuation range of the error is relatively small, which indicates that the proposed RHC-TLP approach is relatively stable. Moreover, in contrast to the RHC approach, the proposed RHC-TLP approach can obtain a lesser

localization time and path length. The main reason is, due to the automaton planning, the proposed RHC-TLP approach can easily find the signal sources from one region to the other region, while the RHC approach needs to adjust the movement direction according to the possible signal sources that result in the repeated movement of robots. Hence, the performance capability of the proposed RHC-TLP approach can be well manifested.

Table 3. Average (Standard error) results of localization time (s) and path length (m) in light of 30 runs for Case 2.

Cases	Localization Time	Path Length
RHC-TLP	141.74 (5.53)	31.37 (0.58)
RHC	288.8 (4.5)	39.88 (0.85)

From Figures 3 and 4, it can be seen that the obstacles can be avoided, and all the signal sources can be found. At the same time, from Figure 5, it can also be seen that there is a certain error between the estimated position and the real position, and the average localization errors become big while increasing the source number.

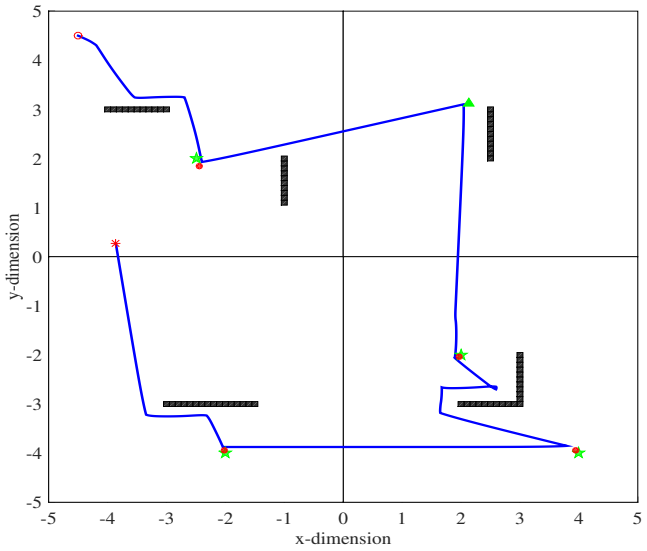

Figure 3. The robot's movement path for Case 1.

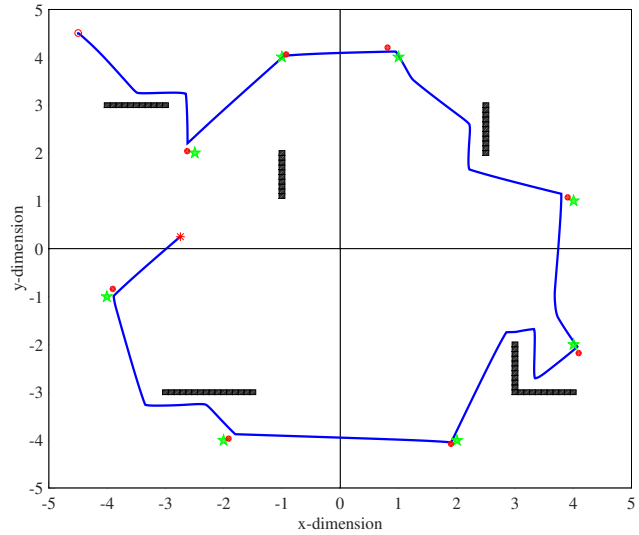

Figure 4. The robot's movement path for Case 2.

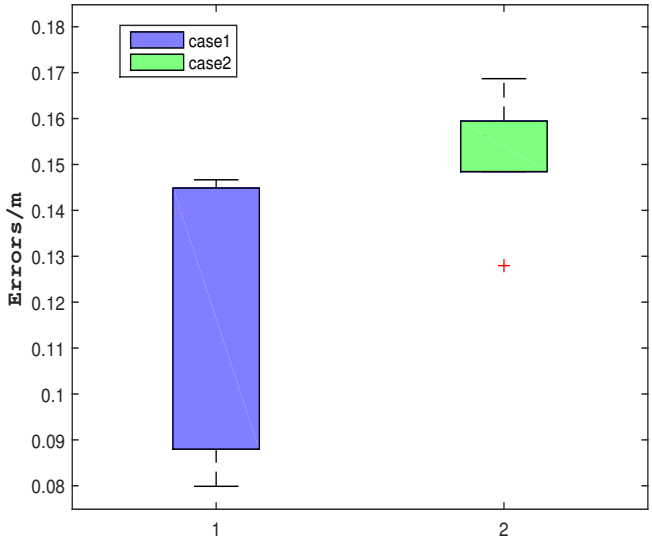

Figure 5. The average localization errors.

