Time-based ray tracing forwarding in dense nanonetworks

Eugen Dedu and Masoud Asghari

Abstract Wireless nanonetworks, consisting of nodes of nanometric size, are an emerging technology with various applications, such as in medicine, metamaterial and agriculture. As in any multi-hop network, routing is an important primitive. Due to their extremely small sizes, nanonodes' energy budget is rather limited, hence any method to reduce the energy used is of paramount importance. The energy is influenced, among others, by the number of packets sent and received, and forwarding nodes. This paper proposes ray tracing forwarding, where only nodes having received duplicate packets in the same time slot can forward them. Simulations in a dense nanonetwork show that the forwarders form a quasistraight line and can deviate at borders. In long nanonetwork environments such as blood vessels and branches, the ray tracing forwarding outperforms the coordinate-free and coordinate-based routing methods found in the literature.

1 Introduction

Nanotechnology allows the design of integrated nanodevices at the nano scales, opening a wide range of novel applications that could not be imagined in macro scales. Due to the extremely limited size, resources and energy of nanodevices, one cannot expect very much from an individual nanodevice. Thus, for a viable solution, the nanonodes should collaborate in a distributed nanonetwork structure to perform

Eugen Dedu
FEMTO-ST Institute, Univ. Franche-Comté, CNRS
Numérica, cours Leprince-Ringuet, 25200 Montbéliard, France
e-mail: eugen.dedu@univ-fcomte.fr

Masoud Asghari
Department of Computer Engineering, Faculty of Engineering, University of Maragheh, P.O. Box 55136-553, Maragheh, Iran
e-mail: mas.asghari@maragheh.ac.ir

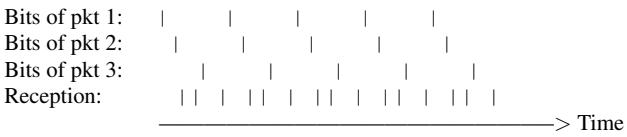


Fig. 1 In TS-OOK, bits are sent with T_s interval, and are received (with a bit of delay) with the same T_s interval.

complex tasks in a considerable range. Electromagnetic nanonetworks consist of nodes of nanometric size (i.e. their components are less than 1 µm, to make nodes with sides less than around 10 µm) communicating in the terahertz band. They usually contain a nanoprocessor, a nanomemory, a nanosensor and a nanodevice to communicate with the other nanonodes. Nanonetworks can be dense, making them a very challenging environment for traditional routing protocols. Moreover, due to their tiny size, classical network techniques are unsuitable. The modulation proposed, Time Spread On-Off Keying (TS-OOK) [7], does not use signal carriers, but pulses. A bit 1 is transmitted by generating a power pulse, and a bit 0 is "transmitted" as silence. The time length of a pulse is $T_p = 100$ fs [7]. Due to hardware constraints, a nanonode cannot send bits one right after the other; the time between two consecutive bits is $T_s = \beta \times T_p$, and $\beta = 1000$ is given as an example of time spreading ratio [7].

Given that bits are sent with T_s interval between them, a receiver can match the bits from one packet by reading bits at T_s interval too, i.e. at time x, $x + T_s$, $x + 2T_s$, etc. Bits received at different times belong to another packet. An example of sending bits and matching them is given in Fig. 1. For simplicity of understanding, we assume that any T_s interval is divided into β time slots, and all bits from one packet are received at the same slot.

Nanonetworks can be connected to the macro world by gateway nodes, enabling the Internet of Nano-Things (IoNT). The communication of nanodevices in the sense of intrabody nanonetworks [3] opens the door to innovative medical applications inside the human body such as molecular-level detection and reporting of pathogens including viruses and bacteria, high-precision drug delivery, targeted monitoring, and neurosurgery. The combination of nanonetworks and metamaterials (artificial structures with unnatural properties) in the form of Software-Defined Metamaterials (SDMs) allows performing geometrically-altering actions on the metamaterial and tuning of its electromagnetic behavior by sending commands to nanodevices. Other applications include wireless robotic materials, military and agriculture.

