3-D ROUTING ALGORITHMS IN MOBILE AD HOC NETWORKS

George Kao

A thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Computer Science

Concordia University

Montréal, Québec, Canada

April 2006

Your file Votre référence
ISBN: 0-494-14324-X
Our file Notre référence
ISBN: 0-494-14324-X

Abstract

3-D Routing Algorithms in Mobile Ad Hoc Networks

George Kao

A unit disk graph and its proximity graphs are often used as the underlying topologies of a mobile ad hoc network. One category of unicast routing algorithms, position-based routing algorithms, has been developed and studied extensively in the context of 2-dimensions (2-D). This, however, poses evident questions in terms of the reliability and efficiency of these algorithms when practically the mobile host is an object positioned in the real world of 3-dimensions (3-D).

Our first proposed 3-D routing algorithm, *Projective FACE* routing on two orthogonal planes, adopts a projective approach to adapt the 2-D *FACE* routing algorithm to a 3-D environment. This algorithm, shown by our simulation, performs significantly better in terms of delivery rate than the other deterministic routing algorithms such as 3-D *Greedy*, 3-D *Compass* routing algorithms.

The *Adaptive Least-Squares Projective FACE* routing algorithm, based on three heuristics, *least-squares projection (LSP) plane*, *adaptive behavior scale (ABS)* and *multi-projection-plane strategy*, which have strengthened the algorithm of *Projective FACE* routing on two orthogonal planes. The *Adaptive Least-Squares Projective*

FACE routing on 16 or more projection planes gives nearly certain delivery rate (100% of delivery rate shown by our simulation).

We provide a comparison of performance for routing in 2-D and in 3-D. Two versions, *half-space partition* and *quadrant-space partition*, for certain deterministic routing algorithms are proposed to accommodate the randomized routing in 3-D. We also study experimentally the properties of geometric graphs in 3-D and the performance of various 3-D routing algorithms on these graphs in this thesis.

Acknowledgments

I would like to thank my supervisors, Dr. Thomas Fevens and Dr. Jaroslav Opatrny. Their invaluable inputs to and guidance for my research are greatly appreciated.

For the years during my studies in Montréal, I am always grateful to the unwavering support from my family. Mom and Dad, I love you. A special thanks to my high school buddy in Vancouver, Wilson, for being a good friend and his encouragement.

Finally, I want to thank the peer support from Edward, Alaa, and Emad. I had fun with them during my study of Master.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Wireless Mobile Ad Hoc Network (MANET)

A wireless mobile ad hoc network consists of mobile hosts that communicate with each other without fixed infrastructure or centralized control. A mobile host usually operates as a router and is able to communicate with another mobile host if the the distance between them is within the minimum of their two direct transmission ranges. If a mobile host needs to send a message to another, the message usually has to be sent through some other intermediate hosts within the networks. Since the mobile hosts move, the underlying topology of the network may change. Hence, the network topology is generally dynamic. Routing efficiently in such network becomes a challenging task.

1.2 Routing in MANET

Mobile hosts are free to move without centralized control. This results in a dynamic network with potentially rapid topological changes. A good routing algorithm for such a network environment has to adapt to the changing network topology. The mobile hosts often use batteries which have a limited energy supply. Having power-efficient routing algorithms keep the overheads low so that the mobile hosts can last for longer periods of time. It is therefore necessary to take all these network properties into consideration when designing routing algorithms for MANETs.

Unicast is defined as the communication between a single sender and a single receiver over a network while multicast is defined as the communication between a single sender and multiple receivers. In this thesis, we consider only the unicast routing task in which a message is to be sent from a source to a destination host in a given wireless network.

A 3-D MANET can be represented by a *geometric undirected graph* $G = (V, E)$ where V is the set of points in the 3-D space representing the positions of the mobile hosts and E is the set of edges. An edge connects a pair of vertices if there is a bidirectional communication between the corresponding hosts. We define $d(u, v)$ as the Euclidean distance between the points u and v,

$$d(u,v) = \sqrt{(u_x - v_x)^2 + (u_y - v_y)^2 + (u_z - v_z)^2}. \tag{1}$$

We also define $S(p, r)$ as the sphere with center point p and radius r.

