Routing on a Curve *

Badri Nath and Dragoş Niculescu
Department of Computer Science
Rutgers University
Piscataway, NJ 08854
{badri,dnicules}@cs.rutgers.edu

ABSTRACT

Relentless progress in hardware technology and recent advances in sensor technology, and wireless networking have made it feasible to deploy large scale, dense ad-hoc networks. These networks together with sensor technology can be considered as the enablers of emerging models of computing such as embedded computing, ubiquitous computing, or pervasive computing. In this paper, we propose a new paradigm called trajectory based forwarding (or TBF), which is a generalization of source based routing and Cartesian routing. We argue that TBF is an ideal technique for routing in dense ad-hoc networks. Trajectories are a natural namespace for describing route paths when the topology of the network matches the topography of the physical surroundings in which it is deployed which by very definition is embedded computing.

We show how simple trajectories can be used in implementing important networking protocols such as flooding, discovery, and network management. Trajectory routing is very effective in implementing many networking functions in a quick and approximate way, as it needs very few support services. We discuss several research challenges in the design of network protocols that use specific trajectories for forwarding packets.

Categories and Subject Descriptors

C.2.1 [**Network architecture and design**]: Wireless communication; C.2.2 [**Network protocols**]: Routing protocols

Keywords

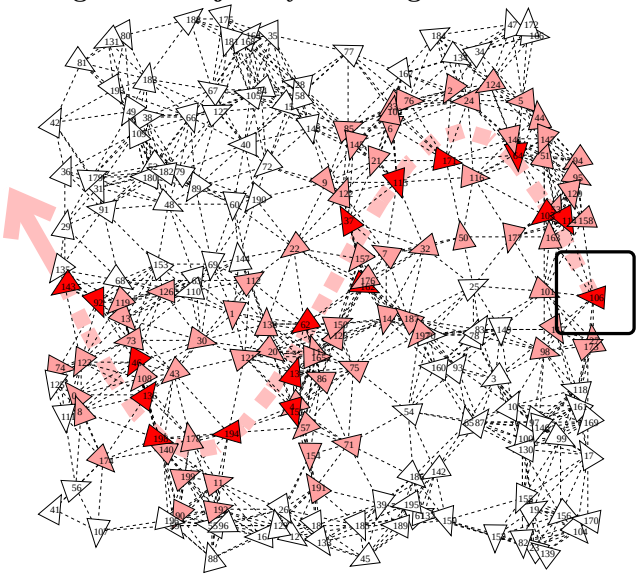
ad hoc networks, trajectory based forwarding, routing

1. INTRODUCTION

Routing packets along a specified curve is a new approach to forwarding packets in large scale dense ad-hoc networks.

*This research work was supported in part by DARPA under contract number N-666001-00-1-8953

Figure 1: Trajectory following a sine curve



The fundamental aspect of considering route paths as continuous functions is the decoupling of the path name from the path itself. The transition from a *discrete view of route path to a continuous view of route path* is important as we move from dealing with sparse networks to dealing with dense networks. The basic idea of routing on curve is to embed the trajectory of the route path in the packet and then let the intermediate nodes forward packets to those nodes that lie more or less on the trajectory. Representing route paths as trajectories is an efficient scalable encoding technique in dense networks. Since a trajectory does not explicitly encode the nodes in the path, it is impervious to changes in specific nodes that make up the topology. We believe that trajectories are a natural namespace to describe route paths when the topology of the network matches the topography of the physical surroundings in which it is deployed which by very definition is embedded computing. Forwarding packets along trajectories can be very effective in implementing many networking functions when standard bootstrapping or configuration services are not available, as will be the case in disposable networks where nodes are thrown or dropped to form a one-time use network.

To effectively forward packets along a trajectory, all that

is needed is a sufficiently dense network and a capability by which nodes can position themselves relative to a coordinate system and estimate distance to its neighbors. The coordinate system could be a relative coordinate system [1], global coordinate system (GPS [2]), or an ad-hoc coordinate system (APS [3]). While an accurate global positioning system such as GPS would be ideal, and some argue that it will be available at a form and cost factor suited for universal deployment, others argue that it is not reasonable to expect GPS to serve all purposes due to non-line of sight and precision requirements. In any case, we believe TBF is an effective mechanism for routing in dense ad-hoc networks. In fact, it can be viewed as a network primitive that serves as a glue between positioning providing services such as GPS, APS[3], or GLS[4] and many applications such as flooding, discovery, routing schemes for path resilience, unicast, multicast and mobile ad hoc routing.