The communication range of the nanodevices is usually less than 1 mm, because the path loss of the THz band in environments with high water concentration such as blood is very high, and nanodevices do not have enough power density (due to their limited sizes) to overcome the path loss in longer distances. Therefore, a multi-hop routing is needed to send data packets over a considerable distance in the nanonetworks. The traditional routing approaches proposed in the literature are not suitable to nanonetworks (e.g. backoff flooding is not destination-oriented, but does flood-

ing). Due to nanodevices' limited resources and energy, using specialized routing techniques based on the nanonetwork operating conditions (e.g., perimeter dimensions and shape, nodes placement and mobility, nodes energy source and limitations, desired applications) to reduce packet dissemination in unnecessary directions and reduce the number of forwarders can improve nanonetwork performance and lifetime.

In this paper, we focus on long nanonetwork perimeters such as pipes and blood vessels, where previously proposed nanonetwork routing techniques cannot work properly. To this end, we propose ray tracing forwarding for nanonetworks by using TS-OOK characteristics to forward packets in quasistraight lines. The ray tracing forwarding creates a narrow packet propagation path from a source to the gateway across a long nanonetwork perimeter without flooding the packet to all the nodes in the network. This reduces the dissemination of the packets to unnecessary nodes, leading to fewer forwarders and less network energy and resource usage.

In the ray tracing forwarding, only nanodevices having received duplicate packets in the same time slot will forward them. With the proposed algorithm, the forwarders form a quasistraight line and can deviate at the borders of the long perimeter, moving the packet across the perimeter from the source to the gateway at the end of the perimeter. The contributions of this paper are as follows:

- For the first time in literature, we utilize the time slots of TS-OOK and signal propagation duration as means of constructing a quasistraight path from the source across the nanonetwork.
- We propose the ray tracing forwarding algorithm to utilize these quasistraight paths for routing in long perimeters and reduce the number of forwarders.
- We implemented and evaluated the ray tracing forwarding in a simulator, comparing its results with the related forwarding methods.

2 Related work

The routing protocols either deliver the data packets to a destination node through the nodes located between the source and destination (unicast or merely zone-cast routing), or flood the whole network so that every node receives a certain message (flooding). Due to the shape of the perimeter, mobility of nanonodes and very high number of homogeneous nanonodes in the dense nanonetworks, traditional addressing and routing protocols cannot be applied in nanonetworks, making the flooding-based forwarding the only viable option in most cases. In *pure flooding*, every node in the network forwards every unique packet that it receives. This flooding results in resource wastage and broadcast storms in dense environments, leading to poor network performance. In *probabilistic flooding*, nodes forward packet with a certain probability [11]. The probability needs to be tuned to prevent broadcast storms and guarantee message delivery. *Backoff flooding* is a highly efficient flooding scheme using a counter-based mechanism to count copies received, and adds a random waiting time (backoff) to packet forwarding [2]. The backoff window

depends on the neighbor density. Any node forwards the packet to all nanonodes within its wireless transmission range only if it did not receive r (redundancy factor) copies of that packet during the backoff period. As a general fact, in flooding (including probabilistic and backoff), a packet is broadcasted in the whole network instead of moving toward the destination, leading to redundant packet reception and wastage of nodes resources (e.g. bandwidth and energy). Therefore, any attempt to reduce packet dissemination in unnecessary directions and reduce the number of forwarders can improve network utilization.

Geoforwarding protocols rely on geographic information about node positions to take the forwarding decision [1]. Node position can be obtained either directly through a GPS receiver in each node, or by triangulation techniques using relative positions of the nodes obtained by anchors and beacons [4]. Due to the small size, limited resources, low energy and very high density of the nanonodes, neither GPS or triangulation-based nanonode positioning, nor any protocol that relies on detailed neighborhood knowledge or routing table can be employed in the nanonetworks [2]. In *RADAR* routing, the nanonetwork environment is a circular area and a central node emits a directional signal at an angle [10]. Nanonodes receiving the signal inside the angle of radiation are in the ON state, and all the other nodes are in the OFF state. RADAR requires a convex region and a symmetric perimeter with a special node in the center. Similar to RADAR, *DEROUS* also requires a beacon node set at the center of a 2D circular area sending packets to classify nanonodes [8]. In *SLR*, the network is in a 3D cubic space and a few anchor nodes are placed at corners. During the setup phase, anchors transmit their packets in sequence, allowing nanonodes to set their coordinates as hop counts from the anchors [13].

