Local Strategies for Maintaining a Chain of Relay Stations between an Explorer and a Base Station

Miroslaw Dynia, Jaroslaw Kutylowski, Friedhelm Meyer auf der Heide,
DFG Graduate College, Heinz Nixdorf Institute and
“Automatic Configuration of Dynamic Intelligent Systems
in Open Systems” Heinz Nixdorf Institute and
Heinz Nixdorf Institute and
University of Paderborn University of Paderborn
Germany Germany
mdynia@upb.de jarekk@upb.de fmadh@upb.de

Jonas Schrieb
Computer Science Department
University of Paderborn
Germany
jonas@upb.de

ABSTRACT
We discuss strategies for maintaining connectivity in a system consisting of a stationary base station and a mobile explorer. For this purpose we introduce the concept of mobile relay stations, which form a chain between the base station and the explorer and forward all communication.

In order to cope with the mobility of the explorer, relay stations must adapt their positions. We investigate strategies which allow the relay stations to self-organize in order to maintain a chain of small length. For a plane without obstacles, the optimal positions are on a line connecting the base station with the explorer; in a setting with obstacles it is a curve around some of the obstacles. Our goal is to keep the relay stations as close to this line/curve as possible.

A crucial requirement for strategies is that they are able to work with imprecise or without localization and odometry information. Furthermore, strategies should be local, i.e., relay stations should not need to know about the state of the system as a whole. The performance measures for strategies are the number of relay stations used (in comparison to the optimal number) and the allowed speed of the explorer (in comparison to its maximum attainable speed).

We contribute by analyzing the performance of an already known strategy Go-To-The-Middle. This strategy assumes a very weak robot model and needs hardly any localization information, but sacrifices performance. Our main contribution is a new strategy, the Chase-Explorer strategy, and its analysis. It needs more advanced robots than Go-To-The-Middle, but achieves near-optimal performance. We further extend it to exploring terrains with obstacles.

Categories and Subject Descriptors
F.2.2 [Nonnumerical Algorithms and Problems]: Geometrical problems and computations; C.2.4 [Distributed Systems]

General Terms
Algorithms, Performance, Theory

Keywords
swarm robotics, self-organization, distributed algorithms, ad-hoc networks

1. INTRODUCTION
We envision a robotic scenario with a mobile explorer and a stationary base station. The explorer moves on a terrain with or without obstacles, guided by its own algorithm. The base station requires a stable communication link to the explorer in order to periodically communicate high-level goals and check the status of the explorer. It needs more advanced robots than Go-To-The-Middle, but achieves near-optimal performance. We further extend it to exploring terrains with obstacles.

Copyright ACM, (2007). This is the author’s version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The extended abstract of this work was published in Proceedings of the 19th Annual ACM Symposium on Parallel Algorithms and Architectures.
There are three main goals when maintaining the communication chain. First, stations neighbor in a chain should always stay in transmission distance (fixed to $R$ here). Second, the number of relay stations used should be as small as possible, since they are expected to be a rare resource. At last, the movement of the explorer should not be hindered by the chain – this might happen if the chain reacts too slowly to the movement of the explorer, and the explorer is forced to slow down in order not to exceed the distance $R$ to the first relay station.

In a terrain without obstacles, the communication chain between the base station and the explorer should optimally be arranged on a straight line. On the other hand, since the explorer moves steadily, it is impossible to maintain a really straight line all the time without using fast multi-hop communication and knowing the exact position of the explorer relatively to the base station.

A terrain with obstacles is expected to be a plane with obstacles represented by points. The situation here is more complicated. There always exists (at least one) shortest path between the current position of the explorer and the base station. It is though unrealistic that the shortest path could be maintained by any local strategy – due to obstacles, a small movement of the explorer may completely change the topology of the optimal path. Thus, the goal for our strategy is to arrange the relay stations on a path which is the shortest path between the explorer and the base station topologically equivalent to the movement path of the explorer. Fig. 2 shows an example of such an optimal, topologically restricted, path.

### 1.1 Problem Model

We model the terrain as a plane with the Euclidean distance measure. The base station, relay stations, the explorer and obstacles are modeled as points on the plane. The position of both the explorer and the relay stations may not fall into the area of any of the obstacles.

The relay stations execute their strategy in *Look-Compute-Move* steps (therefore the model is named the LCM-model). In the first operation of the step the robot gathers new information about its environment by scanning it with its sensors. In the second operation, the sensoric input is analyzed and a decision is made about the behavior of the robot within the current step. During the last operation, the robot moves to a precomputed position. Note that, in contrast to our intuition, the *Move* operation is not necessarily that one which takes the longest time. Very often, gathering the sensoric input is even more time-consuming, if for example laser scans of the surrounding must be taken. For some scenarios we will assume a slightly stronger model, than the mentioned LCM. We then allow the robots to introduce a *Communication* operation at the end of each step. A crucial property of this model is that the robots must prepare the message to be sent during the *Compute* operation and so can react on a message received only one step after receiving it.

Different types of synchronization models are typically used to describe the abilities of robots acting in a swarm. We adopt the notions by [6]. Our work is based on the $FSYAC$ model, in which all LCM-steps are fully synchronized. That means, that if a robot finishes moving earlier then others, it...
waits until the rest is finished (this may be made explicit by inserting a *Wait* operation into the definition of a step).

We bound the speed of our robots. The explorer is able to move by at most one distance unit per time step. Each of the relay stations is able to move by at least one distance unit per time step if such a movement is requested by its strategy. This can be seen as a lower bound on the movement abilities of the relay stations.

The transmission distance of the wireless transceivers is given by a constant $R > 2$. We say that two stations can communicate when the distance between them is at most $R$. This is the typical communication model known as a (unit) disc graph.

We arrange relay stations in a chain which spans from the explorer to the base station. Relay stations are denoted $v_1, \ldots, v_n$, starting with $v_1$ at the relay station next to the explorer. The last relay station in time step $t$ (the number of stations may vary depending on the time step) is denoted by $v_n$. We will often use $v_0$ to denote the explorer and $v_{n+1}$ to denote the base station. For relay station $v_i$ its neighbor in the direction of the explorer is $v_{i-1}$ and the other one is $v_{i+1}$. Let $p_t(i)$ be the position of station $v_i$ at the beginning of time step $t$. We set $b := p_t(n + 1)$ to be the position of the base station.

A strategy solving the relaying problem must ensure that a valid multi-hop communication path between the base station and the explorer is maintained all over the time, so that neighboring stations in a chain must be within distance $R$. More formally, $|p_t(i) - p_t(i-1)| \leq R$ for all $1 \leq i \leq n_t + 1$.

### 1.2 Robot model

In this paper we investigate two types of robots. For the *Go-To-The-Middle* strategy the robots are simpler and have the following abilities:

- ability to sense the relative position of other robots in distance at most $R$,
- odometry and local measurements are precise,
- no memory, the robots are oblivious,
- no communication between robots is allowed,
- all robots execute the same strategy and have no IDs.

For the *Chase-Explorer* strategy the robot type has the following abilities:

- ability to sense the relative position of other robots in distance at most $R$,
- odometry and local measurements are imprecise, with an additive error of $\epsilon$,
- the robots possess a precise compass,
- the robots possess memory,
- the robots are allowed to communicate with each other, but information received from a neighbor in step $t$ may be only propagated to the other neighbor in step $t + 1$,
- all robots execute the same strategy.

### 1.3 Performance Measures

We consider two measures for the performance of a strategy maintaining the communication chain. The first measure is the number of stations used, comparing to the number of relay stations necessary in an optimal chain.

The second measure is a guaranteed speed lower bound for the explorer. In certain types of strategies the explorer has to introduce wait cycles or to slow down, in order to allow for the communication chain to catch up. This introduces a lower bound on the average speed the explorer can attain and thus decreases the performance of the system as a whole.

### 1.4 Related Work

The amount of literature on communication in mobile wireless networks has developed significantly during last years. Most of this work considers connected networks in which the primary goal is to organize communication in the network despite the mobility.

The problem we are studying is similar to the efforts to ensure connectivity in a mobile network with the use of mobile backbones (e.g. [14]) and topology control (e.g. [10, 15, 1]). On the other hand, the concept of backbones provides only solutions for dense, connected networks, where the main goal is to select an easily manageable subset of edges/vertices from the connection graph. We are dealing with a mobile communication structure which is designed to work in larger, sparse networks, where it is necessary to control the mobility of the chain to ensure connectivity.

Similar problems related to robots with low capabilities have been considered in work on swarm robotics. Strategies have been developed to let robots gather [5, 6, 7, 2, 4] and on forming various geometrical patterns [3, 12, 13]. Note that most of this work assumes that robots are allowed to form patterns anywhere on the terrain, whereas we want them to form a line (or curve) in a specific place.

In [8] we have first presented the *Go-To-The-Middle* strategy. We have analyzed it in a setting with a stationary explorer, connected to the base station by a relay station chain forming an arbitrary curve. We have shown that starting with such an arbitrary configuration, the relay stations reach positions near to optimal after $O(n^2 \log n)$ steps, where $n$ is the number of relay stations in the start configuration. This result is obtained by reducing the problem to a stochastic process (similar to a Markov chain) and bounding its convergence speed.

### 1.5 Our Contribution

In Section 2 we provide new insights into the *Go-To-The-Middle* strategy (introduced in [8]). In particular, we provide a lower bound on its performance in a dynamic setting (where the explorer moves), showing that there are scenarios where the speed of the explorer drops to $O(1/d)$ where $d$ is
is the distance of the explorer to the base station (see Theorem 1). This shows that the most intuitive strategy for the proposed problem has serious performance drawbacks, as the speed of the explorer drops with its distance to the base station. We also show that the Go-To-The-Middle strategy is able to match that lower bound by allowing to move the explorer with an average speed of $\Omega(1/d)$ (see Theorem 7).

