Trajectory-Based Forwarding Mechanisms for Ad-Hoc Sensor Networks

Murat Yuksel, Ritesh Pradhan, Shivkumar Kalyanaraman
Rensselaer Polytechnic Institute, Troy, NY
yuksem@ecse.rpi.edu, pradhr@ecse.rpi.edu, shivkuma@ecse.rpi.edu

Abstract— **Routing in ad-hoc sensor networks is a complicated task because of many reasons. The nodes are low powered and they cannot maintain routing tables large enough for well-known routing protocols. Because of that, greedy forwarding at intermediate nodes is desirable in ad-hoc networks. Also, for traffic engineering, multipath capabilities are important. So, it is desirable to define routes at the source like in Source Based Routing (SBR) while performing greedy forwarding at intermediate nodes.**

We investigate Trajectory-Based Routing (TBR) which was proposed as a middle-ground between SBR and greedy forwarding techniques. In TBR, source encodes trajectory to be traversed and embeds it into each packet. Upon the arrival of each packet, intermediate nodes decode the trajectory and employ greedy forwarding techniques such that the packet follows its trajectory as much as possible.

In this paper, we provide techniques to efficiently forward packets along a trajectory defined as a parametric curve. We use the well-known Bezier parametric curve for encoding trajectories into packets at source. Based on this trajectory encoding, we develop and evaluate various greedy forwarding algorithms.

I. Introduction

Ad-hoc sensor networks have their own characteristics which lead to significant amount of research in the area. Particularly, routing in ad-hoc sensor networks is a complicated task because of many reasons. For example, nodes are low powered and they cannot maintain routing tables large enough for well-known link-state or distance-vector routing protocols. This is known as *stateless routing* [1], since nodes do not maintain routing tables representing network state. Moreover, nodes are mobile which makes it harder to converge for typical proactive routing protocols.

So, because of its stateless nature, greedy forwarding (e.g. GPSR [1] and Cartesian Routing (CR) [2]) of packets at intermediate nodes is desirable in ad-hoc networks. Also, for traffic engineering, multipath capabilities (e.g. Source Based Routing (SBR) [3]) are desirable. However, it is not possible to employ well-known multipath routing techniques (e.g. MPLS [4], or others [5]) in mobile networks. Niculescu and Nath [6] proposed Trajectory-Based Routing (TBR) as a middle-ground between SBR and greedy forwarding techniques. In TBR, source encodes trajectory to traverse and embeds it into each packet. Upon the arrival of each packet, intermediate nodes employ greedy forwarding techniques such that the packet follows its trajectory as much as possible. This way, routing becomes source-based while there is no need for routing tables for forwarding at intermediate nodes.

Furthermore as another motivation for TBR, there is a new trend toward *application-driven networking* [7] where applications can communicate with network and customize network behavior based on their own requirements. For example, consider an image processing application which collects pictures taken at different nodes in the network and merges them into a 3D picture of a scene. Consider the example network in Figure 1. Assume that the application is running at nodes A and B, and wants to create a big picture for the west of mountains. Observe that traditional shortest-path routing is not suitable for this type of application since the shortest path from A to B traverses nodes that are far from the west of mountains. A more suitable routing for this application is to route such that traffic of this application traverses nodes that are close to the trajectory defined as *the west of mountains*. This trajectory is also drawn as a parametric curve in the Figure 1. So, TBR is promising for such applications, examples of which can be extended.

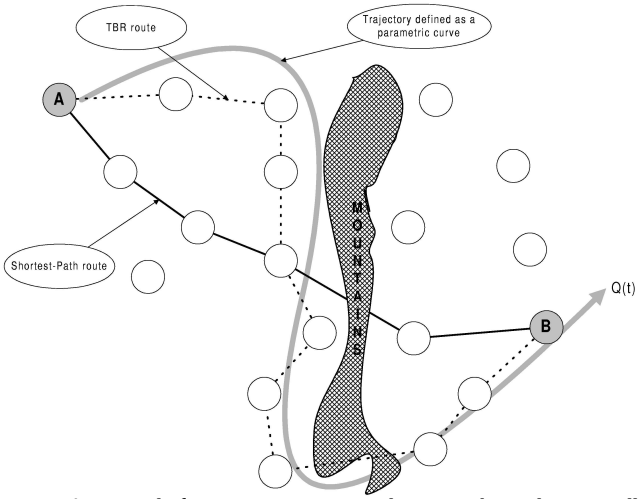


Fig. 1. An example for using TBR in an application: The application collects photos of the "west of mountains", which causes best route to be different than traditional shortest-path routing.