Straight-line routing protocol is proposed for wireless sensor networks (WSNs) in [5, 9]. It is a form of random-walk routing protocol which constructs a straight path between the source and destination using two-hop information without using geographic information. The method constructs the path in a hop-by-hop manner, where after each hop a node that lies on the extended line of the path is selected as the next hop. It assumes that after receiving a packet, each node is able to determine the distance from its transmitter according to the signal strength [5, 9]. The distances between nodes are much shorter in nanonetworks than in WSNs. Unlike WSNs, nanonodes usually use TS-OOK modulation in the THz band where bits 1 are transmitted by ultra-short pulses of duration T_p, and bits 0 are implied by the absence of the pulse (as described in the Introduction). The receiver on the nanonodes is very basic the can only detect THz band pulses. Due to the short distances, TS-OOK modulation and simplification of the radio receivers, nanonodes cannot determine accurate distances based on the received pulse strength, rendering the protocol inappropriate for nanonetworks. However, TS-OOK employs temporal multiplexing by using very short pulses with an accurate pulse duration to detect consecutive bits of each packet. In this paper, we employ these TS-OOK characteristics (pulse propagation and arrival times) in a ray tracing fashion to forward packets in quasistraight lines and reduce the number of forwards.

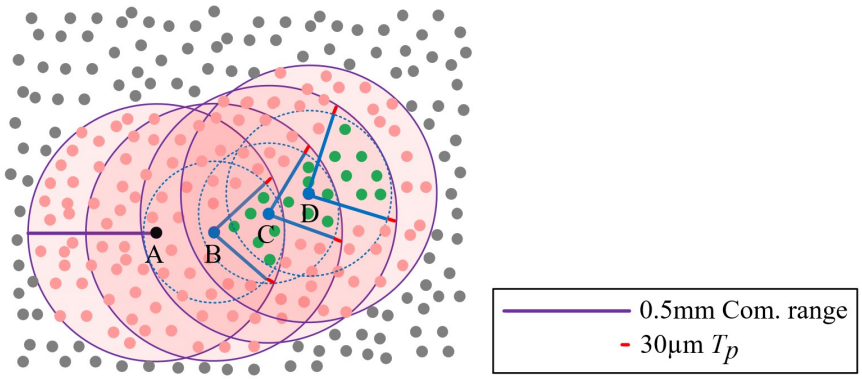


Fig. 2 Sketch of ray tracing forwarding: potential forwarders are in green (they are "collinear" with the previous two transmitters), and effective forwarders (*B*, *C* and *D*) are in blue.

3 Ray tracing forwarding

The network model is a long strip such as a pipe or blood vessel. The network is dense, i.e. any node has at least 50–100 neighbors.

The ray tracing method *does not make any assumption* by itself. As presented below, it needs nodes to be able to listen at T_s interval, and to send bits at a given time (i.e. there is no random/nondeterministic backoff added at the device level), but these characteristics are already fulfilled in nanonetworks.

The ray tracing aims to choose forwarders in a straight line. For that, it uses the *timing of packet receptions*: when two 1 bit pulses (note: 0 bits do not emit pulses and cannot be used) from two copies of the same packet sent by two transmitters arrive in the same time slot (explained in the Introduction), the receiver considers itself to be collinear with the two transmitters. Which 1 bits are taken from the packets does not matter, since consecutive bits of one packet are spaced by $1\,T_s$, hence are in the same slot.

The ray tracing forwarding is shown in Fig. 2 and is explained in the following.

All the nodes execute the same procedure, in a loop. Thus, we describe the procedure recursively: the base case (initialization), and the recursive step (general case).

Initialization step: A source *A*, called the *sender*, sends a packet. All the nodes inside the communication range receive it, and one of them, *B*, called the *steering node* because it produces the forwarding direction, forwards it.

Recursive step: We are at the time where node *A* has already sent a message, and node *B* is currently sending a message. This message will be received in all its communication range. All the nodes in the network are in one of the following cases:

- It is the first packet it receives: node just memorizes the time slot.
- It is the second packet it receives: if the current packet is *not* in the same time slot as the first one, then it discards the packet and forgets about this communication;

```
Input:
  timefirst[] = null

if timefirst[id] == null then     // packet id is seen for the first time
    timefirst[id] = (crttime % Ts) / Tp     // % is modulo operation
    second[id] = false
elseif second[id] == false then     // 2nd packet id received
    second[id] = true
    if (crttime – timefirst[id])%Ts < Tp or (crttime – timefirst[id])%Ts > Ts – Tp then
        // same time frame, so collinearity
        backoff = pktsize ×Ts× random int number in [0,1000)
        schedule event e[id] to forward p[id] at crttime+backoff
        scheduled[id] = true
    end
else // 3rd or subsequent packet
    if scheduled[id] == true then
        unschedule event e[id]
        scheduled[id] = false
    end
end
```