In Section 3 we define the Chase-Explorer strategy and analyze its performance. This strategy does not restrict the movement of the explorer at all, i.e., the explorer is able to move with its maximum speed all the time, without having to care whether the chain of relay stations keeps up (see Theorem 13). Furthermore, the strategy uses only 1.5 times the optimal number of relay stations, if the strategy is provided with a sufficiently exact localization (see Theorem 14).

We also provide a localization scheme which works with our relay station chains and provides sufficiently precise measurements for our strategies to work properly. It depends only on local, imprecise measurements between neighboring relay stations. It furthermore allows the explorer to obtain a rough estimate of of its relative position to the base station, without using any localization infrastructure (like GPS) and odometry.

Section 4 extends the Chase-Explorer strategy for a scenario with obstacles. For this scenario, we show that the the speed of the explorer is inversely proportional to the complexity of the terrain, i.e., the number of obstacles.

2. GO-TO-THE-MIDDLE STRATEGY

We briefly describe the Go-To-The-Middle strategy and proceed with the analysis of its performance. This strategy is solving the relaying problem on planar terrain without obstacles, using the simple type of robots described earlier. In the Look-operation a relay station $v_i$ observes the positions $p_i(i + 1)$ and $p_i(i - 1)$ of its neighbors. In the Compute-operation it calculates the point $p$ lying directly in the middle between $p_i(i + 1)$ and $p_i(i - 1)$ and eventually moves to that point during the last Move-operation.

Let $n_i$ be the number of relay stations in the chain and $d_i = |b - p_i|_2$ be the distance between the explorer and the base station at the beginning of time step $t$. In the Go-To-The-Middle strategy the explorer has the responsibility to manage the number of relay stations in the chain, so that is always proportional to its distance to the base station, i.e. $n_i = \gamma \cdot d_i/R$. Within this analysis we will only consider the case when the explorer moves around the base station in a fixed distance $d_i = d$, therefore we have $n_i = n$.

For a vector $w$ with $n$ elements let $|w| = \sum_{i=1}^{n} w[i]$. In contrast, we set $\|w\| = \sum_{i=1}^{n} |w[i]|$, where $|w[i]|$ is simply the absolute value of $w[i]$.

2.1 Lower bound

For the lower bound on the performance of the Go-To-The-Middle strategy we will show the following theorem.

**Theorem 1.** There exists a movement pattern for the explorer such that the Go-To-The-Middle strategy forces the explorer to slow down to a speed of $O(1/d)$.

For the proof of the lower bound we will introduce a measure $V$. We will show in Lemma 5 that a movement of the explorer by some vector $e$ around the base station increases the value of $V$ by at least $O(n \cdot e)$. On the other hand, in Lemma 6 we will show that the Go-To-The-Middle strategy is able to decrease the value of $V$ by at most some constant factor during one time step. The measure $V$ will be defined in such a way, that its value over some specific threshold implies that the relay station chain got disconnected. As the Go-To-The-Middle strategy and a properly behaving explorer guarantee that this does not happen, both the increase and decrease of the measure $V$ must balance out in the long run. Therefore, the explorer will be forced to slow down to an average of $O(1/d)$.

Let $L_t$ be the interval connecting the base station and the explorer at the beginning of time step $t$. For the construction of the lower bound we will be moving the explorer in one direction around the base station. Let us fix this direction to clockwise. Let $L_{2t}$ be a line orthogonal to $L_t$, starting at the base station. The line $L_{2t}$ divides the plane in two half-planes. The half-plane occupied by the explorer will be denoted $F_t$. The following technical lemma shows that the relay stations in the beginning of the chain are always located in $F_t$ (in appendix).

**Lemma 2.** The first $|d/R|$ stations are in $F_t$.

For the sake of simplicity of the notation let the set $S$ denote the first $|d/R|$ stations in the chain. By $V_t[i]$ we denote the distance between $p_i(i)$ and the line $L_t$ (understood in the usual ways as the length of the shortest interval between $p_i(i)$ and $L_t$). The measure $V$ is only defined for stations in $S$. Let the distance be positive if the station is on the counter-clockwise side of $L_t$ and negative otherwise. Since all stations are in $F_t$, this is well-defined. We define $V'$ as the distance between $p_{i+1}(i)$ and $L_t$. Define by $\tilde{p}_i(i)$ the projection of the position $p_i(i)$ on the line $L_{i-1}$.

Observe, that for two stations $v_i$ and $v_{i+1}$ the condition $|p_i(i) - p_{i+1}(i + 1)|_2 \leq R$ implies $|V_t[i] - V_t[i + 1]| \leq R$. For station $v_i$ in the chain it must hold $V_t[i] \leq i \cdot R$ as otherwise the chain would be disconnected. From this, the following fact follows.

**Fact 3.** In order for the stations in $S$ to keep connected it must hold

$$|V_t[i]| \leq |S| \cdot n \cdot R \leq n^2 \cdot R.$$  

The following lemma shows the easy fact that at least half of the stations in $S$ are in distance at least $d/2$ to the base station (in appendix).

**Lemma 4.** There are $|S|/2$ relay stations in $S$, such that for each of them $|b - p_i(i)|_2 \geq d/2$ and $|b - \tilde{p}_i(i)|_2 \geq d/2$. 

In the following lemma we investigate how a movement of the explorer influences the measure $\mathcal{V}$.

**Lemma 5.** Let the length of the explorer’s movement vector be $e$ in time step $t - 1$. Then

$$|\mathcal{V}_t| - |\mathcal{V}_{t-1}| \geq \Omega(e \cdot n)$$

**Proof.** Let us choose any relay station $v_i$ from set $S$. Denote by $\tilde{r}$ the distance between the base station and the projection of $p_t(i)$ on $L_{t-1}$, i.e. $\tilde{r} = |b - p_t(i)|_2$ and similarly $r = |b - p_t(i)|_2$. We will show that $|\mathcal{V}_t| - |\mathcal{V}_{t-1}| \geq \Omega(e)$, and for that purpose we have to distinguish four cases, depending on the sign of $\mathcal{V}_{t-1}$ and $\mathcal{V}_t$.

1. **Case** $\mathcal{V}_{t-1} \geq 0$ and $\mathcal{V}_t \geq 0$. For simplicity of notation let $z_{t-1} = \mathcal{V}_{t-1}$ and $z_t = \mathcal{V}_t$. Let both angles $\alpha$ and $\beta$ be rooted at $b$. Angle $\alpha$ is the angle between $L_{t-1}$ and $L_t$. Angle $\beta$ is spanned between the interval $(b, p_t(i))$ and $L_{t-1}$. From obvious geometrical observations we can give bounds on the cosine and sine of both angles, so that

$$\cos \alpha = 1 - \frac{e^2}{2d^2}, \quad \sin \alpha = \sqrt{1 - \frac{e^2}{4d^2}} \cdot \frac{\tilde{r}}{d}$$

$$\cos \beta = \frac{\tilde{r}}{\sqrt{z_{t-1}^2 + \tilde{r}^2}}, \quad \sin \beta = \frac{z_{t-1}}{\sqrt{z_{t-1}^2 + \tilde{r}^2}}.$$

Therefore

$$\sin(\alpha + \beta) = \sin \alpha \cos \beta + \cos \alpha \sin \beta = \frac{e}{d} \cdot \sqrt{1 - \frac{e^2}{4d^2}} \cdot \frac{\tilde{r}}{\sqrt{z_{t-1}^2 + \tilde{r}^2}} + \left(1 - \frac{e^2}{2d^2}\right) \cdot \frac{z_{t-1}}{\sqrt{z_{t-1}^2 + \tilde{r}^2}}.$$

In order to bound $z_t - z_{t-1}$ from below we use the trigonometrical bound on the angles.

$$z_t = \sqrt{z_{t-1}^2 + \tilde{r}^2} \cdot \sin(\alpha + \beta) = z_{t-1} + \frac{\tilde{r} \cdot e}{d} \cdot \sqrt{1 - \frac{e^2}{4d^2}} - z_{t-1} \cdot \frac{e^2}{2d^2}.$$

We can bound $z_{t-1} \leq n \cdot \mathcal{R}$ and then

$$z_{t-1} \cdot \frac{e^2}{2d^2} \leq \frac{\mathcal{R}^2 \cdot e^2}{2n} \leq O(e/n),$$

since $d = \mathcal{R} \cdot n/\gamma$. Since we have a bound on the maximum speed of the explorer of $e \leq 1$ for sufficiently large $d$ we have

$$\sqrt{1 - \frac{e^2}{4d^2}} \geq 1/2.$$ Therefore

$$\mathcal{V}_t - \mathcal{V}_{t-1} = z_t - z_{t-1} \geq \frac{\tilde{r} \cdot e}{2d} - O(e/n). \quad (2)$$

2. **Case** $\mathcal{V}_{t-1} \leq 0$ and $\mathcal{V}_t \leq 0$. Here we are dealing with a situation very similar to the former case. Laying $z_t = -\mathcal{V}_{t-1}$ and $z_{t-1} = -\mathcal{V}_t$ we can use Eq. (2) and obtain

$$\mathcal{V}_t - \mathcal{V}_{t-1} = z_{t-1} + z_t \geq \frac{\tilde{r} \cdot e}{2d} - O(e/n).$$

3. **Case** $\mathcal{V}_{t-1} \leq 0$ and $\mathcal{V}_t \geq 0$. In this case, $p_t(i)$ lies “inbetween” the intervals $L_t$ and $L_{t-1}$ as shown on Fig. 5. We will show that either $max\{-\mathcal{V}_{t-1}, \mathcal{V}_t\} \geq r \cdot e/2d$.