All mobile hosts in the network are commonly assumed to have the same transmission power. The maximum transmission rage for each mobile host is therefore at most R. Let the point u be any point in the unit disk graph (UDG). All other points that are inside the sphere $S(u, R)$ should connect to the point u. The set of edges E, representing the communication links, of the UDG satisfies $\{\{u, v\} : u, v \in V, d(u, v) \leq R\}$. In Figure 1, the sphere has the radius R so all other points inside the sphere should connect to the center point u.

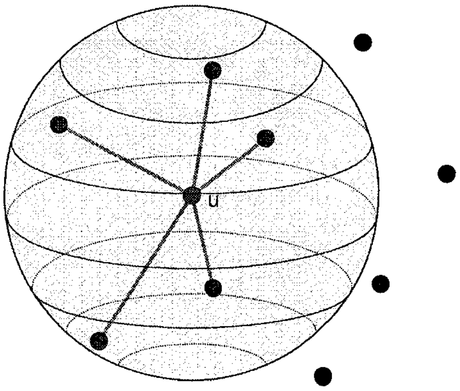


Figure 1: 3-D unit disk graph.

1.3 Our Motivations

MANETs are well-known to have many applications such as disaster relief operations, conferencing, and environment sensing. Many routing algorithms [GSB03] [MWH01] as well as several geometric graphs representing underlying topologies of a MANET have been proposed and studied extensively in the context of 2-D during the past few years. Since a mobile host is indeed an object positioned in the real world of 3-D, our goal is to extend these routing algorithms and geometric graphs to 3-D so that they can better represent real-world scenarios.

1.4 Thesis Overview

In this thesis, we first extend some well-known 2-D routing algorithms for MANET. We then propose several heuristics based on 2-D *FACE* routing [BMSU01] [KK00] by adapting the algorithm to the 3-D environment. Our first proposed algorithm, *Projective Face* routing algorithm on two orthogonal planes, by our simulation, gives significantly better delivery rate than the other deterministic routing algorithms such as *Greedy* [Fin87], *Compass* [KSU99], *Ellipsoid* [YS04] and *Most Forward* [TK84] routing algorithms. Our *Adaptive Least-Squares Projective FACE* routing algorithm on 16 or more planes not only gives nearly certain delivery rate (100% by our simulation) on UDG but also has improved routing path over *Projective Face* routing algorithm.

1.5 Thesis Organization

The rest of the thesis is organized as follows. Chapter 2 surveys the routing algorithms with an emphasis on position-based routing and the undirected geometric graphs that are adopted to model a MANET by researchers. In Chapter 3, we study the properties of the unit disk graph (UDG), Gabriel graph (GG) [GS69] and the relative neighborhood graph (RNG) [JT92] in 3-D as well as the performance of 3-D routing algorithms on these graphs. We also extend some existing localized position-based routing algorithms from 2-D to 3-D. We describe our heuristics based on the *FACE* routing algorithm and propose two novel routing algorithms, which are the *Projective FACE* and the *Adaptive Least-Squares Projective FACE* routing algorithms. We present our experimental results and discussions in Chapter 4 and finally conclude the thesis and list our interests for further research in Chapter 5.

Chapter 2

Background

Routing efficiently and reliably is one of the most significant key challenges in MANETs. This issue has been actively addressed by many research efforts in the last decade. In this chapter, we survey various categories of proposed unicast routing algorithms, in particular position-based routing algorithms and their desired characteristics. Since the underlying topologies of MANETs are usually represented by undirected geometric graphs, we also further discuss several proposed subgraphs and their properties.

2.1 Categories of Unicast Routing Algorithms in MANET

Fundamental routing issues can be well understood by having a strong development on unicast routing issues and techniques. We limit our discussion only on the unicast routing techniques without further extending to the multi-destination routing such as multicast.

Unicast routing algorithms are generally divided into two categories, topology-based and position-based routing algorithms [MWH01]. Topology-based algorithms use a routing table and the destination address to make the decision in forwarding messages or packets [MWH01]. The position-based routing algorithms rely on location information of the mobile hosts by using the technologies such as Global Position System (GPS) [GSB03].

2.1.1 Desirable Characteristics

The desirable qualitative characteristics of routing algorithms to evaluate the performance of routing algorithms are listed and elaborated below.

Localized Method

Localized algorithms are distributed algorithms where simple local behavior or information is sufficient to achieve the global objective. As for a global or centralized position-based routing algorithm, each mobile host has to know the location of every other host in the network. If such global information is available, this becomes equivalent to problem of finding the shortest path. Some well-known routing algorithms for finding the shortest path are Dijkstra algorithm [Dij59] and Bellman-Ford algorithm [FF62]. In a localized position-based routing algorithm, each mobile host makes routing decisions solely based on the location of itself, its neighbors (usually one-hop), and the destination hosts. The localized routing algorithms are certainly more preferable if they can approximately match the performance of global routing algorithms.