6. Experimental Results

In what follows, the effectiveness of Algorithm 1 is validated for signal source localization. The real testing environment is shown in Figure 6.

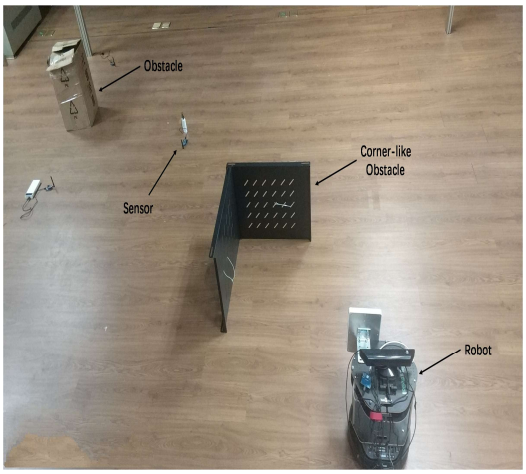

Figure 6. The experimental environment.

The number of signal sources is 3, and their positions are (0.5, 3), (2.5, 3.5) and (3, 0.5). Table 4 illustrates the average results and standard errors on localization time and path length. Figures 7–9 show the movement trajectories of the mobile robot and the predicted position of the signal sources, where the green star denotes the real source positions, and the red dot refers to the predicted source positions. The red circle is the starting position, and the black blocks are the obstacles. Hence, RHC-TLP can control the mobile robot to find all of the signal sources. From Table 3, one can see that the localization time and path length increases significantly with the number of obstacles from 0 to 2. The main reason is that the robot avoids obstacles in the movement process. Figure 10 shows the average localization errors. One can see that as the number of obstacles increases, the average localization errors also increase.

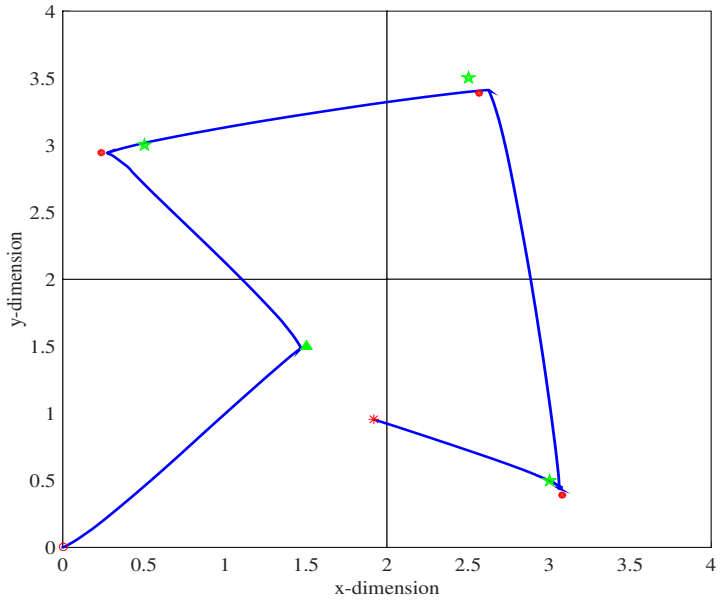

Figure 7. The robot movement trajectory in the real environment with 3 signal sources and 0 obstacles.

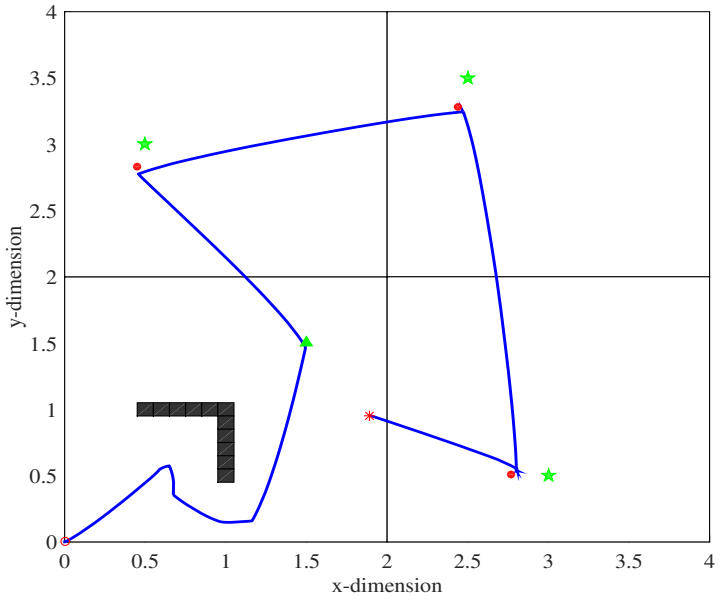

Figure 8. The robot movement trajectory in the real environment with 3 signal sources and 1 obstacle.

