An Implementation Framework for Trajectory-Based Routing in Ad-Hoc Networks

Murat Yuksel, Ritesh Pradhan and Shivkumar Kalyanaraman
Electrical, Computer and Systems Engineering Department
Rensselaer Polytechnic Institute, Troy, NY 12180
yuksem@rpi.edu, rspradhan@alum.rpi.edu, shivkuma@ecse.rpi.edu

***Abstract*— Routing in ad-hoc networks is a complicated task because of many reasons. The nodes are low-memory, 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, multi-path 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. In this paper, we investigate Trajectory-Based Routing (TBR) which was proposed as a middle-ground between SBR and greedy forwarding techniques. We address various issues regarding implementation of TBR. We also provide techniques to efficiently forward packets along a trajectory defined as a parametric curve.**

I. Introduction

Because of its stateless nature, greedy forwarding (e.g. FACE [1], GPSR [2] and Cartesian Routing (CR) [3]) of packets at intermediate nodes is desirable in ad-hoc networks. Also, for traffic engineering, multi-path capabilities (e.g. Source Based Routing (SBR) [4]) are desirable. However, it is not possible to employ well-known multi-path routing techniques (e.g. MPLS [5], or others [6]) in ad hoc, particularly mobile, networks. Niculescu and Nath [7], [8] 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. Also, TBR needs support for positioning of wireless nodes. In this paper, we assume that a positioning service is available. This assumption is reasonable as the use of GPS [9] as well as other GPS-free positioning tools are becoming more popular [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. For trajectory encoding, we propose to use Bezier curves [13] which give a lot of flexibility in the greedy forwarding of TBR while it is possible to define a broad range of curves with them. We also describe a protocol for implementing longer and more complex trajectories as a concatenation of Bezier curves. Given this trajectory encoding technique at source, we present various mechanisms to perform forwarding at intermediate nodes.

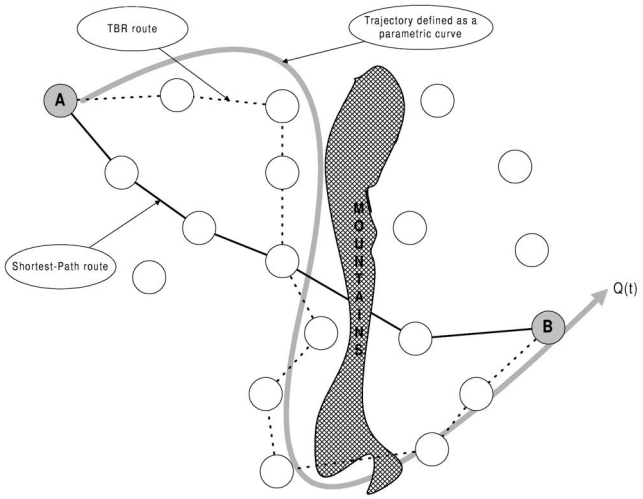


Fig. 1. An example for using TBR in an application: The application collects photos of the "west of mountains", which causes best route to be different than traditional shortest-path routing.

Contributions of this paper can be listed as follows:

- Implementation of TBR using Bezier curves, which gives enough flexibility to define a large range of trajectories (e.g. zig-zag, circular).
- An implementation protocol of TBR for longer and more complex trajectories, virtually without any limit on the length and complexity of the trajectory.
- Implementation of the forwarding methodology, Least Deviation from Curve(LDC), that performs optimally in terms of obeying the trajectory in forwarding.
- Evaluation of the forwarding technique LDC as well as several other intuitive forwarding techniques.

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. In Section III, we briefly describe ways of bounding packet header size for trajectory encoding with Bezier curves. Next in Section IV, we propose various greedy algorithms for packet forwarding in TBR with Bezier curves. In Section V, we present ns-2 simulations of the forwarding algorithms and evaluate their performance.

II. Using Bezier Curves 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 one *control* point is called as **quadratic Bezier curve**, while the one which is defined by two *control* points is known as **cubic Bezier curve**. More details about basic calculations for Bezier curves can be found in [13]. There are other forms of Bezier curves such as **quintine Bezier curves** (three control points), but our choice of using cubic Bezier curve was dictated by its computational simplicity.