Let us assume without loss of generality that $\mathcal{V}_t \geq -\mathcal{V}_{t-1}$. For simplicity of notation let $z := \mathcal{V}_t$. Let $\alpha$ be the angle between $(b, p_t(i))$ and $L_{t-1}$ and $\beta$ be the angle between $(b, p_t(i))$ and $L_t$. As $\mathcal{V}_t \geq -\mathcal{V}_{t-1}$ we have $\beta \geq \frac{\pi}{4}(\alpha + \beta)$. As $e$ is reasonably small in comparison to $d$, we have $\beta \leq \pi/2$ and therefore $z = r \cdot \sin \beta \geq r \cdot \sin \frac{\pi}{4}(\alpha + \beta)$. By one of the common trigonometrical identities we have

$$z \geq r \cdot \sin \frac{\pi}{4}(\alpha + \beta) \geq \sqrt{1 - \cos(\alpha + \beta) \geq \frac{2}{2} \cdot \frac{1 - e^2}{2d^2}}.$$

From the Law of Cosines it holds $\cos(\alpha + \beta) = 1 - 2e^2/2d^2$, so that eventually $z \geq r \cdot e/2d$. Therefore we have

$$\mathcal{V}_t - \mathcal{V}_{t-1} \geq \frac{r \cdot e}{2d}.$$

4. **Case** $\mathcal{V}_{t-1} \geq 0$ and $\mathcal{V}_t \leq 0$. This is impossible to happen, since the movement of the explorer always increases $\mathcal{V}_{t-1}$. 

**Figure 3:** Case $\mathcal{V}_{t-1} \geq 0$ and $\mathcal{V}_t \geq 0$

**Figure 4:** Case $\mathcal{V}_{t-1} [i] \leq 0$ and $\mathcal{V}_t [i] \leq 0$

**Figure 5:** Case $\mathcal{V}_{t-1} [i] \leq 0$ and $\mathcal{V}_t [i] \geq 0$
Bringing all cases together we have that for any relay station \( i \) it holds
\[
V_i[i] - V'_i[i] \geq \frac{e}{2d} \min \{r, \bar{r}\} - O(e/n) .
\]
By Lemma 4 there are at least \( |S|/2 \) stations such that \( \min \{r, \bar{r}\} \geq d/2 \). This implies that for each of these stations we have \( V_i[i] - V'_i[i] \geq e \cdot d/4 - O(e/n) \). Therefore summing up over all \( |S|/2 \) such stations we obtain
\[
|V_i| - |V'_i| \geq \Omega(e \cdot n) .
\]
\[\square\]

In contrast to the previous lemma, we now investigate how much the measure \( V \) may decrease when Go-To-The-Middle is applied. The result is that it is only by a constant factor.

**Lemma 6.** It holds
\[
|V_i| - |V'_i| \leq \mathcal{R}
\]

**Proof.** Let \( D_t \) denote the distance between the line \( L_t \) and the first station not included in \( S \), i.e. \( v_{d/R+1} \). The following equation comes from the fact that a node executing the Go-To-The-Middle strategy assumes a position which, in terms of the measure \( V \), is the average of the positions of its both neighbors
\[
V'_i[i] = \begin{cases} 
0 & \text{if } i = 0 \\
\frac{1}{2} \cdot (V_i[i] - 1 + V_i[i+1]) & \forall i \in [1, \ldots, |S| - 1] \\
\frac{1}{2} \cdot (D_t + V_i[|S| - 1]) & \text{if } i = |S| .
\end{cases}
\]
For a more in-depth discussion of this property see [8]. Therefore we have
\[
|V'_i| = \sum_{i=1}^{[S]} V'_i[i] = \frac{1}{2} (D_t + V_i[|S| - 1]) + \\
\sum_{i=1}^{[S]-1} \left( \frac{1}{2} \cdot (V_i[i] - 1 + V_i[i+1]) \right) + \\
\sum_{i=2}^{[S]} \frac{1}{2} \cdot V_i[i] = \\
\frac{1}{2} (D_t + V_i[1] + V_i[|S|]) + \sum_{i=1}^{[S]-1} \text{\# } V_i[i] = \\
\frac{1}{2} |V_i| - \frac{1}{2} \cdot (V_i[1] + V_i[|S|] - D_t) \geq |V_i| - \mathcal{R} ,
\]
since \( V_i[1] \leq \mathcal{R} \) and \( V_i[|S|] - D_t \leq \mathcal{R} \). \( \square \)

After introducing bounds on the change of the measure \( V \) we are ready to prove the upper bound on the average speed of the explorer.

**Proof of Theorem 1.** We construct the input sequence by letting the explorer move by a fixed vector \( e \) in clockwise direction around the base station. We choose the start configuration so that all relay stations are on their optimal positions, so that \( |V_i| = 0 \). Note that it holds
\[
|V_{i+1}| = |V_i| + |V_{i+1}| - |V'_i| - (|V_i| - |V'_i|) .
\]
Therefore by Lemma 5 and 6 it holds
\[
|V_{i+1}| = |V_i| + t \cdot \Omega(e \cdot n) - t \cdot \mathcal{R} .
\]
As by Eq. (1) the value of \( |V_i| \) is upper bounded and therefore
\[
t \cdot \Omega(e \cdot n) - t \cdot \mathcal{R} \leq n^2 \cdot \mathcal{R} .
\]
Thus, for \( t \geq n^2 \cdot \mathcal{R} \) it holds \( e = O(\mathcal{R}/n) = O(1/d) \) if treating \( \mathcal{R} \) as constant. \( \square \)

**2.2 Upper bound**

We present an upper bound on the performance of the Go-To-The-Middle strategy, by showing that on average the explorer will not be hindered to a speed lower than \( \Omega(1/n) \). This matches exactly the lower bound stated earlier. Recall that we are only dealing with movement of the explorer in the same distance to the base station, therefore \( d = d_t \) and \( n = n_t \).

**Theorem 7.** Let the explorer move on a circle with diameter \( d \) around the base station. Then its average speed is \( \Omega(1/d) \).

As earlier for the lower bound, we need to introduce a measure \( \mathcal{U} \). For each relay station there is a position on the interval \( L_t \) it should ideally position on. More specifically, for station \( v_i \), this is position \( p^\text{opt}_t(i) \) on the interval \( L_t \) in distance \( \frac{d}{2n} \) to the explorer. These positions simply distribute the available relay stations equally over the interval \( L_t \). Let \( p^\text{opt}_X(i) \) and \( p^\text{opt}_Y(i) \) denote respectively the \( X \) and \( Y \)-coordinates of \( p^\text{opt}_t(i) \) in some arbitrary, but fixed euclidian coordinate system. As the relay stations are distributed evenly on the interval \( L_t \), the following fact follows.

**Fact 8.** It holds
\[
p^\text{opt}_X(i) = p^\text{opt}_X(i) + p^\text{opt}_Y(i) = p^\text{opt}_Y(i)
\]
for \( u = X, Y \) and \( i \in [1, n] \).

We define a family of vectors \( U_{X,i} \) and \( U_{Y,i} \), such that \( U_{X,i}[t] \) denotes the difference in the \( X \)-coordinates of \( p_t(i) \) and \( p^\text{opt}_t(i) \) and \( U_{Y,i}[t] \) does the same for the \( Y \)-coordinates. More formally we have
\[
U_{X,i}[t] = p_{X,t}(i) - p^\text{opt}_{X,i}(i) \quad \text{and} \quad U_{Y,i}[t] = p_{Y,t}(i) - p^\text{opt}_{Y,i}(i) .
\]
As both the base station and the explorer lie on the interval \( L_t \) on their proper positions, we have \( U_{u,i}[0] = 0 \) and \( U_{u,i}[n+1] = 0 \) for both \( u = X, Y \). Observe, that for two stations \( v_i \) and \( v_{i+1} \) the condition \( |p_t(i) - p_t(i+1)| \leq \mathcal{R} \) implies \( |U_{u,i}[t] - U_{u,i}[i+1]| \leq \mathcal{R} \) for both \( u = X, Y \) yielding the following fact.

**Fact 9.** For any time step \( t \) and \( u = X, Y \) it holds
\[
\|U_{u,i}\| = \sum_{i=0}^{n_t} |U_{u,i}[i]| \leq 2 \cdot \sum_{i=1}^{n_t} i \cdot \mathcal{R} \leq n^2 \cdot \mathcal{R} .
\]
Similarly to the family \( U_{t+1} \), by \( U_{t} \) we will denote the vector of distances between the positions of relay stations at the end of time step \( t \) and their optimal positions at the beginning of time step \( t \). Formally,
\[
U_{X,t}[i] = p_{X,t+1}(i) - p_{X,t}^{\text{opt}}(i),
\]
\[
U_{Y,t}[i] = p_{Y,t+1}(i) - p_{Y,t}^{\text{opt}}(i).
\]

For the purpose of proving Theorem 7 we have to show that the Go-To-The-Middle strategy decreases the value of the vectors \( U_{t} \) for \( u = X, Y \) at least by some constant factor in every step.