Fig. 3 Ray tracing forwarding algorithm.

otherwise (i.e. the two slots are the same), it uses a backoff: if it receives any copy before the backoff, discard it, elsewhere send it at the same time slot.
The window from which the backoff is chosen is large, hence only one of them will forward, whereas all the others will see the copy of this packet and as such will discard their packet.
Geometrically, all the nodes in the intersection of the discs (communication range) of the first and the second packet receive both packets. Some of them receive the two packets in the same time slot, depicted as green nodes in Fig. 2, and consider themselves as collinear with A and B. The other nodes receive them in different slots, and they discard the packet and forget about the communication.

- It is the third or ensuing packet it receives: discard it (because its forwarding time has already passed, and the propagation is already in front of it).

Once node B sent the packet, the procedure is repeated (recursive step) with node B becoming sender A, and node C becoming steering node B.

In a multi-flow case, we suppose each packet generated by a source has a different id (made for example from the concatenation of three strings: source id, flow/port id, packet sequence number). The algorithm executed by each node upon reception of a packet p[id] is shown in Fig. 3.

Finally, after the multi-hop forwarding, the packet reaches the destination zone. Here, a special procedure is executed instead of the ray tracing. We consider that the destination node is a gateway with large resources, and especially with a high receiving sensitivity, which can capture the packet sent by the nodes in its vicinity (i.e., the strip width). Upon reception of a packet, it sends a high power packet,

received by all the nodes in its vicinity, which informs nodes to stop further propagation. The ray tracing forwarding ends as the gateway has received the packet and the propagation stops.

4 Evaluation

4.1 Nanonetwork simulator used

Given the high number of nodes involved, experiments are not possible, hence we use simulations. Among the nanonetwork simulators, only BitSimulator [6] allows to simulate more than around one thousand nodes. It is much more scalable compared to the other nanonetwork simulators [12]. BitSimulator targets routing and transport protocols, and pay attention to some low-level bit transmission peculiarities, such as transmission time and bit-dependent collision at the receiver. It uses TS-OOK modulation. It also provides a visualization program, very useful to *see* what happens in the network. It is free software and has been used to validate the results of several articles[1].

We implemented the ray tracing forwarding in BitSimulator. The other routing protocols were already included in the simulator.

We provide a Web page[2] allowing to reproduce all the results of this article.

4.2 Base scenario

The simulation parameters are shown in Table 1. The network is a rectangular strip (6 mm $\times$ 3 mm) and has 5000 nanonodes placed randomly using a uniform distribution, plus 2 nanonodes placed manually, as shown below. Node antennas are omnidirectional, and the communication range is CR=0.5 mm, i.e. the network is $3/0.5 = 6$ CR high and $6/0.5 = 12$ CR wide, enough to test the linear forwarding. Neighbour density (also called node or network density) is $5002 * \frac{\pi * 0.5^2}{6*3} = 218$ neighbors. The TS-OOK low-level parameters (described in the Introduction) are the ones proposed in the literature, i.e. $T_p = 100$ fs and $\beta = 1000$.

The sender node (of id 0), on the left of the network, and the steering node (of id 1), a bit on its right, are manually placed to start the algorithm. The sender node generates one packet (the packet has 80 bytes, a random number, and a random payload). The steering node retransmits it. Afterward, all the nodes execute the ray tracing algorithm.

[1] http://eugen.dedu.free.fr/bitsimulator

[2] http://eugen.dedu.free.fr/bitsimulator/aina23

Table 1 Simulation parameters.

Parameter	Value
Number of nodes	5002
Network size	6 mm × 3 mm
Communication range	0.5 mm
⇒ Neighbour density	218