A. Forwarding Along a Cubic Bezier Curve

Shape of a Bezier curve is dependent on the locations of the control points. A sample cubic Bezier curve is shown in Figure 2-a. It can also be represented in its parametric form, $Q(t)$. When parameter $t = 0$, it represents the source point of the curve, while $t = 1$ represents the destination point of the curve. More specifically, a cubic Bezier curve is represented algebraically as:

$$Q(t) = \mathbf{X} = \mathbf{A}\,t^3 + \mathbf{B}\,t^2 + \mathbf{C}\,t + \mathbf{X}_0 \qquad (1)$$

where

$$\mathbf{X} = \begin{bmatrix} x \\ y \end{bmatrix} \mathbf{A} = \begin{bmatrix} a_x \\ a_y \end{bmatrix} \mathbf{B} = \begin{bmatrix} b_x \\ b_y \end{bmatrix} \mathbf{C} = \begin{bmatrix} c_x \\ c_y \end{bmatrix} \mathbf{X}_0 = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}.$$

For the third order polynomial shown in (1), the coefficients **A**, **B**, **C** are unique if the following equation system is satisfied[1]:

$$\begin{aligned} \mathbf{C} &= 3\,(\mathbf{X}_1 - \mathbf{X}_0) \\ \mathbf{B} &= 3\,(\mathbf{X}_2 - \mathbf{X}_1) - \mathbf{C} \\ \mathbf{A} &= \mathbf{X}_3 - \mathbf{X}_0 - \mathbf{C} - \mathbf{B} \end{aligned} \qquad (2)$$

Here, $\mathbf{X}_0$, $\mathbf{X}_1$, $\mathbf{X}_2$, and $\mathbf{X}_3$ are vectors similar to $\mathbf{X}$ containing the *x* and *y* coordinates of *source point*, *control point-1*, *control point-2*, and *destination point* respectively.

So, given the coordinates of the source (x_0, y_0), destination (x_3, y_3), and the two control points (x_1, y_1) and (x_2, y_2), one can calculate constants **A**, **B**, and **C** from the equation system (2), thereby recovering the complete Bezier curve.

Our idea is to use this mathematics to encode the complete trajectory into each packet, by putting the coordinates of source, destination, and the two control points into packet header. Then, solve the equation system in (2) to decode the complete trajectory at any intermediate node.

In order to implement forwarding algorithms, we need to find where each node corresponds on the trajectory. This is actually the point on the curve closest to the node, which can be found by well-known algebraic techniques.

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-b, 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.

[1]Please refer to [13] for proof.

III. Long or More Complex Trajectories

If we consider applications such as traversing a river or capturing eastern face of a mountain into a picture, these applications will require consideration 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 in the packet header, because we will have to encode each and every control point which would make the header bulky. Also, computation for decoding such a Bezier curve is extremely difficult during the time of greedy forwarding.

As shown in Figure 2-c, one way to define long trajectories is to split the trajectory into smaller pieces which can be represented by cubic Bezier curves (i.e. two control points). So, the complete trajectory is defined as a concatenation of a series of cubic Bezier curves. We call the concatenation points as *middle points*, shown as I_1 and I_2 in the Figure 2-c.

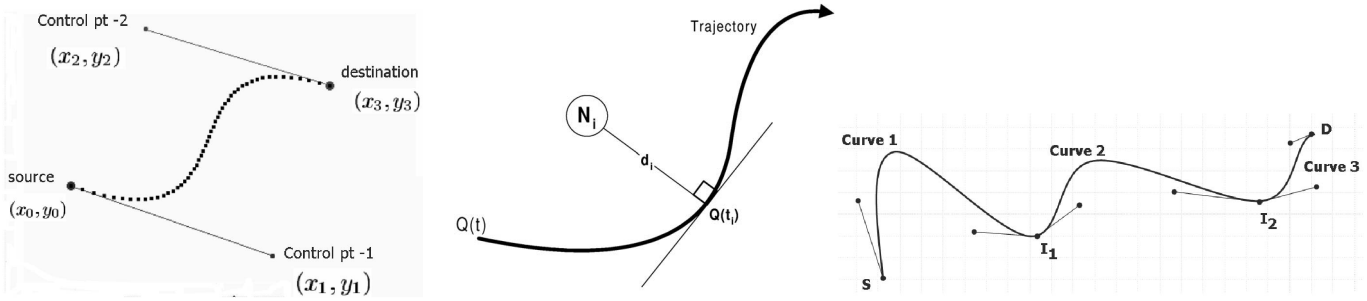
Before starting the actual transmission of data, the source can probe the ad-hoc network by sending a control-plane packet which includes the whole trajectory with n control points. Upon arrival of that probe packet, an intermediate node divides the whole trajectory into equal pieces[2], and checks whether 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 will continue until all pieces of the trajectory is captured by a SIN which keeps the control points of the next cubic Bezier curve.