Table 4. Average results (Standard errors) on localization time (s) and path length (m) in light of 10 runs.

Obstacles	Localization Time	Path Length
0	77.6 (3.9)	10.23 (0.13)
1	91.6 (4.5)	10.83 (0.21)
2	96.4 (5.2)	11.05 (0.29)

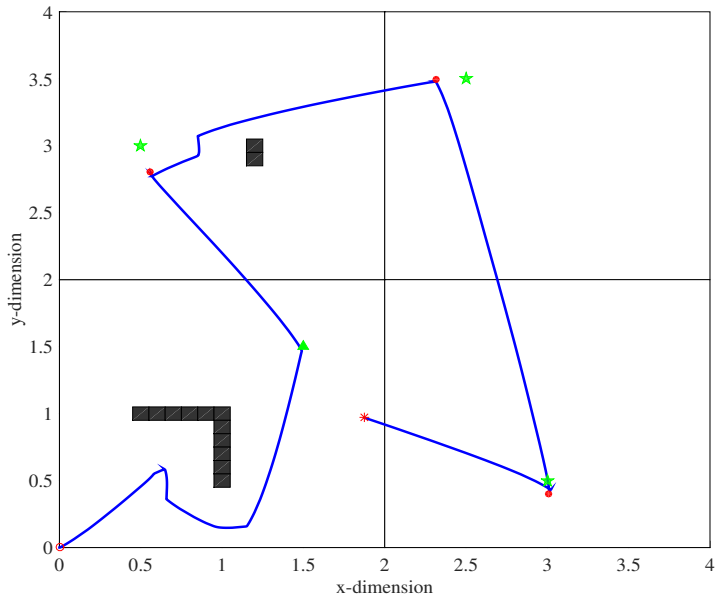

Figure 9. The robot movement trajectory in the real environment with 3 signal sources and 2 obstacles.

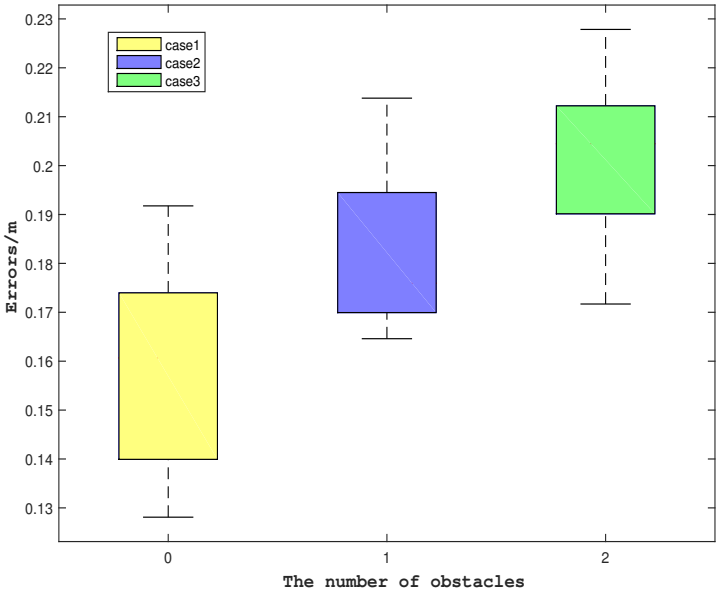

Figure 10. The average localization errors between the actual location and the estimated location.

7. Conclusions

This paper has addressed signal source localization. A temporal logic planning approach has been designed such that a product automaton is built from the temporal logic specifications and is updated based on the signal sources estimated by a particle filter. Then, an RHC with temporal logic planning has been designed, through which the mobile robot can search for signal sources and avoid obstacles, and the corresponding trajectory meets the given LTL formula. Finally, the performance capabilities of the proposed control approach have been verified through simulation and experimental results.

Author Contributions: D.C. and B.G. conducted the simulations and experiments; X.C. developed the methods and wrote the paper; Q.L. reviewed and edited the paper. All authors have read and agreed to the published version of the manuscript

Funding: This work was supported in part by the National Natural Science Foundation of China under Grant 62073108 and by the Science and Technology Project of Zhejiang Provincial Education under Grant Y202146803.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare that there is no conflict of interest regarding the publication of this paper.