For example, in figure 1 a trajectory representing a sine wave ($x = t$, $y = \sin t$, with the appropriate rotation transformation applied to orient the curve as desired) can be specified in the packet by the source. By using a forwarding technique that delivers the packet to nodes that are close to the trajectory, the packet can be made to traverse the shape of the trajectory specified in the packet as shown in figure . The dashed line is the specified path and the dark shaded nodes are the nodes that forward the packets, while the light shaded ones overhear the packet from the broadcast medium without actually forwarding it. The simulated network is a unit disk graph of 200 randomly positioned nodes.

This technique of forwarding packets along a curve or a trajectory is a generalization of source based routing[5, 6] and Cartesian routing[7, 8]. As in source based routing, trajectories are specified by the source but do not explicitly encode the path on a hop-by-hop basis. As in Cartesian routing, the nodes are selected based on proximity but the proximity is to the trajectory instead of the final destination. TBF combines the best of the two methods: packets follow a trajectory established at the source, but each forwarding node makes a greedy decision to infer the next hop based on local position information, while the overhead of representing the trajectory does not depend on path length. In a network where node positions are known, the packet may be forwarded to the neighbor that is geographically closest to the desired trajectory indicated by the source node. If the destination node is known, the trajectory followed by the packet might be a line, and the method reduces to Cartesian forwarding. A recent work that addresses routing scalability is anchored geodesic packet forwarding [9], where the source specifies a list of anchors between which Cartesian routing is to be used. Other routing solutions in wireless network are complementary to TBF, such as algorithms for guaranteed delivery. The GFG algorithm presented in [10, 11] can be adapted to work with arbitrary trajectories, thus providing guaranteed delivery for routing schemes based on TBF.

Trajectory based forwarding (TBF) has a number of unique features that makes it a candidate as a primitive in dense ad-hoc networks. Here are some distinguishing features of trajectory based forwarding:

- Forwarding based on trajectories decouples the path name from the path itself. Since a trajectory can be specified independent of the nodes that make up the trajectory, routing can be very effective even when the intermediate nodes that make up the path fail, go into doze mode, or move. Reactive routing algorithms (LSR[5], DSR[6]) explicitly encode the path. Proactive route maintenance algorithms (AODV[12]) need to update topology changes. Thus, communication costs increase in proportion to the topology changes induced by mobility. In DSR, any changes to the path results in route discovery, while in AODV, results in propagation of route updates. In TBF, route maintenance is virtually free and unaffected by mobility rate, as the path specification is independent of the names of the nodes involved in forwarding.

- The specification of the trajectory can be independent of the ultimate destination(s) of the packets. In fact, the trajectory specified in the packets need not have a final destination! A packet can be let go along a line or a curve for a desired length. This has implications for implementing networking functions such as flooding, discovery and multicast.

- Trajectories are a natural namespace for specifying paths in embedded systems. Here, packets are expected to take routes that closely mirror the physical infrastructure in which the network is embedded. The topography of the network is a good indication of its topology.

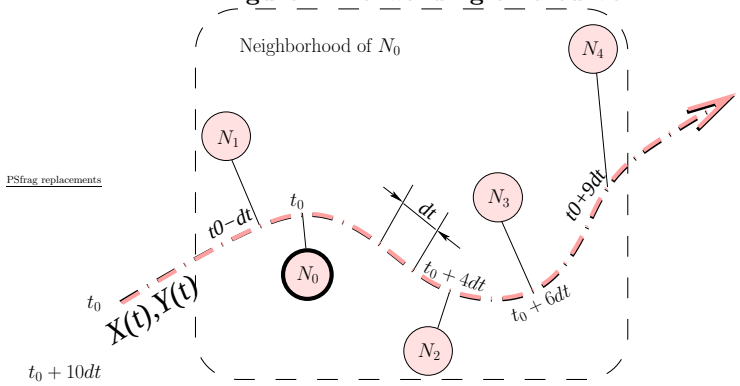
- For packets to follow the trajectory closely, nodes need to have the capability to position themselves in a coordinate system. A system such as GPS or APS would be sufficient for this purpose, but TBF can be made to work even without GPS, as a coordinate system relative to the source can be constructed by positioning only the nodes in the neighborhood of the trajectory. However, the accuracy of the actual path followed will depend upon the errors due to localizations and coordinate transformations.