**Lemma 10.** Assume that the number of relay stations in the chain hasn’t changed in time step \( t \). Then for \( u = X, Y \) it holds
\[
\|U_{u,t}\| - \|U_{u,t}^t\| \geq \frac{\|U_{u,t}[1]\| + \|U_{u,t}[n]\|}{2}.
\]

**Proof.** The proof of this lemma relies on the property
\[
U_{u,t}[i] = \begin{cases} 0 & \text{if } i = 0 \\ \frac{1}{2} (U_{u,t}[i-1] + U_{u,t}[i+1]) & \forall i \in [1, \ldots, n] \\ 0 & \text{if } i = n + 1. \end{cases}
\]

This is similar to an analogical fact we have used for the measure \( V \) in Lemma 6. We have
\[
\|U_{u,t}\[i]\| = \sum_{i=0}^{n} \frac{1}{2} (|U_{u,t}[i-1]| + |U_{u,t}[i+1]|)
\]
\[
\leq \sum_{i=0}^{n-1} \frac{1}{2} |U_{u,t}[i]| + \sum_{i=2}^{n} \frac{1}{2} |U_{u,t}[i]|
\]
\[
= \frac{1}{2} \cdot \|U_{u,t}[1]\| + \sum_{i=2}^{n} \frac{1}{2} |U_{u,t}[i]| + \frac{1}{2} \cdot \|U_{u,t}[n]\|
\]
\[
= \|U_{u,t}[1]\| - \frac{1}{2} \cdot (\|U_{u,t}[1]\| + \|U_{u,t}[n]\|).
\]

Complementary to the last statement we want to show that the increase of the vectors \( U_{u,t} \) caused by the explorer’s movement is bounded from above.

**Lemma 11.** Let the length of the explorer’s movement vector be \( e \) in time step \( t - 1 \). Then for both \( u = X, Y \)
\[
\|U_{u,t}\| - \|U_{u,t}^t\| \leq e \cdot n.
\]

**Proof.** Recall that the optimal position \( p_{i}^{\text{opt}}(i) \) of the station \( v_i \) is located on the interval \( L_i \) in distance \( i \cdot d/n \) to the base station. By assumption we have \( |p_i(0) - p_{i+1}(0)|_2 = e \) and obviously \( p_i(0) \) is located on \( L_i \) and \( p_{i+1}(0) \) on \( L_{i+1} \). Take now the point \( p_{i}^{\text{opt}}(i) \) on the interval \( L_i \) and the point \( p_{i+1}^{\text{opt}}(i) \) located on the interval \( L_{i+1} \). By a geometric argument, we have \( |p_{i}^{\text{opt}}(i) - p_{i+1}^{\text{opt}}(i)|_2 \leq e \) for all \( i \in [1, n] \). This implies \( |p_{i}^{\text{opt}}(i) - p_{i+1}^{\text{opt}}(i)| \leq e \) for both \( u = X, Y \). From this we can follow that \( \|U_{u,t}[i]\| - \|U_{u,t}^t[i]\| \leq e \). Summing up over all relay station we obtain the statement of the lemma. □

Let us now create a potential function \( \Phi \). At the beginning we have \( \Phi_1 = \|U_{X,1}\| + \|U_{Y,1}\| \). We define recursively
\[
\Phi_{t+1} = \Phi_t + \sum_{u=X,Y} \|U_{u,t+1}\| - \|U_{u,t}\| - \left(\|U_{u,t}\| - \|U_{u,t}^t\|\right)
\]
\[
= \|U_{X,t+1}\| + \|U_{Y,t+1}\|.
\]

Take any movement sequence of the explorer on a circle with diameter \( d \) around the base station. The sequence can be described by a sequence of lengths of the movement vectors of the explorer \( e_1, \ldots, e_m \). We only care for the lengths of the vectors here, neglecting their direction. We assume that in a step \( t \) the explorer either moves with its maximum speed 1 (i.e. \( e_t = 1 \)), or that its movement is restricted by the relay station chain and \( e_t < 1 \). Denote by \( T \) the set of all time steps \( t \) such that \( e_t < 1 \). For each time step out of \( T \) the following important fact holds.

**Lemma 12.** There exists a constant \( \varphi > 0 \) such that for each time step \( t \) in which \( e_t < 1 \) it holds
\[
\|U_{X,t+1}[1]\| + \|U_{Y,t+1}[1]\| \geq \varphi.
\]

**Proof.** If the explorer is restricted in time step \( t \), it means that \( |p_{t+1}(1) - p_{t+1}(0)|_2 > R - 1 \). Otherwise the explorer could move for a distance of 1, without exceeding the maximum allowed distance \( R \) to \( v_1 \). Note that since there are \( m \) relay stations in the chain than would be really necessary, we have \( |p_{t+1}(1) - p_{t+1}(0)|_2 = R/\gamma \). So we have
\[
|p_{t+1}(1) - p_{t+1}(0)|_2 > R - 1 - R/\gamma,
\]
and therefore
\[
\|U_{X,t+1}[1]\| + \|U_{Y,t+1}[1]\| > R - 1 - R/\gamma.
\]

Setting \( \varphi = R - 1 - R/\gamma > 0 \) we can follow the statement of the lemma □

**Proof of Theorem 7.** The average speed of the explorer during the whole execution is given by \( \frac{\sum_{i=1}^{m} e_i}{m} \). We want to bound the average speed from below. For both \( u = X, Y \) we have by Lemma 11
\[
\|U_{X,t+1}\| - \|U_{X,t}\| \leq e_t \cdot n.
\]

Furthermore by Lemma 10 and Lemma 12 for all \( t \in S \) holds
\[
\|U_{u,t+1}\| - \|U_{u,t}\| \geq \frac{\|U_{u,t+1}[1]\| + \|U_{u,t}[n]\|}{2} \geq \varphi/2.
\]

Therefore we have
\[
\Phi_t \leq n \cdot \sum_{i=1}^{m} e_i - |S| \cdot \varphi/2,
\]

As \( \Phi_t = \|U_{X,t+1} + U_{Y,t+1}\| \geq 0 \) it must hold
\[
\sum_{i=1}^{m} e_i \geq \frac{|S| \cdot \varphi}{2n}.
\]

Since the explorer has moved by 1 in all time steps not included in \( S \) we have another bound \( \sum_{i=1}^{m} e_i \geq m - |S| \). Combining both bounds we obtain
\[
\sum_{i=1}^{m} e_i \geq \frac{|S| \cdot \varphi}{4n \cdot m} + \frac{m - |S|}{2m}.
\]
3. CHASE-EXPLORER STRATEGY

In this section we describe the Chase-Explorer strategy and provide its analysis for the obstacle-free scenario.

\[
p_{i-1}(i-1) \quad R \quad p_i(i-1) \quad p_i(i) \quad R \quad p_i(i) \quad p_{i+1}(i)
\]

**Figure 6: The Chase-Explorer strategy**

Chase-Explorer works as follows: at the beginning of a time step \( t \), relay station \( v_i \) looks at the position of the station \( v_{i-1} \), which is \( p_i(i-1) \). Relay station \( v_i \) computes the coordinates of a point which is on the interval connecting \( p_i(i-1) \) and the base station, and in distance \( R := R - 1 \) from \( p_i(i-1) \). The positions are depicted in Fig. 6. Informally speaking, the relay stations try to keep as near to the direct line connecting the base station and the explorer as possible. At the same time they try to maintain a distance of \( R \) to the previous station in the chain. This is different to Go-To-The-Middle, where stations place themselves according to the positions of their neighbors only. All relay stations move simultaneously, thus stations have to precompute their new positions during the Look and Compute parts of each step.

The movement of the explorer can change its distance to the base station. Thus, it may be necessary to change the number of relay stations in the chain. The base station decides to insert a new relay station when the last relay station \( v_n \) reaches a distance larger than \( R \) to the base station at the end of time step \( t-1 \). The new relay station is inserted on the line between the position \( p_i(n_i) \) and the base station, keeping a distance of \( R \) to the position \( p_i(n_i) \). On the contrary, if the last relay station comes too near to the base station, it is removed.

Chase-Explorer ensures by its design that the explorer is able to move freely, without being hindered by the chain. To ensure correctness of the Chase-Explorer strategy it is necessary to show that the relay stations will be able to chase the explorer without exceeding their maximum speed and that the distance limit of \( R \) won’t be exceeded between neighboring stations.

**Theorem 13.** Assume that relay stations know the precise location of the base station relative to their own position. Then, for the Chase-Explorer strategy and an explorer with maximum speed 1 it holds

- the speed of the relay stations won’t exceed 1,
- the distance between neighbors in the chain never exceeds \( R \).

After showing the correctness of the Chase-Explorer strategy we will analyze the behavior of the chain in presence of imprecise localization methods, where stations only know an approximation of the position of the base station, with an appropriately bounded additive error.

**Theorem 14.** Let \( \gamma_i(i) \) be the hop-distance of \( i \) to the base station, i.e. \( \gamma_i(i) := n_i - i + 1 \). If station \( v_i \) knows the position of the base station relative to its own position with additive error at most \( \frac{\varepsilon}{R} \cdot \gamma_i(i) \) then

\[
n_i \leq 1.5 \cdot \frac{d_i}{R-1} + 1.
\]

The following theorem shows limits for the additive error.

**Theorem 15.** If the additive error is larger than \( R \), then there is a route of the explorer and a way to fix the localization errors, so that

\[
\lim_{t \to \infty} n_t = \infty
\]

while \( d_t \) remains constant.

### 3.1 Correctness

We first investigate the correctness of the Chase-Explorer strategy assuming no localization errors.

**Proof of Theorem 13.** First we want to show that no relay station is required to move for a distance larger than 1 during one time step, providing that the explorer does not exceed its maximal speed of 1 per time step. Assume that station \( v_{i-1} \) moves a distance of at most 1 every time step. A movement of station \( v_{i-1} \) is depicted on Fig. 6 between points \( p_{i-1}(i-1) \) and \( p_i(i-1) \). The distances \( |p_i(i) - p_{i-1}(i-1)|_2 = |p_{i+1}(i) - p_i(i-1)|_2 \) are both equal to \( R \). By an obvious geometric argument the distance between \( p_i(i) \) and \( p_{i+1}(i) \) traveled by station \( v_i \) in time step \( t \) is not greater than the distance between \( p_{i-1}(i-1) \) and \( p_i(i-1) \). So, the movement distance of station \( v_i \) is bounded by 1 if station \( v_{i-1} \) has moved by at most 1.