In [6], Niculescu and Nath described basic features of TBR along with a Local Positioning System (LPS). Since it has a greedy forwarding mechanism, TBR needs support for positioning of wireless nodes. For the positioning problem of TBR, various positioning systems such as GPS [8] can be used. However, GPS requires high power availability which violates conditions of low-power ad-hoc sensor networks. As a solution to this, Nath and Niculescu proposed LPS which can be used to enable TBR's implementation at low-power nodes without GPS support. So, in this paper, we assumed that the nodes have a knowledge of their positions with respect to a mutually known coordinate system. This assumption is reasonable as the use of GPS as well as other positioning tools are becoming more popular [9], [10], [11], [12].

In TBR, one important issue to explore is how to efficiently forward packets along a defined parametric curve $Q(t)$. Niculescu and Nath experimented with simple parametric curves such as sine curve, and left the question of how to encode various trajectories into packets as a parametric curve. In this paper, we propose an effective method of encoding trajectories into packets at source. Given this trajectory encoding techniques at source, we present various mechanisms to perform forwarding at intermediate nodes.

The rest of paper is organized as follows: First, in Section II we describe details of Bezier curves and how to use them for trajectory encoding in TBR. Next in Section III, we propose

various greedy algorithms for packet forwarding in TBR with Bezier curves. In Section IV, we briefly describe ways of scaling packet header for longer and more complex trajectories. In Section V, we present ns-2 simulations of the forwarding algorithms and evaluate their performance. Finally, in Section VI we summarize the work.

II. Using Bezier Curves for TBR

In this section, we will discuss the basics of Bezier curves used for TBR. Bezier curves are special types of curves that are used in the area of graphics for representing letters in special purpose fonts. These curves are defined by a number of points - *source*, *destination*, and some *control points*. Depending on the number of control points, they are named accordingly. For instance, a Bezier curve defined by 1 *control* point is called as **quadratic Bezier curve**, while the one which is defined by 2 *control* points is known as **cubic Bezier curve**. More details about basic calculations for Bezier curves can be found in [13].

Given a trajectory defined by a Bezier curve, the nodes can either be on the Bezier curve or could be near the Bezier curve. In order to implement forwarding algorithms, for a node near the Bezier curve, we need to find where this node corresponds on the Bezier curve. This is actually the point on the curve closest to the node.

Finding the Bezier curve point closest to a node is a non-trivial task. In the Figure 2, the node does not lie on the Bezier curve. To calculate the point on the curve which is nearest to the node, we draw a perpendicular on the tangent of the curve. Now, with $Q(t)$ being a third order polynomial and the tangent $Q'(t)$ being a second order polynomial, we get a fifth order polynomial when we have $Q(t)Q'(t) = 0$. One of roots of this equation will be the point on the Bezier curve $Q(t)$ nearest to the node [14]. Roots of a fifth degree polynomial can be computed, but finding roots of the polynomial with order greater than 5 is not possible with current algebraic techniques.

Given the above methodology to find the nearest point a Bezier curve, we now fix a terminology to ease writing rest of the paper. Given a Bezier curve $Q(t)$ and a node N_i as shown in Figure 2, we call the value of parameter t at the curve point closest to N_i as *residual* of N_i and represent it by t_i. The closest curve point itself is called as *residual point of* N_i, and represented by $Q(t_i)$. Finally, we call the distance between the node and $Q(t_i)$ as the *residual distance of* N_i and represent it by d_i.

Given this context, the source node encodes the four points (i.e. source, destination, two control points) that define the Bezier trajectory into packet header. When an intermediate node receives such a packet it decodes the four points and can determine the necessary information (e.g. what is the complete trajectory, how close itself to the trajectory, how close are its neighbors to the trajectory) to implement greedy forwarding.