IV. 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, reach the destination node, reach quickly.

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

[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) A cubic Bezier curve. (b) A node near trajectory $Q(t)$. (c) Concatenation of cubic Bezier curves.

Fig. 2. Bezier curves and their usage in TBR.

of nodes that have residuals larger than t_0 as *neighborhood*[3] *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.

In the following sub-sections, we develop various algorithms for selection of next node within the neighborhood.

A. 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. Also, if transmission power of nodes in the network is relatively small, then this algorithm will perform fine since nodes will not have very large neighborhoods that may cause packets to be forwarded far away from the trajectory. So, the Random algorithm may be useful for wireless networks with nodes having low computational and transmission power.

B. 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. Simply, calculate residual distances of each node in the neighborhood and select the one resulting in the smallest residual distance.

If obeying to the trajectory and computational power are both important, then CTC may be useful. 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.

C. Least Advancement on Curve (LAC)

One might need to traverse all the nodes that are along the trajectory curve, e.g. flooding. A simple algorithm is to forward to the node whose residual lies right next to the residual of the current node. Note that this algorithm is also useful for low computation powered networks. However, again, this might result in significant errors in forwarding. For example, packets could be forwarded to a node very far away from the trajectory, just because it is its residual is the least among possible neighbors.

[3]Note that our definition of neighborhood is different from Niculescu and Nath's definition in [8].

D. 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. Assuming that obeying to the trajectory is more important, a computationally simple algorithm is as follows.

First, define a tolerable residual distance D. Then, go through the neighborhood and try to find a neighbor node N_i having residual distance $d_i < D$. If there are multiple nodes satisfying the condition $d_i < D$, then select the one with smallest residual t_i. If there is no nodes satisfying the condition, then increment D with a step value $\triangle D$ and try again until a node is selected as the next node.

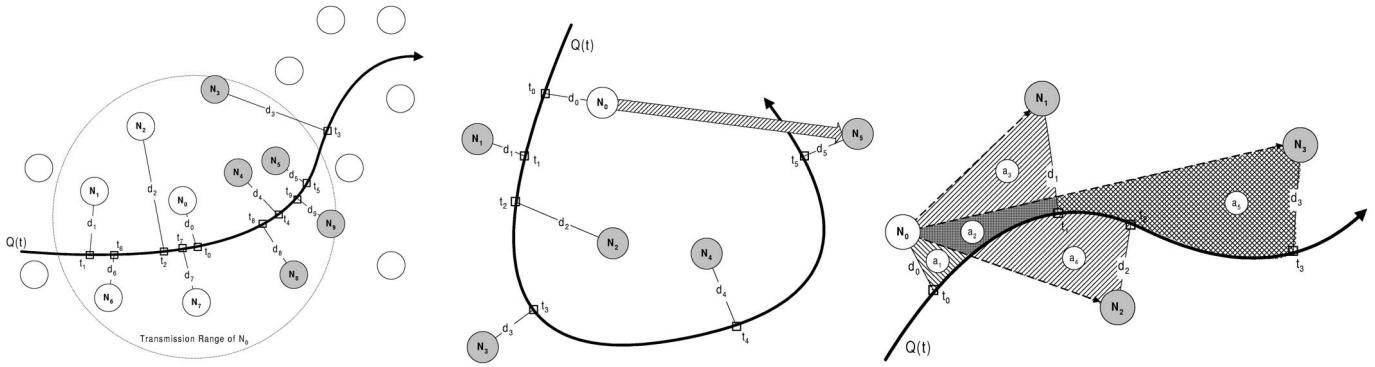
E. Most Advancement on Curve (MAC)

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 since just calculation of residuals will be enough in order to find out the farthest node to the current node. However, MAC forwarding may cause significant violations of trajectory as shown in Figure 3-b. Similar to CTC-LAC, it is also possible to combine CTC with MAC.

F. Lowest Deviation from Curve (LDC)

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:

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



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

Fig. 3. Big pictures of various TBR concepts.