Memoryless Operation

The memory size in mobile hosts is susceptible to the memorization of the path routes. If the network size increases, the path traversed by the routing algorithms increases its length. This results in the need for expanding the size of memory for the mobile hosts. Thus, it is better to avoid memorizing the past routes at any node during the routing process.

Shortest Path

Most routing algorithms use *hop count* as the constant metric if all mobile hosts have the same transmission power. These hosts have fixed transmission power and may not be able to adjust their transmission radii to reach certain desired neighbors. The *hop count* is defined as the number of transmissions on a path that is required by the algorithm to traverse from a source to a destination host. The objective of the routing algorithms is to minimize the *hop count* so it is nearly that for the shortest path, which is defined as the minimal number of hops between the source and the destination. It is also assumed that the delay is proportionally related to the *hop count.*

Single Path Strategy

In the flooding-based strategy, a message is flooded through the whole or portion of network. This causes high communication overheads if the network size is relatively large. The single path strategy is desirable since only one copy of the message is in the network at any time.

Guaranteed Delivery

Delivery rate is defined as the percentage of successful deliveries to the destination host. A principal objective of designing routing algorithms is to guarantee the message delivery.

Loop freedom

The designed algorithm should have the property of loop freedom to avoid the need to memorize the past path for loop detection.

Scalability

The performance of well-designed routing algorithms should adapt well to various sizes of the network. In general, if the algorithms are localized, memoryless, and single-path, they are usually scalable.

2.2 Topology-Based Unicast Routing Algorithms

The topology-based algorithms can be further divided into three types [MWH01].

Proactive Routing Proactive routing algorithms employ classical routing approaches such as distance-vector routing [PB94] or link-state routing [CJL+01]. Unicast routes between all pairs of hosts are retained regardless of the usage of these routes. The high communication overheads make these algorithms inappropriate for networks that have limited resources.

Reactive Routing Reactive routing, also known as on-demand routing algorithms, maintain only the needed routes. Although the cost of maintaining the routes is reduced, the messages or data packets to be delivered usually experience queuing delays at the source while the route is being found at session initiation or the route is being repaired later on after a failure.

Hybrid Routing Based upon the proactive and reactive routing algorithms, some hybrid routing algorithms are proposed. Zone routing protocol (ZRP) [PH97] is one well-known protocol that falls into this type.

2.3 Position-Based Unicast Routing Algorithms

Even though topological-based routing algorithms have been proposed for several years, they require high communication overheads in MANET. The availability of small GPS receivers with low-power consumption for finding the relative coordinates of mobile hosts draws great attention from researchers to develop the position-based routing algorithms for MANET. The assumption for this category of algorithms is that every mobile host knows the position of itself, its neighbors (usually one-hop neighbors), and the destination. Every mobile host learns locations of its one-hop neighbors by exchanging messages between neighbors. The location of the potential destination host can be possibly obtained through a location service such as DREAM system [PB94], Quorum system [HL99], and Grid system [LJD$^+$00]. Position-based routing algorithms do not require any explicit route discovery or route maintenance mechanisms unlike the topology-based algorithms. These routing algorithms therefore can potentially be more efficient when the underlying network topology changes frequently.

2.3.1 Flooding-Based Routing Algorithms

One advantage of the flooding-based routing algorithms that use positions of the mobile hosts is their simplicity and these algorithms may be more efficient than the other algorithms in terms of message delivery while the rate of data transmission is low or the network is relatively small. In the location aided routing (LAR) algorithm [KV98], the location information is used to limit the scope of route request zone. Only the nodes within the request zone forward route requests. If the route discovery using the smaller request zone fails to find the route to the destination, the sender initiates another route discovery with a larger request zone. The flooding-based algorithms have higher reliability in terms of delivery rate since the messages may be delivered to the destination on multiple paths. However, the drawback is that high communication overheads are potentially incurred.