We consider the distance between a station \( v_i \) and its neighbor \( v_{i-1} \) at the end of time step \( t \). Obviously the distance between \( p_i(i) \) and \( p_{i-1}(i-1) \) is exactly \( R \) after time step \( t \). Since the station \( v_{i-1} \) can move for a distance of at most 1 during the time step \( t \), the distance between \( p_{i-1}(i-1) \) and \( p_i(i-1) \) is at most 1. Thus, by the triangle inequality, the distance between \( p_i(i) \) and \( p_{i-1}(i-1) \) is at most \( R + 1 \). This assures that the distance between the relay station \( v_i \) and the explorer is at most \( R + 1 \leq R \) at the beginning of each time step. 

Due to space considerations, we won’t argue about the local correctness of the Chase-Explorer strategy when only imprecise approximations of the neighbor’s and base station’s
position are known to the relay stations. Instead, we turn our attention to the more interesting question on how the chain behaves globally if stations have only an imprecise approximation of their global position.

### 3.2 Unprecise Base Station Localization

In the following, we will consider the behavior of the *Chase-Explorer* strategy when the relay stations know only an imprecise estimate of the base station’s position relatively to their own position. Nevertheless, as all relay stations possess a compass, we assume that they share a common orientation of their coordinate system. The only difference is, that the position of the base station in this coordinate system is only approximated by the relay stations. Formally, we denote the position of the base station known by station \(v_i\) as \(b_i(i)\) at time step \(t\), whereas \(b\) is the real base station’s position. In the remaining part we will denote the interval or line defined by two points \(x, y\) as \((x, y)\).

It is not hard to imagine that the strategy is able to work with a localization scheme such that \(|b - b_i(i)| < \epsilon\). We wish for the chain to work properly and to attain a reasonable performance (in terms of the number of relay stations used).

Recall, that \(\gamma(i) := n_i - i + 1\). We aim at showing that a localization system such that \(|b_i(i) - b| < \epsilon \cdot R \cdot \gamma(i)\) is enough. This means, that stations which are further apart from the base station are allowed to have a larger error in their localization. Assume that the value \(n_i\) always changes at the beginning of a time step. Therefore for during step \(t\), such that \(n_i = n_{i-1} + 1\) we already have an increased value of \(\gamma(i)\) for station \(i\).

This weak accuracy brings some problems: since the accuracy depends on the number \(\gamma(i)\) and the number \(\gamma(i)\) depends on the accuracy (due to weak accuracy the chain does not follow an optimal line and additional relay stations must be introduced) one might be worried that the chain gets infinitely long, with the accuracy getting weaker pararally. Theorem 14 shows that this behavior does not occur if \(\epsilon\) is bounded sufficiently. Theorem 15 shows the contrary, i.e. that a localization system with \(\epsilon\) large can lead to unstable behavior.

**Proof of Theorem 14.** Let \(r_i(i) := |b - p_i(i)|_2\) be the distance of the station \(v_i\) to the base station in time step \(t\). Then we define

\[
u_i(i) := |r_(i-1) - r_i(i+1)|
\]

\[
u_i(i) := |r_i(i) - r_i(i+1)| - u_i(i) + 1
\]

Let \(\alpha_{i-1}(i)\) be the angle at \(p_{i-1}(i)\), between the intervals \(b_i, p_{i-1}(i)\) and \(b_i(i+1), p_{i-1}(i)\). With other words \(\alpha_{i-1}(i)\) is the angle, measured at \(p_i(i)\) between the real position of the base station and the approximation of the base station’s position known by \(v_i\).

Let \(|b - b_i(i+1)|_2 \leq \epsilon(i+1) = \epsilon \cdot R \cdot \gamma(i)\). We introduce three lemmas, which relate the values of \(u, \alpha, r\) to each other. The technical proofs can be found in appendix.

**Lemma 16.** For every \(i + t\) such that \(\gamma(i) > 2\)

\[u_i(i) \geq R \cdot (\cos(\alpha_{i-1}(i)) - \sin(\alpha_{i-1}(i)))\]

**Lemma 17.** For every \(\alpha_{i-1}(i)\)

\[
\cos \alpha_{i-1}(i) \geq \sqrt{1 - \frac{\epsilon_i(i+1)^2}{\gamma_{i-1}(i)}}
\]

\[
\sin \alpha_{i-1}(i) \leq \frac{\epsilon_i(i+1)}{\gamma_{i-1}(i) - \epsilon_i(i+1)}
\]

**Lemma 18.** Let \(u \leq u_i(i)\) and \(u \geq 3\) for all \(i, t\) and \(i\) such that \(i \leq n_i - 2\). Then for all \(t\) and \(i\) such that \(i \leq n_i - 2\) it holds

\[\gamma_{i-1}(i) \geq \frac{1}{3} \cdot \gamma_i(i+1) \cdot u\]

Take any time step \(t - 1\). Let \(u = 0.7 \cdot R\). We want to show that \(u_i(i) \geq u\) for all \(i\) such that \(\gamma(i) \geq 1\) if \(u \leq u_i(i+1)\) and \(R \geq 5\). For the station standing next to the base station we assume that it may be at any distance to the base station, therefore \(u_i(n_i - 1)\) may be as low as 0.

Bringing together Lemma 16, 17, and 18 we obtain the following lower bound

\[u_i(i) \geq R \cdot \left(\sqrt{1 - \frac{9 \cdot \epsilon^2 \cdot R^2}{u^2}} - \frac{\epsilon \cdot R}{1/3 \cdot u - \epsilon \cdot R}\right)\]

To prove our claim it should hold

\[R \cdot \left(\sqrt{1 - \frac{9 \cdot \epsilon^2 \cdot R^2}{u^2}} - \frac{\epsilon \cdot R}{1/3 \cdot u - \epsilon \cdot R}\right) \geq u\]

Plugging in \(u = 0.7 \cdot R\) the above holds for all \(\epsilon \leq 1/25\). Therefore we have \(u_i(i) \geq 0.7 \cdot R\) for all \(i\) and \(t\) such that \(\gamma(i) \geq 1\).

For the sake of contradiction assume now that there is a time step such that the number of relay stations used by *Chase-Explorer* exceeds \(1.5/d_i/|\gamma| + 1\) (recall that \(d_i\) is the distance of the explorer to the base station). By our previous considerations this would mean that \(d_i \geq 0.7 \cdot R \cdot (1.5/d_i/|\gamma| + 1) \geq 1.05 \cdot d_i > d_i\), which is clearly a contradiction. □

**Proof of Theorem 15.** The distance of relay station \(v_i\) to the base station clearly cannot exceed \(R \cdot \gamma(i)\). So, if we have \(\epsilon > R\) the adversary can select a position \(b_i(i)\) all around station \(v_i\). This allows the adversary to completely control the shape of the chain. More specifically, it can create an infinitely long chain with stations such that \(r_i(i) \geq r_i(i-1)\). This part of the chain increases its distance to the relay stations instead decreasing it. This shows that for \(\epsilon > R\) the chain may increase to an infinite length, while the explorer remains in the same distance to the base station. □

### 3.3 Localization Scheme

We design a localization scheme such that *Chase-Explorer* can work without any GPS-like system. The only requirement is that stations can measure the positions of their local
neighbors with some precision \( \epsilon \). Then we can design a localization scheme with an additive error of

\[
E := \frac{\epsilon}{1 - 1/(0.7 \cdot R)} \cdot \gamma_t(i).
\]

For simplicity, let us fix the position of the base station \( b \) to the center of the coordinate system. Let \( \tilde{p}_t(i) \) be the position of station \( v_t \) as \( v_t \) approximates it. Note that this notation is equivalent to that used previously, where a station knew of station \( v_t \)'s position in each time step. Relay station \( v_t \) can then measure its position relatively to \( v_t \) and compute \( \tilde{p}_{t+1}(i) = (i-1) \). If \( \|\tilde{p}_{t+1}(i) - p_t(i)\|_2 < j \cdot \epsilon \) and the local measurement has an error of \( \epsilon \), then

\[
\|\tilde{p}_{t+1}(i) - p_{t+1}(i-1)\|_2 < (j+1) \cdot \epsilon.
\]

This scheme works continuously, so that a station has a new approximation of its position in each time step.

Base station \( v_t \)'s approximation in time step \( t \) depends on a message which went \( j \) hops from the base station to \( v_t \). Unfortunately \( j \) must not be necessarily equal to \( \gamma_t(i) - 1 \). During the time when the message traveled from the base station to \( v_t \), several relay stations might have been removed. On the other hand, by Theorem 14 it holds \( \omega_t(i) \geq 0.7 \cdot R \) and so in \( j \) time steps only \( j/(0.7 \cdot R) \) relay stations might have been removed. Thus, \( j - j/0.7 \cdot R \leq \gamma_t(i) \) and the error of the localization scheme is at most \( E \).

Note that this scheme guarantees that the explorer will never get lost in terrain, even if there is no infrastructure for localization (GPS) and its own odometry cannot be relied on. Thus, the communication chain can be even used for having some (although imprecise) localization.

4. TERRAIN WITH OBSTACLES

Let us recall the model we are dealing with. We are concerned with a plane with obstacles represented by points. We are aiming at maintaining a chain of relay stations which is as short as possible but also topologically equivalent to the path traveled by the explorer. We are using communication between relays to coordinate their actions. We though assume a very weak communication model: messages may travel only one hop per time step, which means that the messages have a propagation speed proportional to the movement speed of stations.

We will denote the movement sequence executed by the explorer as \( \sigma \). The Chase-Explorer strategy will be able to put the explorer into wait mode, in which it stops executing \( \sigma \) and waits for Chase-Explorer to finish some maintenance. As Chase-Explorer is deterministic, we allow the adversary to generate \( \sigma'[t] \) basing on the behavior of Chase-Explorer for \( \sigma[1], \ldots, \sigma[t-1] \).