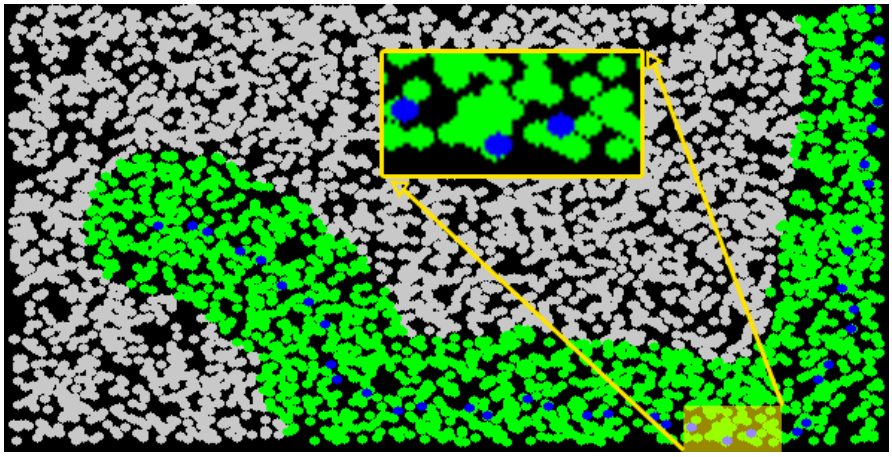


Fig. 4 VisualTracer capture, showing forwarders (in blue), receivers (in green) and the other nodes (in grey) during the ray tracing forwarding in the 5002 node scenario; the small region shows the auto-deviation capability of the ray tracing forwarding.

4.3 Ray tracing propagation features

Fig. 4 presents the forwarders, the receivers and the remaining nodes for the whole simulation in one representative case. It can be seen that the forwarding nodes are quasilinear, and that the propagation auto-deviates when reaching borders. Both features are explained in the following.

4.3.1 Quasilinear forwarding

The goal of ray tracing is to reduce the number of forwarders when transmitting a packet from a source to a destination. Ideally, the propagation would go straight from the source to destination. However, in reality the line is *quasi*straight. The nonlinearity is due to time resolution error and to the next node placement gaps in space.

The time resolution error (T_p) occurs because the locus of points having received two packets in the same time slot (of length T_p) resembles to a triangle, as shown in Fig. 2. Any of these points can be the forwarder.

The node gaps are due to the impossibility of having nodes *everywhere* in the network. Obviously, there are spaces, and no forwarder can be there.

To conclude, the propagation line is not straight, but *quasi*straight. An example is given in Fig. 4, the left part, from the left source until reaching the bottom border, where the propagation deviates.

4.3.2 Auto-deviation at borders

Our ray tracing technique has auto-deviation capabilities when reaching network borders or walls, i.e. it deviates the forwarding path pushing further the packet propagation. This section explains the reason of this desired and unexpected feature.

Let us look a zoom of some part at the bottom border of Fig. 4, shown at its top. At this time, the sender is the left blue node and the steering node is the middle one.

The potential forwarders, i.e. having received the packet in the same time slot, are located at the right of the middle node, in a triangle. The bottom 3/4 of the triangle is black, i.e. no node is there, because it is outside the network. Hence, the potential forwarders are found only on the top quarter of the triangle, and one of them will become a forwarder (the right blue node in our case). This turns counterclockwise the line between the sender and steering nodes, making it deviate from the bottom border.

To conclude, the deviation happens because:

- The potential forwarders are inside the network, not outside.
- There *are* nodes inside the network that are potential forwarders.

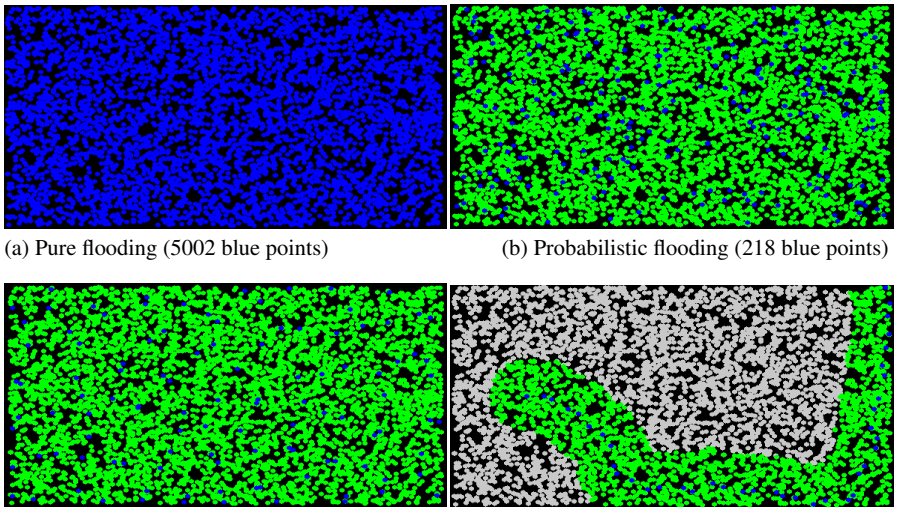
It should be noted that the deviation is not always successful, as can be seen at the top-right of Fig. 4, where the propagation stops. Some specific mechanism, outside of the scope of this article, is needed for that.