III. Greedy Forwarding Algorithms for TBR

Given a neighborhood and a trajectory to follow for the packet, a node may follow different forwarding strategies depending on application and user criteria. One can define various objectives for forwarding in TBR:

- *Obey the trajectory:* There might be cases where obeying the trajectory is critical. For example, if the trajectory is passing through just near enemy area in a battlefield, then making sure that packets are obeying the trajectory and are not getting to the enemy area is important. This becomes particularly important when packets include secure information that must not reach to enemy wireless agents.
- *Reach the destination node:* As another criteria, if application generating the packets is sensitive to packet loss, then one might find it more convenient to forward the packet to the destination node if it is in the neighborhood of the forwarding node although it might be disobeying the trajectory significantly.
- *Reach quickly:* If the information being sent is delay sensitive and the similarity of route to trajectory is not of much importance, then it becomes more convenient to forward the packets such that they reach to the destination as quick as possible.

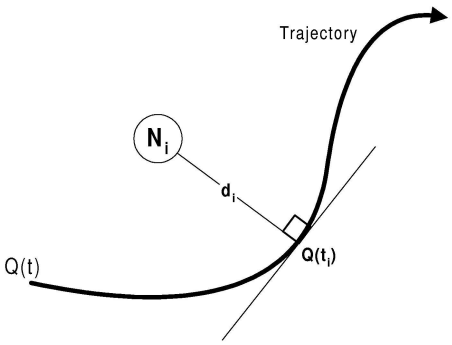


Fig. 2. A node near a trajectory defined by a Bezier curve $Q(t)$.

For usefulness of the forwarding strategy, the forwarding algorithm must make sure that the packet advances along the trajectory curve. In other words, a node should not forward a packet backwards along the trajectory curve. For example, in Figure 3-a, consider node N_0 with residual t_0. Although there are other nodes within the transmission range of N_0, the forwarding algorithm must forward packets to one of the gray nodes whose residuals are larger than t_0. We will call the set of nodes that have residuals larger than t_0 as *neighborhood*[1] *of* N_0. Within the neighborhood, selection of which node to forward packets next depends on various user and application objectives, some of which were itemized above. We now define various techniques that can be used for this selection process:

- *Random:* A simple algorithm is to select the next node randomly from the neighborhood. This algorithm is beneficial when computation power is of critical importance.
- *Closest to Curve (CTC):* Another computationally simple algorithm is to select the node which is closest to the curve among the nodes in neighborhood. This algorithm is pretty straightforward to implement. However, it may result in significant errors in forwarding such as shown in Figure 3-b. Since residual distance d_5 of node N_5 is smaller than residual distances all the other nodes in the neighborhood, N_0 forwards packet to N_5 which causes a significant violation of the trajectory.
- *Least Advancement on Curve (LAC):* One might need to traverse all the nodes that are along the trajectory curve. A simple algorithm for that is to forward to the node whose residual lies right next to the residual of the current node. However, again, this might result in significant errors in forwarding such as in Figure 3-c. Although N_1 is the farthest node from the trajec-

[1] Note that our definition of neighborhood is different from Niculescu and Nath's definition in [6].

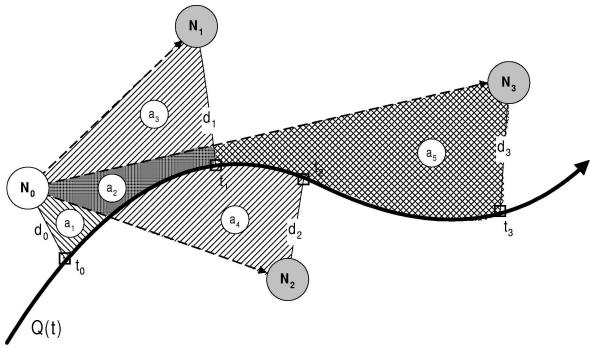


Fig. 4. Big picture of LDC forwarding

tory curve, N_0 forwards packets to N_1 because t_1 is less than residuals of all the other nodes in the neighborhood of N_0.