- Multipath routing has several advantages, such as increased reliability and capacity, but is seldom used because of the increased cost of maintenance. Using TBF, an alternate path is just another trajectory requiring no more maintenance than the main trajectory.

2. FORWARDING ON A TRAJECTORY

The trajectory is usually decided by the source and can be conveniently expressed in parametric form $X(t), Y(t)$. The meaning of parameter t is also decided by the source, as well as the resolution at which the curve will be evaluated, indicated by dt. For the simplicity of the explanation, assume that t indicates the distance along the curve. The neighborhood of a node N_0(figure 2) is defined as the portion of the curve and the nodes that are within a certain distance from N_0, indicated by a dashed rectangle in the figure. In the simplest case, the neighborhood could be the smallest rectangle enclosing all N_0's one hop neighbors. In a network in which node positions are known, the main question is how to choose a next hop that best approximates the trajectory. Assume node N_0 receives a packet with the trajectory indicated by the curve $X(t), Y(t)$ and the value t_0 that corresponds to the point on the curve that is closest to N_0. Using

Figure 2: Forwarding on a curve

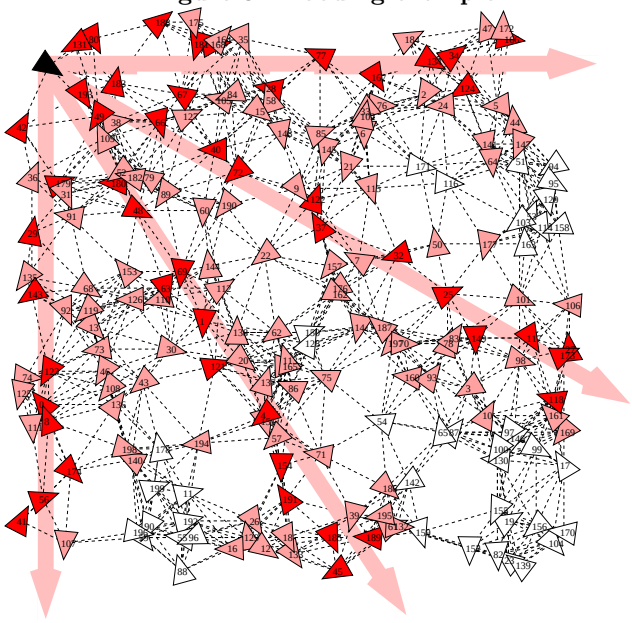


sampling of the curve at dt spaced intervals, indicated by dots in the dashed trajectory curve, N_0 can compute all the points of the curve that reside inside its neighborhood. For all neighbors $N_1..N_4$, their corresponding closest points on the curve are $t_0, t_0 + 4dt$, $t_0 + 6dt$, $t_0 + 9dt$. When referring to curve fitting, these values are called residuals. In fact, the mentioned method computes an estimation of the residuals instead of the true ones, which would require either infinite precision, or usage of higher derivatives of $X(t)$ and $Y(t)$. Since choosing a next hop for the packet should be towards advancement on the trajectory, only portion of the curve with $t > t_0$ is considered. For this reason, node N_1 receives t_0 as the closest point, instead of $t_0 - dt$, which would be closer to the perpendicular from N_1 onto the curve.