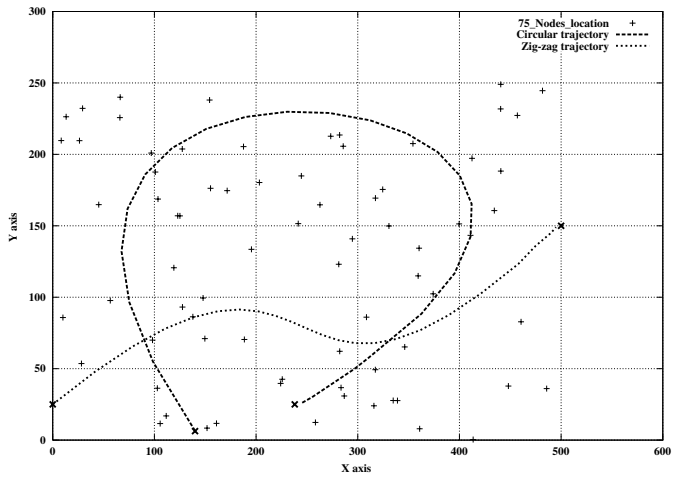


Fig. 4. Experimental single-piece trajectories: Zig-zag and circular single-piece trajectories on randomly located 75 nodes.

for all N_i in neighborhood of N_0. Figure 3-c 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.

V. Simulations

We performed ns-2 simulations to evaluate the forwarding algorithms developed for TBR, via two metrics: average deviation from unit trajectory and average path length.

We simulated the forwarding algorithms for two different trajectories: circular and zig-zag. Trajectories are shown in Figure 4-a 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. Transmission range of antennas is 5m in radius. The wireless nodes are exchanging beacons with an interval of 10s. Each node maintains a neighbor table with a beacon timeout of 110s. The nodes are randomly distributed over a rectangular area 250mX500m. We picked a source-destination pair such that source and the destination are close to the starting and ending points of the trajectory respectively. The source generates CBR traffic with average packet size of 0.5KB. Total simulation time is 1000s.

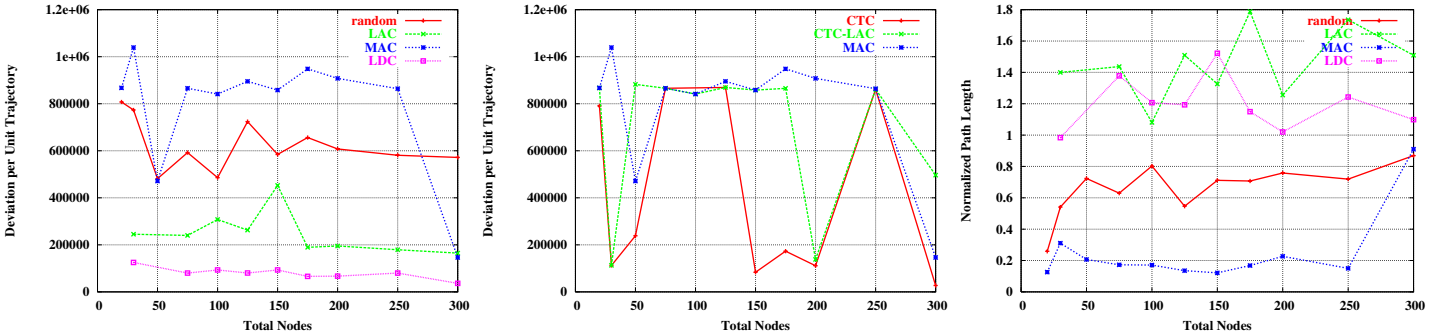
Figures 5-a-b and 6-a-b show average deviation of packets' routes from the ideal trajectory, for the case of circular and zig-zag trajectories respectively. We observe that LDC is outperforming the other forwarding algorithms in the case of circular trajectory. Sometimes, CTC outperforms LDC which explains the fact that LDC is making local optimization without considering next hop's choice. This causes CTC to win sometimes. In both trajectories, we see that LDC and CTC is converging to each other as density of nodes increases. However, we observe CTC failure (as in Figure 3-b) in some cases, e.g. when number of nodes is 250 in circular trajectory.

Also, LAC and MAC performs worse than the others in general, which is caused by LAC's and MAC's ignorance on obeying to trajectory. As expected, CTC-LAC performs in between CTC and LAC. Nicely, we observe that Random forwarding algorithm performs average compared to others.