2.3.2 Deterministic Routing Algorithms

A deterministic algorithm has behavior that can be completely predicted from the input. On the contrary, a randomized algorithm relies on an internal random number generator to make decisions. Several localized deterministic routing algorithms were developed for routing on 2-D networks. The *Greedy* routing scheme was proposed by Finn [Fin87]. The *Greedy* routing algorithm selects the neighboring node that is the closest to the destination node. In the two-hop *Greedy* routing algorithm [SL01], the neighboring node which is the closest to the destination is selected from both one-hop and two-hop neighbors. A variant of *Greedy* routing algorithm, *Ellipsoid* routing algorithm [YS04], is proposed such that the sum of the Euclidean distances from the current to the selected neighboring node and from the current to the destination node is minimized. In [TK84], the authors proposed *Most Forward* routing algorithm, in which the neighboring node that gives the greatest progress with respect to its projection on the line formed by the current and the destination nodes is selected. In the *Compass* routing algorithm [KSU99], the angle formed by the selected neighboring node, the current node and the destination node is minimized.

Since these routing algorithms fail to deliver the packet in some situations, the *FACE* routing [BMSU01] [KK00] was proposed to achieve the guaranteed delivery under the circumstance that the underlying network topology is a geometric planar graph, the edges of which only intersect at their common end-vertices (no crossing edges). The principal idea of this algorithm is to route along the edges of the faces that lie on the line formed by the source and the destination nodes. Figure 2 demonstrates the *FACE* routing algorithm on a planar graph. The algorithm is as follows.

Algorithm *2-D FACE routing*

Input: The source node, s, the destination node, t and a 2-D geometric planar graph.

```
1. $p \longleftarrow s$.
2. Repeat step 3 to 5.
3. Let $f$ be the face with a point $p$ on its boundary that intersects the line segment
   between $p$ and $t$.
4. Traverse $f$ until reaching an edge that intersects $\overline{pt}$ at some other point $p' \neq p$.
5. $p \longleftarrow p'$
6. Until $t$ is reached.
```

The authors [BFNO03] consider the instability in the transmission ranges of the mobile nodes possibly caused by the natural or man-made obstacles. This robust *FACE* routing algorithm can still guarantee the delivery if the ratio of the maximum transmission range to the minimum transmission rage is at most $\sqrt{2}$. In [ANO05], a generalization of *FACE* routing that can guarantee delivery in some non-planar graphs is considered. The proposed algorithm requires some memory to memorize partial paths being traversed. To deal with the drawback of *FACE* routing algorithm, where the route discovered is usually longer than the path found using other deterministic routing algorithms, the Other Adaptive Face Routing (AFR) [KWZ02] is proposed to avoid long paths by using a bounding region.

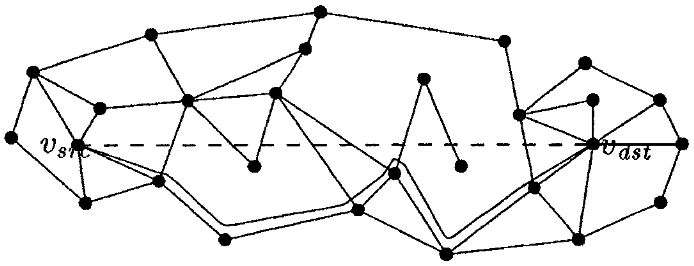


Figure 2: The *FACE* routing algorithm on a planar graph from [BMSU01], where v_{src} is the source and v_{dst} is the destination.

2.3.3 Randomized Routing Algorithms

In randomized routing algorithms, the current node randomly selects one of its neighboring nodes, usually from a subset of all its neighboring nodes, to forward the message or the data packet. The randomized heuristic increases the delivery rate of the routing algorithm. However, the trade-off is that the length of the path traversed is increased. In the random progress algorithm [NK84], a node, u, randomly selects the neighboring node with equal probability from a set of selected neighboring nodes. Assuming that there are k neighboring nodes, the node, u, will select one neighboring node with probability $\frac{1}{k}$. Fevens et al. [FHN04] proposed a class of randomized algorithms called AB (Above-Below) algorithms for position-based routing. The AB algorithms selects one neighboring node of the node, u, above the line formed by the node, u, and the destination node, d, and the other neighboring node below that line.

One of these two neighboring nodes are randomly selected for routing according to some probability distribution. The selection of neighboring nodes and the probability distribution determine the specific algorithm.

2.4 Undirected Geometric Graphs

A MANET is often modeled by a unit disk graph, UDG. There is an edge between two nodes if and only if their Euclidean distance is within a given transmission radius, R, which is usually assumed to be the same for all nodes. All nodes are assumed to have the same R. The number of edges of the UDG could be as large as the square order of the number of nodes.