4.4 Comparison with related forwarding methods

The ray tracing method does not use any coordinate. We therefore compare it with coordinate-free protocols, and afterward with coordinate-based protocols.

4.4.1 Comparison with coordinate-free methods

The only coordinate-free methods we found appropriate to nanonetworks are flooding protocols: pure flooding, probabilistic flooding (with probability of 4 %, the minimum value which still allows propagation in the whole network), and the highly optimized auto-adaptive backoff flooding, all of them described in Related work (Sect. 2). We compare the ray tracing method with them.



(a) Pure flooding (5002 blue points) (b) Probabilistic flooding (218 blue points)

(c) Backoff flooding (110 blue points) (d) Ray tracing (40 blue points)

Fig. 5 VisualTracer capture of the four compared methods, showing forwarders (blue nodes) and receivers (green nodes) for the 5002 node scenario.

Method	Forwarding nodes	Packet receptions
Pure flooding	5002 (100 %)	980 146
Probabilistic flooding	218 (4.3 %)	42 804
Backoff flooding	110 (2.2 %)	20 326
Ray tracing	40 (0.8 %)	6 825

Method	Forwarding nodes	Packet receptions
SLR	1654 (33.0 %)	335 423
Counter-based SLR	33 (0.66 %)	6 987
Ray tracing	29 (0.58 %)	5 259

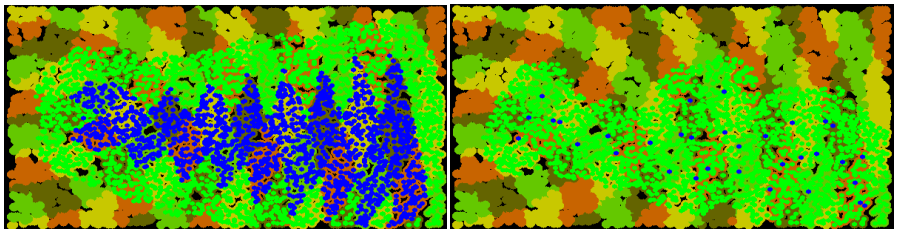
Table 2 Comparison of the ray tracing with related flooding (left) and destination-oriented routing (right) methods.

Fig. 5 shows the propagation of the compared protocols. It shows the placement of senders in the whole network for flooding, and the directed routing for ray tracing. It also shows that the ray tracing has the fewest number of senders and receptions.

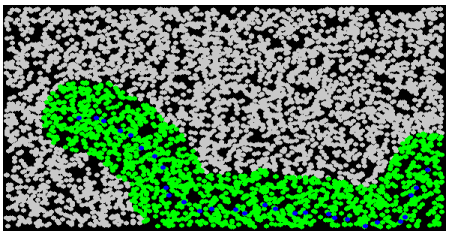
Table 2 shows the number of forwarding nodes and of *packet receptions* (a node receiving two packets is counted as two packet receptions). It can be seen that ray tracing reduces considerably the number of forwarders (5002 to 40) and receptions. It outperforms the highly optimized backoff flooding, given that it has a quasistraight forwarding, unlike backoff flooding which floods the network.

4.4.2 Comparison with coordinate-based methods

SLR is a destination-oriented routing protocol proposed for nanonetworks [13], which uses a setup phase where nodes assign coordinates. We note that it can be improved by combining it with backoff flooding (called “counter-based SLR” in the



(a) SLR (1654 blue points) (b) Counter-based SLR (33 blue points)



(c) Ray tracing (29 blue points)

Fig. 6 VisualTracer capture of the three compared destination-oriented methods, showing forwarders (blue nodes) and receivers (green nodes) for the 5002 node scenario.

following). Both SLR and backoff flooding are described in Related work (Sect. 2). We compare ray tracing method with them.

In the scenario, one of the nodes found on the bottom right of the network is selected as the destination. Also, for a fairer comparison, for the two SLR-based methods, the packet exchanges during the SLR setup phase is not counted, given that it is done only once, at the network deployment.