Figures 5-c and 6-c show average path length traversed by packets normalized to the length of the ideal trajectory, for the case of circular and zig-zag trajectories respectively. We can observer that, as expected, LAC performs worst in terms of path length. MAC outperforms all the other for the circular trajectory, however it is beaten by CTC and CTC-LAC for the zig-zag trajectory. That difference becomes more evident as density of nodes increases.

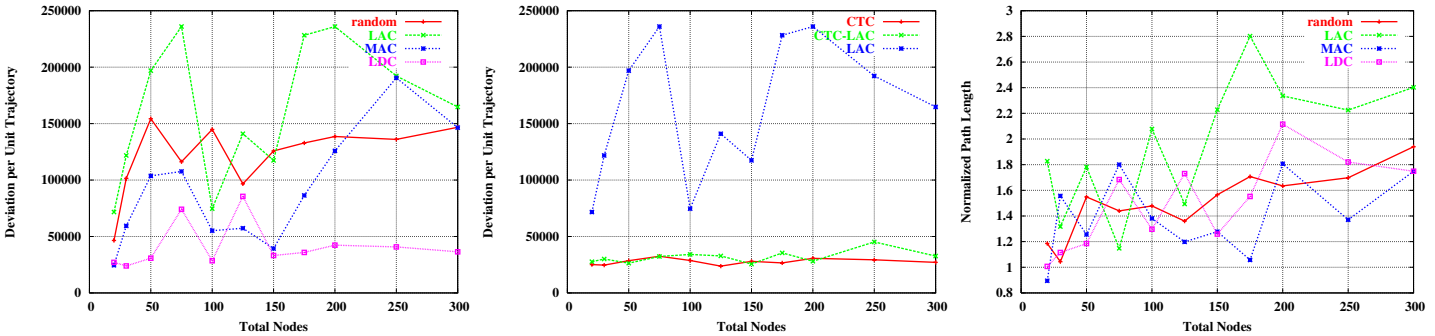
For the circular trajectory, normalized path length is approximately 1 for LDC, which also shows that LDC is the one that obeys the trajectory most. However, for zig-zag trajectory, LDC becomes larger than 1 as density of nodes increases. This means LDC is best for moderately populated ad-hoc networks. This discourages use of LDC for very dense networks since its computational overhead is more for denser networks (as number of neighbors will increase too).

Also, Random again performs average compared to the others in terms of path length. So, an interesting finding is that Random forwarding is good in order to achieve an average performance while avoiding a lot of computational overhead



(a) Deviation from trajectory (b) Deviation from trajectory (c) Normalized path length

Fig. 5. Simulation results for the **circular trajectory**. (a)-(b): Average deviation from trajectory. (c): Path length normalized to trajectory length.



(a) Deviation from trajectory (b) Deviation from trajectory (c) Normalized path length

Fig. 6. Simulation results for the **zig-zag trajectory**. (a)-(b): Average deviation from trajectory. (c): Path length normalized to trajectory length.

of more complex forwarding mechanisms. For Random forwarding, probability of reaching destination was more than 80%. For all the others, it was more than 95%.

VI. Summary

In this paper, we studied various implementation issues of Trajectory-Based Routing(TBR) for stateless routing in ad-hoc networks. We proposed to use Bezier curves for defining trajectories in TBR. We particularly evaluated several forwarding algorithms based on trajectories defined by Bezier curves. We ran extensive simulations in order to evaluate the forwarding algorithms. By introducing signaling phase to the protocol, we also proposed a 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 for different forwarding algorithms. Finally, routing for mobile source and destination is also an open issue.

Acknowledgment

This work is funded by NSF grants NSF-STI 0230787 and NSF-ITR 0313095, and also by Intel Corporation.

References

[1] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks," *ACM/Kluwer Wireless Networks*, vol. 7, no. 6, pp. 609–616, 2001.

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

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

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

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

[6] 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.

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

[8] D. Niculescu and B. Nath, "Trajectory based forwarding and its applications," in *Proceedings of ACM MOBICOM*, 2003.

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

[10] Y.-B. Ko and N. H. Vaidya, "Location-aided routing (lar) in mobile and ad-hoc networks," in *Proceedings of ACM MOBICOM*, 1998.

[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] P. J. Schneider and D. H. Eberly, *Geometric Tools for Computer Graphics*, 2002.

[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.