Several policies of choosing a next hop are possible:

- node closest to the curve, with the minimum residual. This policy would favor node N_2 and would tend to produce a lower deviation from the ideal trajectory. This strategy should be chosen when it is important for the packets to follow the trajectory closely to possibly determine the state around the trajectory. Since the packet stays close to the trajectory, there is less likelihood for a packet to wander away from the intended trajectory due to errors in localization.
- most advancement on t, choosing N_4. This policy should also be controlled by a threshold of a maximum acceptable residual in order to limit the drifting of the achieved trajectory. It would produce paths with fewer hops than the previous policy, but with higher deviation from the ideal trajectory. This strategy should be favored when delay is an important metric and the packet needs to reach the destination or the perimeter of the network in minimum number of hops.
- centroid of the feasible set, choosing N_3. The path traversed will be more coarse compared to the choice of choosing the node closest to the trajectory and will cover the center of activity of the region without having too many small hops. The centroid is a way to uniformly designate clusters along the trajectory, and a quick way to determine the state of the network along the trajectory.
- randomly choose among the best nodes obtained from above three choices. This is useful when node positions are imperfect, or when it may be necessary to route around unknown obstacles. The randomness in the choice of the next hop can help mitigate the effects of small obstacles.
- in mobile networks a forwarding policy that might provide better results would be to choose the next hop which promises to advance along the trajectory, or one that is expected to have the least mobility in the future. Another interesting choice would be to choose a node that is moving towards the trajectory, rather than the one that is moving away. This strategy is expected to work better when trajectories and neighborhood information are cached, and for a given trajectory, the packet is forwarded to the same node. However, either a periodic evaluation of the position of neighbors relative to the trajectory, or cache invalidation is necessary.

Figure 3: Flooding example



3. BASIC TRAJECTORIES AND APPLICATIONS

In this section we will show how some simple trajectories can be used to implement some basic networking functions such as flooding, discovery, and network management.

Flooding: using TBF, flooding can be replaced with a number of outgoing radial lines that are reasonably close to each other to achieve a similar effect without all the communication overhead involved by receiving duplicates in classical flooding. More generally, a source would indicate the directions and the lengths of the lines that would achieve a satisfactory coverage. The coverage relies on the typical broadcast property of the wireless medium, in which several nodes overhear the packet being forwarded.

In figure 3, the node in the upper left corner broadcasts along

Figure 4: Flooding performance

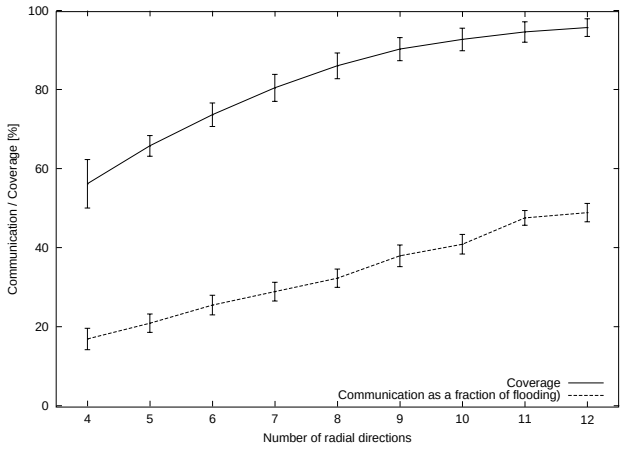
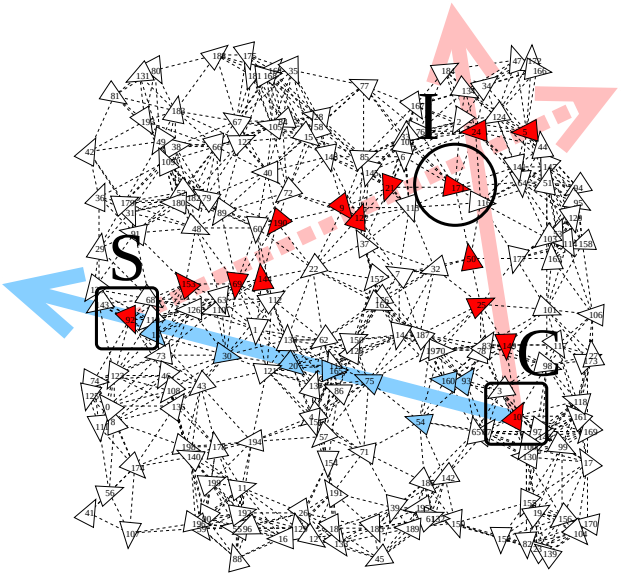