2.4.1 Desired Features

In the routing algorithms that adopt the flooding strategy, a mobile host forwards a packet to all its neighbors in the network to discover a path, which potentially incurs high communication overhead. One method to reduce such overhead is to allow each mobile host to communicate only with a selected subset of the neighboring mobile hosts. This approach can be seen as retaining geometric spanning subgraphs.

Several geometric spanning subgraphs are proposed to restrict the size of the network. The properties of these spanning subgraphs are found to be extremely important for the routing algorithms. Typically, they have the following properties.

Sparseness Many routing algorithms benefit from the decrease in the number of edges of a connected graph, achieved by extracting its spanning subgraph, while being able to maintain the connectivity. This property enables these routing algorithms, particularly the flooding-based algorithms, to run on a topology in a more efficient manner.

Geometric Planarity A 2-D geometric planar graph is a graph in which no crossing edges exist or the edges of which only intersect at their common end-vertices. This property is required for some routing algorithms such as 2-D *FACE* routing algorithm to ensure the performance of guaranteed delivery.

Localized Construction The localized construction of a spanning subgraph only needs the information of neighboring nodes for each node. Some localized routing algorithms require a planar underlying topology which can also be constructed and maintained in a localized manner so that these routing algorithms are still able to remain so-called localized algorithms.

***t*-spanner** Let G' be a t-spanner of G. G' is a spanning subgraph in which every two vertices are at most t times as far apart on G' than on G. A t-spanner is known to be power efficient for routing.

Some geometric planar subgraphs that can be constructed using only local information are proposed. Gabriel graph (GG) and relative neighborhood graph (RNG) are two well-known spanning subgraphs that are often adopted to extract the planar graph locally in 2-D. If $\{u, v\}$ is an edge in the UDG, only the one-hop neighbors of either the point u or v are required to be tested if the edge $\{u, v\}$ should be removed.

	GG	RNG
Lower Bound	$\Omega(n^{1/2})$	$\Omega(n)$
Upper Bound	$O(n^{1/2})$	$O(n)$

Table 1: Spanning ratio of GG and RNG in 2-D from [BDEK02].

As long as the UDG is a connected graph, the connectivity of Gabriel graph and relative neighborhood graph are also preserved. Bose et al. [BDEK02] study the spanning ratio of these two graphs in 2-D. Table 1 shows the upper and lower bounds asymptotically. They proved that the spanning ratio of Gabriel graph is precisely $\frac{4\pi\sqrt{2n-4}}{3}$ and that of relative neighborhood graph is precisely $n-1$.

2.4.2 Gabriel Graph (GG)

Let q be the mid-point of an edge $\{u,v\}$. The edge $\{u,v\}$ exists between the points u and v if no other point w from $V-\{u,v\}$ is present inside the sphere $S(q, \frac{d(u,v)}{2})$. The set of edges E of GG satisfies $\{\{u,v\} : u, v \in V, d(u,v) \leq min_{w \in V-\{u,v\}}\{\sqrt{d^2(u,w)+d^2(v,w)}\}\}$.

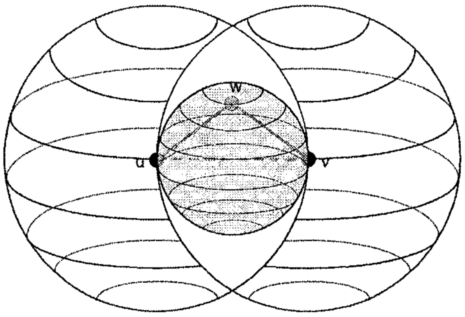


Figure 3: 3-D Gabriel graph. Since the point w is inside the shaded sphere, the edge $\{u,v\}$ is removed.

2.4.3 Relative Neighborhood Graph (RNG)

An edge $\{u, v\}$ exists between the points u and v if no other point w is present inside the lens region formed by the intersection of the two spheres, $S_u(u, d(u, v))$ and $S_v(v, d(u, v))$. The set of edges E of RNG satisfies $\{\{u, v\} : u, v \in V, d(u, v) \leq max_{w \in V - \{u,v\}}\{d(u, w), d(v, w)\}\}$.