- *Hybrid of CTC and LAC (CTC-LAC):* Another possibility is to combine CTC and LAC when one wants to traverse as many nodes as possible while trying to obey the trajectory curve. Combining CTC and LAC can be done in various ways depending on importance of obeying the trajectory relative to importance of traversing as many nodes as possible.
- *Most Advancement on Curve (MAC):* If delay is of more importance, one might want to forward the packets to the farthest node along the curve. This is again a simple algorithm to implement however MAC forwarding may cause significant violations of trajectory as shown in Figure 4.
- *Lowest Deviation from Curve (LDC):* When obeying the trajectory is very crucial, it is possible to select the next node such that the taken route deviates from the trajectory as less as possible. However, this requires extra computations. We now describe how to implement such an algorithm.

In order to obey the trajectory at most level, at a current node N_0, the best next node N_i should be selected such that the line between N_0 and N_i must have the smallest deviation from the trajectory compared to the other lines between N_0 and any other node in N_0's neighborhood. Let A_i be the area between the line N_0-N_i and the curve, i.e. the total deviation of the forwarding from the trajectory. In order to minimize the average deviation from the trajectory, the next node selection must minimize ratio of A_i by the change in residuals $t_i - t_0$, i.e. the deviation from trajectory per unit length of the curve. So for node N_0, we can write the ratio to minimize as:

$$R_i = \frac{A_i}{t_i - t_0} = \frac{Area(N_0, N_i, Q(t_0), Q(t_i))}{t_i - t_0}$$

for all N_i in neighborhood of N_0. Figure 4 shows big picture of the necessary area calculations for LDC forwarding at node N_0. To illustrate an example, N_0 needs to calculate $A_1 = a_1 + a_2 + a_3$, $A_2 = a_1 + a_4$, and $A_3 = a_1 + a_2 + a_5$.

The problem is that, however, calculation of A_i requires extra computations and is not trivial. Closed-form analytical expressions for A_i are very hard to obtain. Fortunately, we can approximate A_i by numerical techniques similar to the method of Riemann sums [14] in numerical integration. For a detailed description on how to approximate A_i, please refer to [13].

IV. More Complex and Longer Trajectories

If we consider applications such as traversing a river or eastern face of a mountain, these applications will require consider-

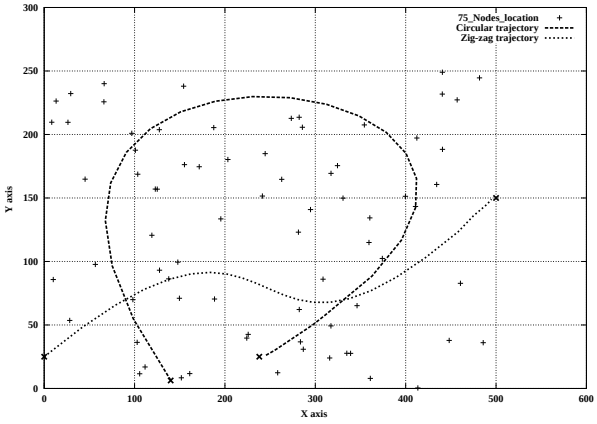


Fig. 5. Experimental trajectories seen in a scenario with 75 nodes.

ation of curves which could be represented by using much more number of control points than two. Such a curve will be very difficult to encode as a single Bezier curve in the packet header because we will have to include more than two control points. Also, computing such a Bezier curve is extremely difficult during the time of greedy forwarding.

One way to define long trajectories is to split the trajectory into smaller pieces which can be represented by cubic Bezier curves (i.e. 2 control points). Before starting the actual transmission of data, the source can probe the network by sending a control-plane packet which includes the whole trajectory with n control points. Upon arrival of that probe packet, an intermediate node divides the whole trajectory into equal pieces[2], and checks whether itself is close enough (e.g. within 5m of radius) to one of those middle points. If so, that particular node identifies itself as a *Special Intermediate Node (SIN)* for this source-destination pair and sends an acknowledgement to the source. The source confirms SIN by replying to the acknowledgement (this is necessary to resolve contention for being SIN if there are multiple candidates close to the desired middle point). After this confirmation from the source, the SIN records the control points for the next cubic Bezier curve in the trajectory. This process continues until all pieces of the trajectory is captured by a SIN.