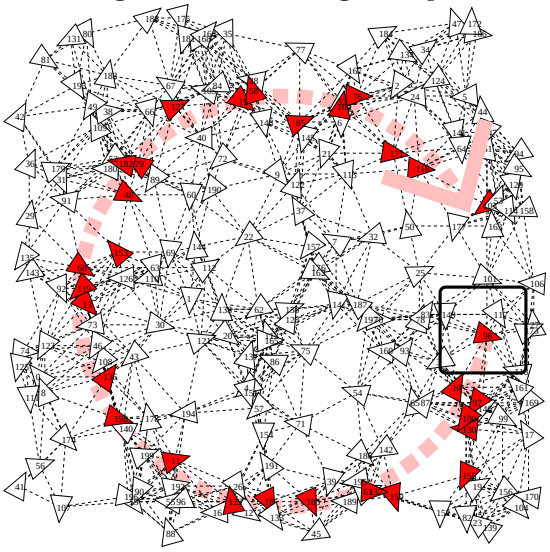
Figure 5: Discovery example



four directions achieving a coverage of 82%. Using another node in the middle of the network(157), and varying the number of spokes, we compare the number of packets used by regular flooding and TBF flooding. In figure 4, we can see that while TBF flooding can achieve almost complete coverage, it only uses half of the communication employed by regular flooding. Thus, flooding along spokes is a quick and dirty way of sending information for purposes of say, configuring sufficient number of nodes to bootstrap the network. Other trajectories such as H-trees or fractals can also be used to provide the required coverage. An interesting research issue is the tradeoff between coverage and the complexity of the trajectory specification and evaluation.

Resource discovery: many algorithms use initial discovery phases based on flooding[6, 13] in order to find a resource or a destination. A replacement scheme using trajectories is as follows(figure 5): servers S advertise their position along arbitrary lines and clients C will replace their flooding phase

Figure 6: Boomerang example



with a query along another arbitrary line which will eventually intersect the server's line. The intersection node I then notifies the client about the position of the destination. The client can then form another trajectory and send requests directly to the server. Discovery and subsequent routing can be done without knowing the position of any of the nodes, as a trajectory can be built in the relative coordinate system of the client. To guarantee that the client and server lines always intersect, it is necessary to send four spokes from both server and client in four cardinal directions (independently and randomly chosen by each). Another necessary condition is that the network be large enough to accommodate the intersection point, and for this, it has to be large enough to fit a circle with diameter CS.

Management: An interesting trajectory is a **boomerang,** where the packet takes a certain route and returns to the sender (figure 6). A circle is a simple example of this trajectory and can be used to secure the perimeter of the network by sending a challenge response token along the trajectory. All nodes who respond properly can be considered to be authenticated. A packet that returns to the source after being routed along a circle or a boomerang path is in effect implementing a self ARQ.

4. RESEARCH CHALLENGES

4.1 Specifying and determining trajectories

There are a number of choices in representing a trajectory: functional, equation, or a parametric representation. Functional representation(e.g. $y(x) = ax + b$) cannot be used to specify all types of curves, such as vertical lines. Equation representation (e.g. $x^2 + y^2 = R^2$) requires explicit solution to determine the points on the curve and cannot easily handle the notion of progress. Parametric representation is ideally suited for the purpose of forwarding. The parameter of the curve is a natural metric to measure the forward progress along the path and can be considered a proxy for the hop count. Given the choice of a parametric form, the next issue is how the trajectories should be interpreted. Trajectories can have several parameters and each node needs

to correctly interpret these parameters. One approach is to have a convention where the nodes know how to interpret the fields given a well known set of trajectories.