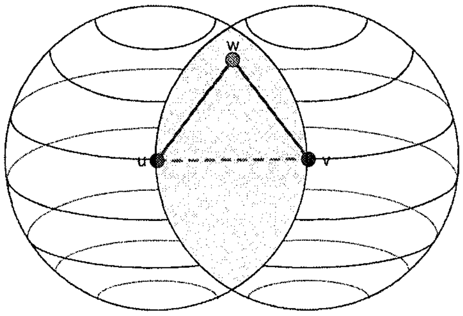


Figure 4: 3-D relative neighborhood graph. Since the point w is inside the shaded lens, the edge $\{u, v\}$ is removed.

2.4.4 Other Subgraphs

Yao graph [Yao77], YG_k, is also used as the underlying topology of the network since it is proved to be t-spanner. At each node u, any k equal-separated rays originated at u define k cones. In each cone, choose the closest node v to u with distance at most one, if there is any, and add an edge between u and v. It is showed that t is at most $\frac{1}{1-2\sin\frac{\pi}{k}}$. Yao graph is, however, not guaranteed to be a planar graph. Li et al. proposed a novel structure called Planarized Local Delaunay Triangulation (PLDel) [LCW02]. This structure can be locally constructed. They proved that PLDel is planar and a t-spanner, where t is $\frac{4\sqrt{3}}{9}\pi$. Restricted Delaunay Triangulation (RDT)

[GGH$^+$01b] is another subgraph that can be computed in a fully distributed manner. This structure is constructed using the hierarchical clustering algorithm [GGH$^+$01a]. This graph is a planar graph and proved to be a t-spanner. Recently, Chavez et al. proposed a distributed Half-Space Proximal (HSP) test [ECU05] that gives a $(2\pi + 1)$-spanner of a UDG. Unlike Yao graph which is rotation variant, this graph is rotation invariant. The authors also show that the graph is strongly connected and has out-degree at most six.

We select two spanning subgraphs, Gabriel graph (GG) and relative Neighborhood graph (RNG), of unit disk graph (UDG) to study the 3-D routing due to them simplicity of extending them to 3-D and their desired property of being able to extract geometric planar graphs from 2-D graphs.

Chapter 3

Routing on 3-D Geometric Graphs

3.1 Localized Position-Based Routing Algorithms in 3-D

Localized position-based routing algorithms are distributed algorithms. In localized position-based routing, every host in the network is assumed to know its location and its one-hop neighbors' locations and the destination's location. Therefore, each host makes the routing decision solely based on the location information of itself, its neighbors, the source and the destination. We further assume that the hosts are static while the routing is in progress. This is very reasonable assumption since the time it takes for routing is relatively short compared to the mobility of the hosts. Let u be the current node, $(v_1, ..., v_n)$ be the one-hop neighboring nodes of u, s be the source node and t be the destination node. The hop counts of the path discovered by the algorithm between the nodes s and t is denoted by $N_L(s,t)$. The hop counts of the shortest path between the nodes s and t is denoted by $N_D(s,t)$. We define the hop stretch factor as

$$HSF(s,t) = \frac{N_L(s,t)}{N_D(s,t)}. \tag{2}$$

We now extend four well-known localized routing algorithms to 3-D and use them for a comparison with our proposed *Projective FACE* routing algorithms. These localized position-based routing algorithms require only the location of a current node, its neighboring nodes, and the destination node for the current node to make routing decisions.

3.1.1 3-D Compass Routing

The current node u selects its neighboring node that forms the smallest angle, $min\{\angle v_1ut, ..., \angle v_nut\}$, together with the destination node t for forwarding the message or the data packet.

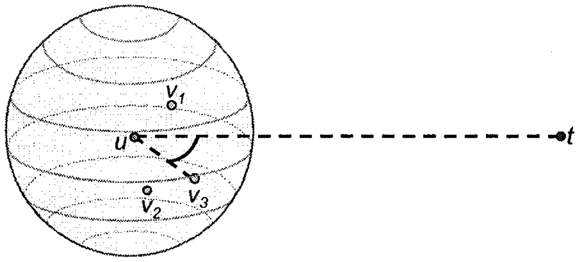


Figure 5: 3-D compass routing.

3.1.2 3-D Greedy Routing

The current node u selects its neighboring node that is the closest, $min\{d(v_1,t), ..., d(v_n,t)\}$, to the destination node t for forwarding the message or the data packet.