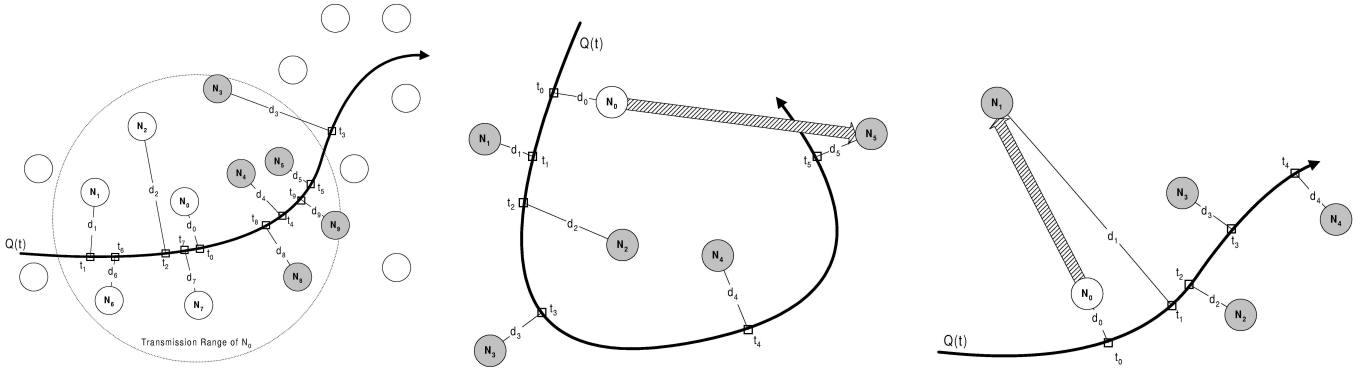
In this manner, there will be $n-1$ SINs, n cubic Bezier curves for a trajectory with $2n$ control points. After such a signaling protocol as described above, the source will no longer have to encode the $2n$ control points into data packets. Rather, it will just need to put 2 control points for the next cubic Bezier curve on the trajectory, since the next SIN will be putting the control points necessary for the following piece of the trajectory.

V. Simulations

Our purpose of ns-2 simulations is to evaluate the forwarding algorithms we have developed for TBR. We particularly look at two metrics: average deviation from trajectory and average path length. Since it is not necessary for this initial purpose above, we do not include mobility in our simulations.

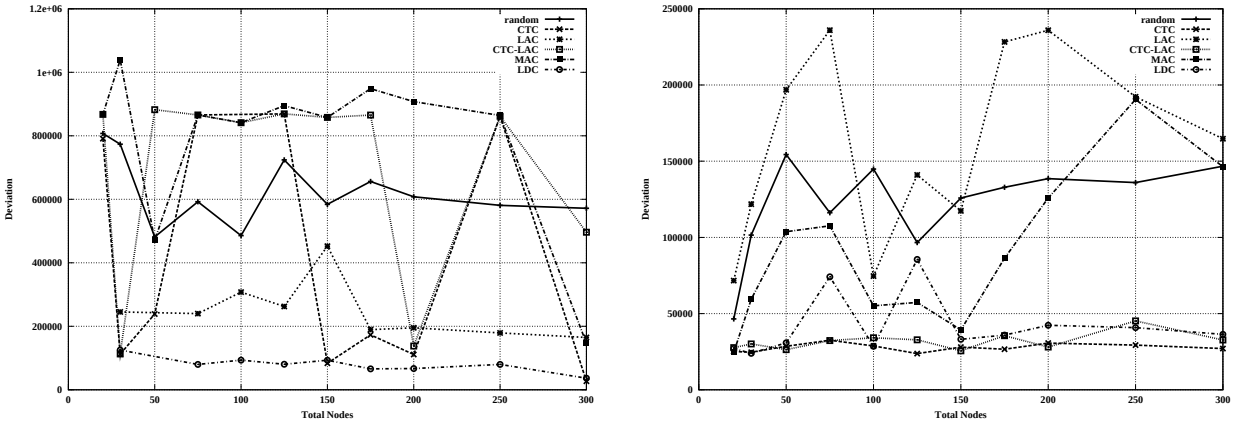
We simulated the forwarding algorithms for two different trajectories: circular and zig-zag. Trajectories are shown in Figure 5 over a scenario with 75 nodes. We varied number of nodes in the simulation from 20 to 300. Each node is a wireless node with an omnidirectional antenna. Total simulation time is 1000s.