A trajectory can also be composed of several simple trajectories. These simple trajectories can be viewed as segments and can be specified along with the appropriate intervals of the parameter over which it is valid. To obtain a continuous trajectory, these intervals need to overlap. A trajectory may involve a complex shape that may not have a simple parametric representation. However, in many cases, this shape can be represented by a simpler set of Fourier components. The more Fourier components in the specification of the trajectory, the better is the reproduction of the trajectory. There is an interesting tradeoff between the accurate reproduction of the trajectory and the overhead of specifying the components and interpreting them. Other flexible and compact ways to encode a complex trajectory are fractal encoding and compiled form encoding. The latter approach, also used in active networking[14], sends in each packet the binary code needed for the parametric evaluation of the curve.

Another important question is how the trajectories are determined. In our initial work on TBF, we assumed that the source knows the trajectories a priori and that is normally derived from the application at hand. This may be the case in many applications of embedded computing where the topology of the network closely mirrors the underlying physical infrastructure in which the network is embedded. As was shown with applications such as flooding and discovery, simple lines or rays are sufficient. However, in other circumstances, the source or the intermediate nodes may have to determine the actual trajectory given the destination or group of destinations. Our initial thinking is to rely on a trajectory mapping service similar to DNS where, given a destination, or a group of destinations, the mapping service returns a trajectory for the source to use. How such a service can be built is a topic for further research.

4.2 Modifying trajectories

Once the source has determined the trajectory to use for routing, it may have to be modified due to network conditions such as obstacles, failures or mobility. The question is how to detect that a modification is needed and who should modify the trajectory. A route failure detection mechanism or an ACK/NACK scheme can be used for detecting the need for a change in the trajectory. A source can then choose a new trajectory for forwarding subsequent packets. Here, the onus of successful delivery is completely on the source. Another choice would be for some authorized intermediate nodes to detect the need for modification and provide a "patch" to the trajectory so that packets can be forwarded around obstacles, failures or local routability conditions such as network congestion. Whatever the reason for providing these patches, the patches should only serve as local detours where the packets eventually get back on track and are forwarded along the original trajectory.

4.3 Implementing network protocols

TBF can be used in implementing many network protocols in ad hoc networks. In static networks, such as sensor networks, packets can be forced by the trajectory to take a certain route around obstacles or congestion. If the destination is mobile and its path is known, it is possible to route towards this path. This means that TBF can naturally use information about trajectories of destinations rather than locations of destinations.

Multipath routing is used for either path resilience[15], or load balancing. Trajectories can easily specify disjoint or braided paths between a source and a destination without the cost of route discovery. Different packets can take different paths, since setting up a route is virtually free.

Routing to multiple destinations has well known uses such as flooding and multicast, but also application specific uses, such as temporary groups which do not justify the cost of setting up as multicast groups. TBF accepts specifying arbitrary trees as trajectories and all nodes along the tree will receive the packet. The tree is the actual physical tree representing the location of recipients. Nodes close to branching points in the tree have the task of duplicating data for the branches downstream. To reduce the overhead of representing the tree in its entirety, the tree representation can be pruned off as the packet is forwarded downstream.

Routing in mobile, ad hoc networks can make use of TBF to forward packets. Since trajectories do not explicitly encode the members of the path, it is less likely to result in a route failure when one more nodes along the path move and there are other nodes close to the path that can forward the packet. Even source mobility is not a problem, as long as the source can determine the new trajectory towards the destination. The important research issue is how to deal with the mobility of the destination. A combination of TBF and location-aided routing[16] may be needed. Another strategy would be to use the knowledge of the trajectory of the mobile node and construct a trajectory that uses a path from the source to a point on the mobile node's trajectory. As long as the destination is moving along the specified trajectory, the packets will reach a mobile destination.

Problems that need further exploration include: who is to determine the trajectory and how (given all destinations and obstacles), how to deal with route failures (for example by patching the trajectory based on local knowledge that is not available globally), and what is the right balance between node centric and position centric addressing.

4.4 Use of positioning techniques

In many networks GPS is not available, but TBF can make use of alternative positioning methods based on heterogeneous capabilities, such as angle measurement, range measurement and compasses. One option is to run a network wide, ad hoc positioning algorithm, such as APS[3]. Another option is to use multimodal sensing to determine local ranges and angles, and eventually to set up local coordinate systems which allow handling of trajectories. This, however, involves successive coordinate systems alignment and is more sensitive to the measurement errors than the first option. For all the mentioned positioning options, the performance of TBF (accuracy, drift) depends on the quality of the positions obtained at each node and therefore, ultimately, on the quality of the measurements.