References

1. Hu, Y.; Lu, Q.; Hu, Y. Event-based communication and finite-time consensus control of mobile sensor networks for environmental monitoring. *Sensors* **2018**, *18*, 2547. [CrossRef]
2. Kim, C.-Y.; Song, D.; Xu, Y.; Yi, J.; Wu, X. Cooperative search of multiple unknown transient radio sources using multiple paired mobile robots. *IEEE Trans. Robot.* **2014**, *30*, 1161–1173. [CrossRef]
3. Leonard, N.E.; Paley, D.; Lekien, F.; Sepulchre, R.; Fratantoni, D.M.; Davis, R. Collective motion, sensor networks and ocean sampling. *Proce. IEEE* **2007**, *95*, 48–74. [CrossRef]
4. Lu, Q.; Han, Q.-L.; Liu, S. A cooperative control framework for a collective decision on movement behaviors of particles. *IEEE Trans. Evol. Comput.* **2016**, *20*, 859–873. [CrossRef]
5. Pan, L.; Lu, Q.; Yin, K.; Zhang, B. Signal source localization of multiple robots using an event-triggered communication scheme. *Appl. Sci.* **2018**, *8*, 977. [CrossRef]
6. Lu, Q.; Han, Q.-L.; Peng, D.; Choi, Y. Decision and event-based fixed-time consensus control for electromagnetic source localization. *IEEE Trans. Cybern.* **2022**, *52*, 2186–2199. [CrossRef] [PubMed]
7. Ögren, P.; Fiorelli, E.; Leonard, N.E. Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment. *IEEE Trans. Autom. Control* **2004**, *49*, 1292–1302. [CrossRef]
8. Song, D.; Kim, C.Y.; Yi, J. Simultaneous localization of multiple unknown and transient radio sources using a mobile robot. *IEEE Trans. Robot.* **2012**, *28*, 668–680. [CrossRef]
9. Chen, D.; Lu, Q.; Peng, D.; Yin, K.; Zhong, C.; Shi, T. Receding horizon control of mobile robots for locating unknown wireless sensor networks. *Assem. Autom.* **2019**, *39*, 445–459. [CrossRef]
10. Cortés, J. Distributed kriged Kalman filter for spatial estimation. *IEEE Trans. Autom. Control* **2009**, *54*, 2816–2827. [CrossRef]
11. Ding, X.; Lazar, M.; Belta, C. LTL receding horizon control for finite deterministic systems. *Automatica* **2014**, *50*, 399–408. [CrossRef]
12. Wongpiromsarn, T.; Topcu, U.; Murray, R.M. Receding horizon temporal logic planning. *IEEE Trans. Autom. Control* **2012**, *57*, 2817–2830. [CrossRef]
13. Yordanov, B.; Tumova, J.; Cerna, I.; Barnat, J.; Belta, C. Temporal logic control of discrete-time piecewise affine systems. *IEEE Trans. Autom. Control* **2012**, *57*, 1491–1504. [CrossRef]
14. Tabuada, P.; Pappas, G.J. Linear time logic control of discretetime linear systems. *IEEE Trans. Autom. Control* **2006**, *51*, 1862–1877. [CrossRef]
15. Rasmussen, C.E.; Williams, C.K.I. *Gaussian Processes for Machine Learning*; MIT Press: Cambridge, MA, USA, 2006.
16. Chen, D.; Lu, Q.; Chen, Y. A Method for Solving Local Minimum Problem of Local Path Planning Based on Particle Swarm Optimization. In Proceedings of the Chinese Automation Congress & China Intelligent Manufacturing International Conference, Jinan, China, 20–22 October 2017; pp. 4944–4949.
17. Chen, Y.; Lu, Q.; Yin, K.; Zhang, B.; Zhong, C. PSO-based receding horizon control of mobile robots for local path planning. In Proceedings of the 43rd Annual Conference of the IEEE Industrial Electronics Society, Beijing, China, 29 October–1 November 2017; pp. 5587–5592.
18. Lu, Q.; Han, Q.-L. Mobile robot networks for environmental monitoring: A cooperative receding horizon temporal logic control approach. *IEEE Trans. Cybern.* **2019**, *49*, 698–711. [CrossRef]
19. Mayne, D.Q.; Rawlings, J.B.; Rao, C.V.; Scokaert, P.O.M. Constrained model predictive control: Stability and optimality. *Automatica* **2000**, *36*, 789–814. [CrossRef]
20. Yan, Z.; Wang, J.; Nonlinear model predictive control based on collective neurodynamic optimization. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 840–850. [CrossRef]
21. Arulampalam, M.S.; Maskell, S.; Gordon, N.; Clapp, T. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans. Signal Process.* **2002**, *50*, 174–188. [CrossRef]
22. Olfati-Saber, R. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Trans. Autom. Control* **2006**, *51*, 401–420. [CrossRef]