The basic idea for Chase-Explorer is that relay stations setup navigation points (or navpoints for short) on the position of obstacles. These work as a base station for the part of the chain which lies behind them (in the direction of the explorer). Such a situation is depicted on Fig. 7. It is natural for the stations behind the obstacle to assume that the navpoint is the base station appropriate for them as long as the explorer stays in the validity area. Relay stations employed in the chain use the standard Chase-Explorer strategy, while exchanging messages about new navpoints and changing the coordinates of the reference navpoint appropriately.

Unfortunately, even if each station passes the notice about the creation of a new navpoint as fast as possible, the chain is brought out of order. This is the case, since during the time which passed between establishing the navpoint and receiving information about this fact, a station \( v_t \) performs its movements aligning to its old reference navpoint. This can cause the chain to become unnecessarily longer. Furthermore, during this time relay stations are heading in a wrong direction and may hit an obstacle that they usually won’t hit. This causes new navpoints to be established, the chain being extended even more — we have to ensure that the chain length nevertheless stays bounded.

The main result of the analysis of Chase-Explorer for a terrain with obstacles is presented in Theorem 19. For its understanding we first have to introduce some notation.

The navpoint which station \( v_t \) uses as its base station will be called \( v_t \)'s reference navpoint. Wherever this does not cause any ambiguity, we will treat the explorer and the base station as navpoints. By \( \delta_t \) we will denote the number of stations employed in the chain in time step \( t \).

We will partition the execution of the input sequence into epochs. Each epoch will consist of a certain number of steps of execution of the input sequence \( \sigma \) (normal phase), followed by some wait steps (maintenance phase). Wherever it is clear from context which epoch is meant, we will denote \( \delta = \delta_t \), where \( t \) is the first time step of epoch \( E_q \). The normal phase of an epoch has duration \( \delta \). During the normal phase we won’t care much about the shape of the chain around obstacles — the relay stations will do what they can to minimize the length of the chain, but we will allow for some losses here. In the maintenance phase we will be bringing the chain to an optimal order. Whenever epoch \( E_q \) has ended, the next epoch \( E_{q+1} \) starts afterwards in the next
time step.

4.1 Navpoints and Terrain Complexity

We assume that stations are able to detect obstacles and setup navpoints where the chain hits an obstacle. The chain may hit an obstacle during its movement in the middle of a time step – for simplicity of description we assume that the station is able to detect that situation when it occurred during a step, setup the navpoint and start to behave accordingly with regard to the navpoint for the rest of the time step. Note, that this is not a violation of the LCM-model, since a station may have recognized the obstacle already at the beginning of the step and planned its movement according to the projected time of hitting the obstacle. After a time step \( t \) a navpoint setup by \( v_i \) will be always on the interval \( [p_t(i - 1), p_t(i + 1)] \). The station \( v_i \) which established a navpoint is responsible for maintaining it first, i.e. remembering the position of the navpoint. Station \( v_i \) may pass the information associated with the navpoint to \( v_{i-1} \) or \( v_{i+1} \) if one of these stations becomes responsible for maintaining the navpoint. There are a few rules governing the behavior of relay stations located next to the navpoint.

- Station \( v_i \) maintaining navpoint \( s \) positions itself in time step \( t \) on the interval \( [s, s_r] \), where \( s_r \) is the current reference navpoint of \( v_i \), in distance \( R - |s - p_t(i - 1)| \). This means, that the station is ensuring that it holds \( |s - p_t(i - 1)| \) or \( |s - p_t(i - 1)| = R \). By the triangle inequality \( |p_t(i - 1)| = |p_t(i - 1) - R| \leq R \).

- When station \( v_i \) maintains navpoint \( s \) and it happens that \( |s - p_t(i - 1)| = R \) then station \( v_i \) passes the responsibility for maintaining \( s_r \) to \( v_{i+1} \), updates its reference navpoint to \( s \) and starts to follow \( v_{i-1} \) in the usual way. This may be imagined as station \( v_i \) passing the navpoint.

- If station \( v_i \) maintains navpoint \( s \) and it holds \( |p_t(i - 2) - s| \leq R \), then the responsibility for maintaining \( s \) is passed over to \( v_{i-1} \) in time step \( t \). Furthermore, station \( v_{i-1} \) updates its reference navpoint to that used by \( v_i \). In that case \( v_{i-1} \) is passing the navpoint.

The terrain complexity for each epoch depends on the number of obstacles in the activity area of the explorer in this epoch. We define the activity area. Let \( A_t(s) \) be the area enclosed by the triangle \( \Delta(s, p_t(0), p_t(1)) \). Let \( N_0 \) denote the normal phase of epoch \( E_q \). Then we define \( A_q(s) = \bigcup_{s \in N_0} A_t(s) \). Let \( S = s_1, \ldots, s_k \) be the navpoints present at the beginning of \( E_q \), whereas \( s_k \) is the base station. Let \( s_1 \) be the navpoint with smallest ID, such that all navpoints \( s_1, \ldots, s_{i-1} \) are in its validity area (the validity area of a navpoint is shown in Fig. 7) and the explorer has not moved outside of its validity area during \( N_q \). Then \( k_q \) denotes the number of obstacles within \( \bigcup_{s \in \{s_1, \ldots, s_{i-1}\}} A_t(s) \).

Now we are ready to state our main result.

**Theorem 19.** Assume that the explorer was allowed to move \( \delta \) time steps during an epoch. Then the number of waiting time steps in this epoch is \( O(R \cdot \delta \cdot k_q) \). For any time step in an epoch, the number of relay stations used within the chain is at most \( (1 + 1/R) \cdot \delta \).

A chain layout \( C \) is defined by the sequence of navpoints it uses and by the orientation of the chain around the corresponding obstacles. Formally, a layout is defined by a sequence \( \langle s_1, z_1 \rangle, \ldots, \langle s_i, z_i \rangle, \ldots, \langle s_k, z_k \rangle \) where \( s_i \) describes the navpoint and \( z_i \) is right if the chain passes the obstacle \( s_i \) on the right side of the line \( \langle s_{i+1}, s_i \rangle \) and \( z_i \) is left otherwise. Navpoints are indexed in the same direction as the relay stations, so that \( s_0 \) defines the explorer and \( s_k \) the base station. Two layouts \( C \) and \( C' \) are topologically equivalent under a set of obstacles \( S \) if starting at layout \( C \) the relay stations employed in the chain can move their positions so that they form layout \( C' \) without having to go over any obstacle from \( S \) (which we do not allow generally). If \( S \) is the complete set of obstacles, then the two layouts are topologically equivalent.

When a navpoint is established by a station, no further notice about this fact is issued to other stations. All stations behind the navpoint still use their old reference navpoint, only those which directly pass the navpoint start using it as their reference navpoint as described earlier. This may appear counterproductive now – we might have as well sent a message which notifies stations about the navpoint – but it keeps the description of the strategy clean and concise and does not cause any significant loss in performance. It has to be noted, that the chain still remains connected if stations use different (and possibly wrong) navpoints. It just makes the chain longer than necessary.

4.2 Maintenance Phase

During the normal phase our strategy has pretty well ignored obstacles. It only considered them as much as to stay connected. During the maintenance phase we have to bring the chain back to order. We aim at achieving an optimal chain layout at the end of the maintenance phase. The base station is responsible for guiding the general layout of the chain in the maintenance phase. For this purpose it collects information about obstacles in relevant areas of the terrain and recomputes an optimal layout of the chain, basing on the partial information about obstacles already known. During the reorganization of the chain basing on the specifications by the base station new obstacles may be encountered which were not known before. This new encounters are communicated to the base station, which is then able to reorganize the chain according to new information.

Denote the layout of the chain at the beginning of the maintenance phase by \( C \). Assume first that the base station knows the positions of all obstacles on the terrain. Then the base station has the possibility to compute a layout for the chain, which is topologically equivalent to \( C \) and which has the shortest possible length. Unfortunately, the base station does not have this full information about the obstacles available. At the beginning of a maintenance phase we may only assume that it knows obstacle positions from previous epochs and the obstacles which caused the chain to setup navpoints in the current epoch. Basing on this partial information, the base station may also compute a layout \( C' \) for the chain, which is topologically equivalent to \( C \) under
the set of known obstacles (i.e. \( C \) and \( C' \) are topologically equivalent only if there are no obstacles on the way between \( C \) and \( C' \)). Otherwise, \( C' \) is infeasible. In this situation encountering a new obstacle provides the base station with new information about the terrain and it update its layout.

The question on how the base station can compute a new layout is treated in Section 5.

The chain of relay stations obtains information about a new layout from the base station. It has to reorganize by letting each station update its reference navpoint and align itself w.r.t. this reference navpoint to its predecessor. This is performed starting at the explorer and proceeding in the direction of the base station. If a station has to realign, it moves from one position to another on a circle with diameter \( 2R \), therefore requiring at most \( 2R \) time steps for this operation. Thus, the reorganization of the chain requires \( 2R \cdot (1 + 1/R) \cdot \delta \) time steps.

If the reorganization of the chain fails as new obstacles turn out on the way, the base station has to compute a new layout and a new reorganization is performed. This must be repeated at most \( k_q \) times, since afterwards the base station knows all obstacles which the chain can encounter. Therefore the maintenance phase takes at most \( O(R \cdot \delta \cdot k_q) \) time steps.

### 5. Computing Shortest Layout

The problem of computing a new chain layout by the base station can be conveniently expressed as the Shortest Polyline Problem, defined below.