[2]For a Bezier curve $Q(t)$ with n control points, these pieces are portions of the whole curve in between points $Q(t/k)$ where $k = 0..n$.



(a) Big picture of TBR forwarding (b) Failure of CTC and MAC forwarding (c) Failure of LAC forwarding

Fig. 3. Big pictures of various TBR concepts.



(a) Circular topology (b) Zig-zag topology

Fig. 6. Average deviation from trajectory in simulation experiments.

Figures 6-a and 6-b show average deviation of packets's routes from the ideal trajectory, for the case of circular and zig-zag trajectories respectively.

VI. Summary

In this paper, we studied Trajectory-Based Routing(TBR) for stateless routing in ad-hoc sensor networks. We proposed using Bezier curves for defining trajectories in TBR. Various shapes for routes can be defined by using Bezier curves. We particularly developed several forwarding algorithms based on trajectories defined by Bezier curves.

We proposed an optimal forwarding algorithm, Least Deviation from Curve(LDC), that obeys to trajectories the most. We ran extensive simulations in order to evaluate the forwarding algorithms. We found that LDC is good for moderately populated ad-hoc networks. Interestingly, we also found that Random forwarding performs average while avoiding significant computational overhead.

We also proposed an initial methodology for extending TBR with Bezier curves to longer and more complex trajectories which can be encoded by larger information. Our proposed method enables routing of data packets through complex trajectories, while keeping the packet header size constant. Future work will include evaluation and improvement of this method with a particular consideration given to signaling overhead.

Several issues remain to be investigated such as effect of mobility patterns, traffic patterns. Also, future work includes studying methods for increasing resilience (i.e. probabiliy of reaching to destination) for different forwarding algorithms. Finally, as another open issue, answering the question of how to route the packets to destination when the destination and the source are mobile, which is generally the case in ad-hoc networks.

References

[1] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Proceedings of ACM MOBICOM*, 2000.

[2] G. Finn, "Routing and addressing problems in large metropolitan-scale networks," Tech. Rep., University of Southern California, March 1987.

[3] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad-hoc wireless networks," *Mobile Computing*, vol. 353, 1996.

[4] E. Rosen, A. Viswananthan, and R. Callon, "Multiprotocol label switching architecture," *IETF RFC 3031*, February 2001.

[5] D. Ganesan, R. Govindhan, S. Shenker, and D. Estrin, "Highly resilient, energy efficient multipath routing in wireless sensor networks," *Mobile Computing and Communications Review (MC2R)*, vol. 1, no. 2, 2002.

[6] D. Niculescu and B. Nath, "Routing on a curve," in *Proceedings of Workshop on Hot Topics in Networks (HOTNETS-I)*, 2002.

[7] J. Follows and D. Straeten, *Application-Driven Networking: Concepts and Architecture for Policy-Based Systems*, IBM Red Book, 1999.

[8] B. Parkinson et al., *Global Positioning System: Theory and Application*, vol. 163, Progress in Astronautics and Aeronautics, 1996.

[9] D. Niculescu and B. Nath, "Ad-hoc positioning system (aps)," in *Proceedings of GLOBECOM*, 2001.

[10] J. C. Navas and T. Imielinski, "Geographic addressing and routing," in *Proceedings of ACM MOBICOM*, 1997.

[11] J. Li et al., "A scalable location service for geographic ad-hoc routing," in *Proceedings of ACM MOBICOM*, 2000.

[12] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," in *Proceedings of ACM MOBICOM*, 2001.

[13] M. Yuksel, R. Pradhan, and S. Kalyanaraman, "Trajectory-based forwarding mechanisms for ad-hoc sensor networks," Tech. Rep., Rensselaer Polytechnic Institute, http://networks.ecse.rpi.edu/~ yuksem/bezier.pdf, 2003.

[14] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, 1992.