The propagation is shown in Fig. 6 (showing the SLR zones too), and the results are given in Table 2. Ray tracing (29 forwarders) outperforms both SLR (1654 forwarders) and counter-based SLR (33 forwarders). The reason is that ray tracing is more linear than counter-based SLR. Thus, even if the latter is quite optimized, ray tracing reduces even more the number of forwarders and receptions.

To conclude, ray tracing outperforms both related coordinate-free and coordinate-based methods. Besides, the latter methods need an additional step at network deployment (to create the coordinate system), which might prevent node mobility and might create problems in nonconvex or very long environments such as blood vessels, giving the only coordinate-free methods comparable to the ray tracing. Instead, ray tracing does not need any coordinate system, but only a direction.

5 Conclusion and perspectives

This paper presented a ray tracing method to forward packets in a quasistraight line in a multi-hop dense nanonetwork. It allows to reduce the number of forwarders and the number of receptions through a quasistraight line, and has deviating capabilities on network borders. Simulations show that it outperforms, in terms of number of forwarders and receptions, all the related methods, either coordinate-free or coordinate-based, without relying on a coordinate system. Thus, our method proves that the packet reception times can indeed be used to do a linear routing.

This paper presented preliminary results of the ray tracing forwarding. We think that this method has a big potential, and several improvements can be done on it. Future work includes analyzing the algorithm in less dense scenarios, where sometimes there is no node in the forwarding direction, in even denser networks, where forwarding can take several directions if the backoff window is not tweaked, and in various networks (with walls or curved). Also, how to choose the steering node automatically, based on the desired direction, and how to improve the deviation.

Acknowledgements This work has been funded by Pays de Montbéliard Agglomération (France).

References

1. Abu-Ghazaleh, N., Kang, K.D., Liu, K.: Towards resilient geographic routing in wsns. In: Int. Workshop on Quality of service&security in wireless and mobile networks, pp. 71–78 (2005)
2. Arrabal, T., Dhoutaut, D., Dedu, E.: Efficient density estimation algorithm for ultra dense wireless networks. In: Int. Conf. on Computer Comm. and Networks (ICCCN), pp. 1–9 (2018)
3. Asghari, M.: Intrabody hybrid perpetual nanonetworks based on simultaneous wired and wireless nanocommunications. Nano Communication Networks **32**, 100,406 (2022)
4. Bulusu, N., Heidemann, J., Estrin, D.: Gps-less low-cost outdoor localization for very small devices. IEEE Personal Communications **7**(5), 28–34 (2000)
5. Chou, C.F., Su, J.J., Chen, C.Y.: Straight line routing for wireless sensor networks. In: 10th IEEE Symposium on Computers and Communications (ISCC'05), pp. 110–115. IEEE (2005)
6. Dhoutaut, D., Arrabal, T., Dedu, E.: BitSimulator, an electromagnetic nanonetworks simulator. In: 5th ACM/IEEE International Conference on Nanoscale Computing and Communication (NanoCom), pp. 1–6. ACM/IEEE, Reykjavik, Iceland (2018)
7. Jornet, J.M., Akyildiz, I.F.: Femtosecond-long pulse-based modulation for terahertz band communication in nanonetworks. IEEE Trans. on Communications **62**(5), 1742–1753 (2014)
8. Liaskos, C., Tsioliaridou, A., Ioannidis, S., Kantartzis, N., Pitsillides, A.: A deployable routing system for nanonetworks. In: IEEE Int. Conf. on Communications (ICC), pp. 1–6 (2016)
9. Liu, H.H., Su, J.J., Chou, C.F.: On energy-efficient straight-line routing protocol for wireless sensor networks. IEEE Systems Journal **11**(4), 2374–2382 (2017)
10. Neupane, S.R.: Routing in resource constrained sensor nanonetworks. Master's thesis, Tampere University of Technology, Finland (2014)
11. Reina, D.G., Toral, S., Johnson, P., Barrero, F.: A survey on probabilistic broadcast schemes for wireless ad hoc networks. Ad Hoc Networks **25**, 263–292 (2015)
12. Sahin, E., Dagdeviren, O., Akkas, M.A.: An evaluation of internet of nano-things simulators. In: 6th Int. Conf. on Computer Science and Engineering, pp. 670–675. Ankara, Turkey (2021)
13. Tsioliaridou, A., Liaskos, C., Dedu, E., Ioannidis, S.: Packet routing in 3d nanonetworks: A lightweight, linear-path scheme. Nano communication networks **12**, 63–71 (2017)