4.5 Error Management

Errors in forwarding have two main causes: trajectory encoding and measurement error. Errors might be produced by encoding when a complex trajectory is not represented accurately, in order to reduce overhead. Measurement errors affect TBF when multimodal sensing is used for positioning (when GPS is not available). Depending on the hardware available on the nodes, errors accumulate along the trajectory and can be modeled as different random walk processes. If, for example, a compass is available in each node, the actual trajectory might drift left or right of a true linear trajectory, but will not change direction. If only ranges or angles are used for local positioning the trajectory may completely go off course. Characterizing these errors analytically would help in providing confidence areas for points on the achieved trajectory. Another research direction would be to investigate the possibility of correcting the trajectories on course, possibly with the use of a small fraction of GPS enabled nodes.

5. CONCLUSIONS

In this paper, we have outlined TBF – a new paradigm of forwarding in large scale, dense ad-hoc networks. Trajectory based forwarding has a number of features that are ideal for use in embedded networks, sensor networks, disposable networks, and networks that support the paradigm of ubiquitous or pervasive computing. We have presented some techniques for implementing trajectory-based forwarding and have shown the benefits of simple trajectories in implementing networking operations such as flooding, discovery and network management. It is our belief that this approach opens up a new area of research and can draw from the results of research efforts in various other disciplines. We have provided a basis for discussion of a number of research issues that needs to be addressed in the area of networking in an embedded world.

6. ACKNOWLEDGMENTS

We would like to thank members of DATAMAN lab for all their constructive comments.

7. REFERENCES

[1] M. Hamdi S. Capkun and J.P. Hubaux. Gps-free positioning in mobile ad-hoc networks. In *Hawaii International Conference On System Sciences*. HICSS-34, January 3-6 2001. Outrigger Wailea Resort.

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

[3] Dragoş Niculescu and Badri Nath. Ad hoc positioning system (APS). In *GLOBECOM*, November 2001. San Antonio.

[4] J. Li et. al. A scalable location service for geographic ad hoc routing. In *6th ACM MOBICOM*, August 2000. Boston, MA.

[5] D. B. Johnson. Mobile host internetworking using ip loose source routing. Technical Report CMU-CS-93-128, Carnegie Mellon University, February 1993.

[6] David B. Johnson and David A. Maltz. Dynamic source routing in ad hoc wireless networks. *Mobile Computing*, 353, 1996.

[7] G. Finn. Routing and addressing problems in large metropolitan-scale internetworks. Technical Report ISI Research Report ISI/RR-87-180, University of Southern California, March 1987.

[8] J. C. Navas and Tomasz Imielinski. Geographic addressing and routing. In *MobiCom'97*, September 26-30 1997. Budapest, Hungary.

[9] Ljubica Blazevic, Silvia Giordano, and Jean-Yves Le Boudec. Self organized terminode routing. In *Cluster Computing*, volume 5, pages 205–218, 2002.

[10] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. In *3rd International Workshop on Discrete Algorithms and methods for mobile computing and communications*, August 1999. Seattle, WA.

[11] B. Karp and H.T. Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. In *6th ACM MOBICOM*, August 2000. Boston, MA.

[12] Charles E. Perkins and Elizabeth M. Royer. Ad hoc on-demand distance vector routing. In *2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, February 1999. New Orleans, LA.

[13] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *6th ACM MOBICOM*, August 2000. Boston, MA.

[14] D. L. Tennenhouse and D. Wetherall. Towards an active network architecture. *Multimedia Computing and Networking*, January 1996. San Jose, CA.

[15] Deepak Ganesan, Ramesh Govindan, Scott Shenker, and Deborah Estrin. Highly resilient, energy efficient multipath routing in wireless sensor networks. In *Mobile Computing and Communications Review (MC2R)*, volume 1, 2002.

[16] Y.-B. Ko and N. H. Vaidya. Location-aided routing (lar) in mobile ad hoc networks. In *MobiCom'98*, October 1998.