**Problem Definition.** The input consists of a set of points \( F \) on a plane and a polyline \( C \). A polyline is equivalent to a chain layout. Although the vertices of the polyline consist of the points of \( F \), we define a point to lie conceptually on one of the sides of the polyline. Therefore a polyline is given by a sequence \((f_i, z_i)\) where \( f_i \) defines a point from \( F \) and \( z_i \) is right if the point lies on the right side of \((f_{i-1}, f_i)\) and \( z_i \) is left if it lies on the left. For the start and end point of the polyline \( z_1 = \emptyset \) meaning that the points are directly on the polyline and the polyline has no orientation with respect to them.

For the notion of topological equivalence of two polylines the idea of points lying on either the left side or right side is essential. For that purpose imagine, that the points from \( S \) lie not directly on the polyline but infinitely near to the polyline on the side determined by the values of \( z \). Two polylines \( C \) and \( C' \) are topologically equivalent iff the corresponding chain layouts are equivalent.

The length of a polyline is measured in the sum of the length of its edges. In the Shortest Polyline Problem we are looking for a polyline \( C' \) which is topologically equivalent to \( C \) and such that there exists no other polyline topologically equivalent to \( C \) which has shorter length. Recall, that we have used \((a, b)\) to denote the interval defined by the points \( a \) and \( b \). This interval includes also the points \( a \) and \( b \). By \( ]a, b[ \) we will denote the same interval excluding point \( a \). Analogously \((a, b)\) (excludes point \( b \)) and \( ]a, b[ \) (excludes both endpoints).

**Topological Equivalence.** We want to formalize the notion of topological equivalency in a way that is convenient for our purposes. Therefore let \( C = (\langle f_1, z_1 \rangle, \ldots, \langle f_k, z_k \rangle) \) be a polyline. Let \( L \) be the line \((f_1, f_k)\). We let the line \( L \) partition the polyline into segments \( S_1, \ldots, S_m \), so that \( \bigcup_{i=1}^m S_i = C \) and two points \( f_i \) and \( f_{i+1} \) belong to the same segment iff and only if the line \( L \) does not cross the interval \((f_i, f_{i+1})\) (meaning that it crosses the interval excluding \( f_{i+1} \)).

Each segment represents a polyline part. The curve formed by the polyline part \( S = (\langle f_1, z_1 \rangle, \ldots, \langle f_{k'}, z_{k'} \rangle) \) and the line \( L \) enclose a part of the area \( A \) of the plane. The subset of \( F \) which is inside of this area will be denoted \( F(S) \). For those points from \( F \) which lie on the polyline part \( S \), a special condition holds. If \( z_i \) is the same as the side on which \( A \) lies with respect to \((f_{i-1}, f_i)\), then \( f_i \) is included in \( F(S) \). Otherwise, it is not. If \( z_i = \emptyset \) then the point is not included in \( F(S) \).

**Definition.** Let \( S_1, \ldots, S_k \) be the segments of polyline \( C \) and \( S'_1, \ldots, S'_{k'} \) those of \( C' \). Then \( C \) and \( C' \) are topologically equivalent if and only if the first points of \( S_i \) and \( S'_{i} \) match, the last points of \( S_k \) and \( S'_{k'} \) match, \( k = k' \) and \( F(S_i) = F(S'_i) \) for all \( i \in [1, \ldots, k] \).

**Outline of Solution.** The algorithm which computes the shortest polyline works by computing part of the solution (by defining an instance of the Shortest Restricted Circumference problem), constructing a new, smaller instance of the Shortest Polyline problem and executing itself recursively on that problem. The shortest polyline returned by the recursive call is merged with the previously computed part of the solution constituting the final solution.

We discuss now the Shortest Restricted Circumference problem and continue with the investigation of the original Shortest Polyline problem later.

### 5.1 Shortest Restricted Circumference

In this problem we are given two sets of points \( P_1 \) and \( P_0 \). The goal is to find the shortest circumference of the points \( P_1 \), which does not include any of the points from \( P_0 \). This is somehow similar to computing a convex hull around \( P_1 \) with the additional restriction of not including any of the points from \( P_0 \). The points from \( P_0 \) may lie directly on the circumference. One can easily anticipate that the solution of this problem can be used to compute the shortest polyline inside of one segment.

**Definitions.** Let \( p \) be the lowest point from \( P_1 \) (in case of ties the leftmost is taken). Then we fix some orientation and sort the points from \((P_1 \cup P_0) \setminus p\) ascending by their angle with respect to this orientation. In case there are two or more points which have the same angle and are all in \( P_1 \) then all but that with the largest distance to \( p \) are deleted. The same is performed if there are more than one points from \( P_0 \) with the same angle – then all but that with the smallest distance to \( p \) are deleted. In case there are two points one from \( P_1 \) and one from \( P_0 \) left, ties are broken arbitrarily which comes first. Let \( p_1, \ldots, p_n \) denote the sequence of points ordered as described.
Let the polyline \( C_r \) connect the points \( p_1 \) and \( p_r \). Let \( CP_r \) be the polygon formed by closing \( C_r \) with \((p, p_1)\) and \((p_r, p)\) and \( A \) be the area in the inside of \( CP_r \), excluding the actual polygon border \( CP_r \). Polyline \( C_r \) is the shortest \( r \)-partial circumference if there is no shorter polyline connecting \( p_1 \) and \( p_r \) such that:

- For each point \( p_j \in P_1 \) such that \( j \leq r \) it holds \( p_j \in A \cup CP_r \),
- For each point \( p_j \in P_o \) such that \( j \leq r \) it holds \( p_j \notin A \).

This definition can be extended to allow also for a \( n + 1 \)-partial circumference which connects \( p_1 \) to itself. Obviously this is the complete circumference we are looking for.

For simplification we assume that a polyline cannot have three consecutive collinear vertices. Let \( \triangle(a, b, c) \) denote the triangle formed by the points \( a, b, \) and \( c \).

Let \( C \) be some polyline, such that \( p_s, p_o, p_t \in P_1 \cup P_o \) are three consecutive vertices. By \(|C|\) we will denote the length of the polyline \( C \). We first want to show that the shortest \( r \)-partial circumference is a polyline whose vertices consist only of points from \( P_1 \cup P_o \). Before that we introduce a technical lemma, whose proof can be found in the appendix.

**Lemma 21.** Let the points \( A, B, C \) define a triangle. Let \( D \) be a point on the interval \((A, B)\) (see Fig. 8). Then

\[
|D - A|_2 + |D - B|_2 \leq |C - A|_2 + |C - B|_2. \tag{3}
\]

**Lemma 22.** Each vertex of the shortest \( r \)-partial circumference is a point from \( P_1 \cup P_o \).

**Proof.** For the sake of contradiction, assume there exists a vertex \( p_y \) of the polyline which is not in \( P_1 \cup P_o \). Let \( p_s \) and \( p_t \) be its neighbor vertices in the polyline. Let \( p_z \in P_1 \cup P_o \) be the nearest vertex to \( p_y \) lying on the line segment \( \triangledown p_s, p_y, p_t \). We will now consider the triangle \( \triangle(p_s, p_y, p_t) \). Decrease the angle at \( p_z \) in the discussed triangle by \( \epsilon \), defining a the triangle \( \triangle(p_y, p_y', p_z) \). By choosing an appropriately small value of \( \epsilon \) we can ensure there are no vertices from \( P_1 \cup P_o \) in the inside of the triangle \( \triangle(p_y, p_y', p_z) \), as there are no such vertices on the interval \( \triangledown p_y, p_y' \). Therefore by adding the vertex \( p_y \) to the polyline and replacing \( p_y \) with \( p_y' \) we have constructed a polyline of smaller length which is a valid \( r \)-partial circumference.

From the previous lemma and the fact that \( p \) is the lowest point from \( P_l \) it follows that \( p \) must be a vertex of the circumference.

**Algorithm.** Our algorithm will compute the shortest \( r \)-partial circumference in the \( r \)-th round of its execution. We will treat the polyline \( C_r \) as a sequence starting with \( p_1 \) and ending with \( p_r \).

```
proc ComputePartialCircumference(r)
    A_f ← ∅, A_o ← ∅
    CC ← ∅
    for i ← r downto 1
        (..., p_i) ← C_i
        if (∃ p_r ∈ A_f, p_z ∈ ∆(p_i, p_r, p))
            and(p_z ∈ A_o, p_z ∉ ∆(p_i, p_r, p))
            CC ← CC ∪ {C_i ∪ (p_r)}
        if p_r ∈ P_l
            A_f ← A_f ∪ {p_r}
        if p_r ∈ P_o
            A_o ← A_o ∪ {p_r}
    return shortest circumference out of CC
```

While computing \( C_r \) the algorithm checks whether it can connect \( p_r \) directly to \( p_1 \) and use \( C_i \cup (p_r) \) as a \( r \)-partial circumference. Therefore it is guaranteed by design that all polylines stored in the set \( CC \) are valid \( r \)-partial circumferences. At the end of the loop, the set \( CC \) contains all valid \( r \)-partial circumferences, and so choosing the shortest one yields the solution.

We still have to investigate the runtime of our algorithm.

**Lemma 23.** The shortest restricted circumference can be computed in time \((|P_l| + |P_o|)^2\).

**Proof.** The algorithm needs \(|P_l| + |P_o|\) rounds to compute the complete circumference. In each of the rounds the loop in the `ComputePartialCircumference` procedure iterates at most \(|P_l| + |P_o|\) times. While a trivial implementation of the check in line 5 would cost time linear in \( A_f \cup A_o \), this can be implemented to be performed in constant time. The lemma follows.

**5.2 Merging segments**

We introduce the segment merging operation, which will help in the later algorithm.

For this purpose we will say that a segment is on the left of \( L \) if the points in \( F(S) \) are on the left side of \( L \). The same holds for the right side. If \( F(S) = ∅ \) the property is undefined.

Two segments \( S_l \) and \( S_{l+1} \) can be merged if either \( F(S_{l+1}) = ∅ \) or \( F(S_l) = ∅ \) or \( S_l \) and \( S_{l+1} \) are on the same side of \( L \). The merging operation is transitive and commutative, therefore the order in which segments are merged is not important. After each merging operation we renumberate the segments.
Corollary 24. After the merging operation there are no segments with $F(S_i) = \emptyset$ and there are no two neighboring segments on the same side of $L$.

5.3 Shortest Polyline Problem

We define the procedure computing the shortest polyline. For this purpose we have to extend the definition of a polyline by allowing it to contain vertices which are not points of $F$. For these points we will set $z_i = \emptyset$, i.e. we will say that they lie directly on the polyline. Although we will use such polylines during the execution of the algorithm, we can guarantee that the final polyline returned by our algorithm does not contain such vertices.

The following procedure takes as input a polyline and computes a part of the shortest polyline. Furthermore, it defines a new, shorter, shortest polyline problem and applies itself recursively on this problem.

For a given polyline $C = ((f_1, z_1), \ldots, (f_k, z_k))$ and set of points $F$, the procedure works in the following manner:

**Step 1** Generate segments around line $L = (f_1, f_k)$

**Step 2** Merge segments to $S_1, \ldots, S_m$.

**Step 3** Define a Shortest Circumference Problem for segment $S_1 = ((f_1, z_1), \ldots, (f_n, z_n))$. Let $p_c$ be the crossing point of $L$ and $(f_n, f_{n+1})$. Then we set $P_1 = F(S_1) \cup \{f_1, p_c\}$ and $P_2 = F \setminus P_1$.

**Step 4** Let $p_1, \ldots, p_k$ be the solution to the Shortest Circumference Problem, numbered in such a way that $p_1 = f_1$ and $p_k = p_c$ (upcoming Lemma 25 shows that it is possible to shift the numeration in such a way).

Find the point $p_l$ which has the largest distance to $L$.

**Step 5** If the segment lies on the left side of $L$, define $z'_l = left$ if $p_l \in P_2$ and $z'_l = right$ if $p_l \in P_1$. If the segment lies on the right side, set $z'_l$ the other way around.

**Step 6** Solve recursively the Shortest Polyline Problem for the polyline 

$$(p_0, \emptyset), (p_1, z'_1), \ldots, (p_{l-1}, z'_{l-1}), (p_l, \emptyset) \cup S_2 \cup \cdots \cup S_m,$$

with resulting polyline $C = ((f_1, z_1), \ldots, (f_w, z_w))$. Return the polyline 

$$(p_1, \emptyset), (p_2, z'_2), \ldots, (p_l, z'_l) \cup ((f_2, z_2), \ldots, (f_w, z_w))$$

as the solution.

**Correctness.** The following two lemmas show that the procedure presented is feasible and leads to a proper solution.

**Lemma 25.** Assume the naming from Step 3 of the described algorithm. The polyline consisting the solution to the Shortest Circumference Problem contains the points $f_1$, and $p_c$ as adjacent vertices.

Proof. Rotate the coordinate system in such a way, that the line $L$ is parallel to the $X$-axis. Then the points $f_1$ and the point $p_c$ are both the lowest points of the set $P_1$. It is therefore clear that both of them must be included in the shortest circumference and that they are constitute adjacent vertices of the polyline forming the shortest circumference. □

**Lemma 26.** Let $C = ((f_1, z_1), \ldots, (f_k, z_k))$. Let $S_1$ be the first segment. Let $p_b$ be the point in $F(S)$ with largest distance to $L$. Then the shortest polyline topologically equivalent to $C$ connecting $f_1$ and $f_k$ connects $f_1$ to $p_b$ and then $p_b$ to $f_k$.

Proof. Let $C'$ be a polyline topologically equivalent to $C$. Let $S'_1$ be its first segment. By our definition it must hold $F(S_1) = F(S'_1)$. Therefore, the shortest polyline must enclose $F(S_1)$. Since the segments have been merged appropriately $F(S_2) \neq \emptyset$ and $S_2$ is on the other side of $L$ than $S_1$. Thus, the polyline must cross the line $L$ in order to enclose all points from $F(S_2)$.

This implies that the point $p_b$ must be enclosed by the polyline enclosing $S_1$. As the point $p_b$ is an extreme of the polyline part enclosing $S_1$, it must be a vertex of the polyline. □

**Lemma 27.** The shortest polyline topologically equivalent to $C$ can be computed in time $|F|^2 \cdot n$, where $n$ is the number of points in the shortest polyline.

Proof. In each execution of the algorithm at least one point of the polyline is computed before the recursive invocation. As the shortest polyline has $n$ points, the algorithm can be recursively invoked at most $n$ times.

Each execution of the algorithm requires the definition of segments, their merging and solving the Shortest Circumference Problem. Each of these problems can be solved in $|F|^2$ time. □

6. ADDITIONAL AUTHORS

7. REFERENCES


APPENDIX

PROOF OF LEMMA 2. Assume that a relay station $v_i$ with $i \leq |d/R|$ is on the other side of the line $L_{B,i}$ than the explorer. Therefore $|d/R|$ relay stations are used to support a chain of length at least $d$. This implies that the distance between at least two stations within the chain is more than $R$, leading to a contradiction.

PROOF OF LEMMA 4. First observe that for any $v_i$, it holds

$$b - p_i(0)_{\alpha} \leq b - p_i(i)_{\alpha}.$$  \hspace{1cm} (4)

It is clear that station $v_i$ can be in distance at most $i \cdot R$ from the explorer, otherwise the chain between $v_i$ and $v_0$ got disconnected. Formally $|p_i(i) - p_i(0)| \leq i \cdot R$. Therefore, $|p_i(i) - b| \geq d - i \cdot R$. For all station with indices not greater than $d/2R$ it holds $|p_i(i) - b| \geq d - i \cdot R \geq d/2$. Together with Eq. (4) this yields the claim.

PROOF OF LEMMA 16. In order to simplify notation let us set $r := r_i(i + 1)$, $u := u_i(i)$, $\alpha := \alpha_{i-1}(i)$. From the Law of Cosines we have (see Fig. 9)

$$r^2 = (r + u)^2 + R^2 - 2(r + u) \cdot R \cdot \cos \alpha .$$

Looking at the above as on a polynomial of $u$ yields the following positive root

$$u = R \cdot \cos \alpha + \sqrt{r^2 - R^2 \sin^2 \alpha} - r \geq R \cdot (\cos \alpha - \sin \alpha) .$$

The inequality comes from the fact that $\sqrt{r^2 - c^2 \sin^2 \alpha - r}$ is a non-decreasing function of $r$. Since $r \geq R \cdot \sin \alpha$ for the triangle to exist, we can plug $r = R \cdot \sin \alpha$ and obtain the requested lower bound.

PROOF OF LEMMA 17. Let us consider the triangle between $b$, $b_i(i)$ and $p_i(i)$ as shown on Fig. 10. Stating $\cos \alpha$ as a function of $x = |b_i(i+1) - p_i(i)|$ one obtains

$$\cos \alpha = \frac{x^2 + r_{i-1}(i)^2 - \epsilon_i(i+1)^2}{2x \cdot r_{i-1}(i)} .$$  \hspace{1cm} (5)

This function obtains its minimum for

$$x = \sqrt{r_{i-1}(i)^2 - \epsilon_i(i+1)^2} .$$
Figure 10: Imprecise localization of the navpoint

Plugging in this $x$ into Eq. (5) yields the lower bound. Using the same triangle one obtains the upper bound on

$$\sin \alpha_{t-1}(i) = \frac{\epsilon_i(i + 1)}{x} \leq \frac{\epsilon_i(i + 1)}{r_{t-1}(i) - \epsilon_i(i + 1)} .$$

\[ \square \]

**Proof of Lemma 18.** Observe that the distance $r_{t-1}(i)$ is equal to a sum of $\hat{u}_{t-1}$

$$r_{t-1}(i) = \sum_{j=i}^{n_{t-1}} (r_{t-1}(j) - r_{t-1}(j + 1)) = \sum_{j=i}^{n_{t-1}} \hat{u}_{t-1}(j) .$$

Observe that $n_{t-1} \geq n_t - 1$, since the number of relay stations in the chain can change by at most 1 during one time step. Then we have

$$r_{t-1}(i) \geq \sum_{j=i}^{n_{t-1}} (u_{t-1}(j) - 1) \geq (n_{t-1} - i) \cdot \min_{j \in \{i, n_{t-1} - 1\}} u_{t-1}(j) - 1 \geq (n_t - i - 1) \cdot (u - 1) .$$

Note that we have lower bounded $u_{t-1}(n_{t-1})$ by 0, since the last relay station can be very near to the base station. Since $u \geq 3$ it follows

$$r_{t-1}(i) \geq (n_t - i - 1)(u - 1) \geq \frac{1}{3} \cdot \gamma(i + 1) \cdot u .$$

\[ \square \]

**Proof of Lemma 21.** Figure 8 shows an example of the discussed situation. Let $a := |D - A|_2$, $b := |C - A|_2$, and $z := |C - D|_2$. Obviously $|C - B|_2 - |D - B|_2 = z$. Let $\alpha$ be the angle at $A$ between $\langle A, D \rangle$ and $\langle A, C \rangle$. Note that $z \geq 0$. By the Law of Cosines it holds

$$z^2 = a^2 + b^2 - 2 \cdot a \cdot b \cdot \cos \alpha \geq a^2 + b^2 - 2 \cdot a \cdot b = (a - b)^2 .$$

Since $z \geq 0$ from $(a - b)^2 \leq z^2$ follows $a - b \leq z$. As $a - b \leq z$ is equivalent to Eq. (3) the lemma holds. \[ \square \]