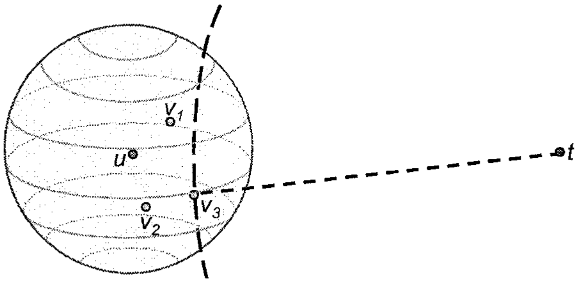


Figure 6: 3-D greedy routing.

3.1.3 3-D Ellipsoid Routing [YS04]

The current node u selects its neighboring node that gives the smallest sum of distances, $min\{d(v_1,u)+d(v_1,t),...,d(v_n,u)+d(v_n,t)\}$, from itself to the neighboring node and then to the destination node t for forwarding the message or the data packet.

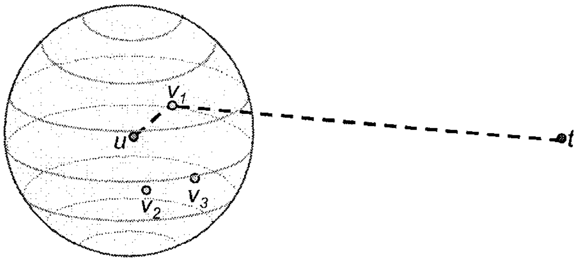


Figure 7: 3-D ellipsoid routing.

3.1.4 3-D Most Forward Routing

Let $(v_1',...,v_n')$ be the nodes perpendicularly projected on the line ut respectively. The current node u selects its neighboring node whose projected node is the closest, $min\{d(v_1',t),...,d(v_n',t)\}$, to the destination node t for forwarding the message or the data packet.

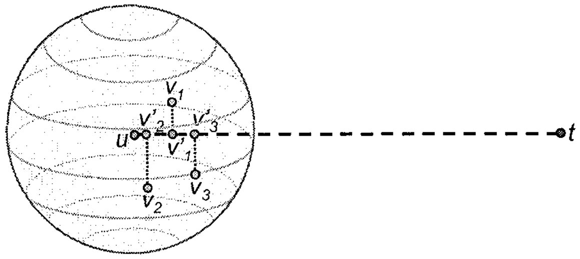


Figure 8: 3-D most forward routing.

3.2 *Projective FACE* Routing Algorithms

3.2.1 The Problem of 2-D Face Routing Algorithm on 3-D Geometric Graphs

FACE routing, by using the right-hand rule, guarantees the delivery on a 2-D geometric planar graph. The line $\overline{st}$ that connects the source and destination nodes determines the 2-D faces to be traversed as shown in Figure 9. However, this line does not determine these faces in a 3-D geometric graph. The *FACE* routing algorithm is thus not directly applicable on a 3-D graph.

We propose several heuristics based on the projective approach to deal with the problem described above. Since a geometric planar subgraph such as GG or RNG cannot be extracted from the projected graph by using only the local information before projection (see Figure 10), the delivery cannot be guaranteed while routing on this projected graph. Figure 10 shows a 3-D geometric graph being projected onto a plane. While the neighboring nodes of each node are preserved after the projection, the projected graph on the plane gives the crossing edges that cannot be eliminated by using only local information. The projected graph is thus a non-planar geometric graph with crossing edges that may potentially cause looping for *FACE* routing algorithms. The subgraphs, GG and RNG, that we adopt to evaluate our routing algorithms are computed from the 3-D UDG before the projection. The number of edges is reduced by extracting these subgraphs and the number of crossing edges of these projected subgraphs is reduced as well. As a consequence, our simulation results have reflected the better performance on these projected subgraphs by showing the increase in the delivery rate and the decrease in the hop stretch factor.

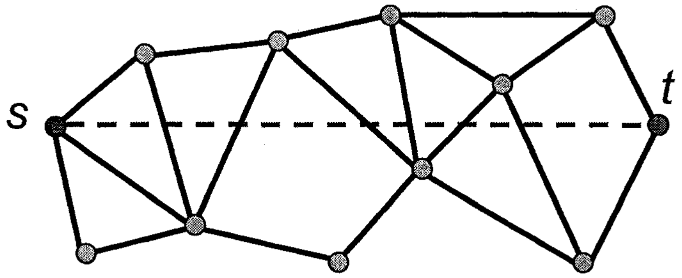


Figure 9: The line $\overline{st}$ determines the 2-D faces to be traversed.

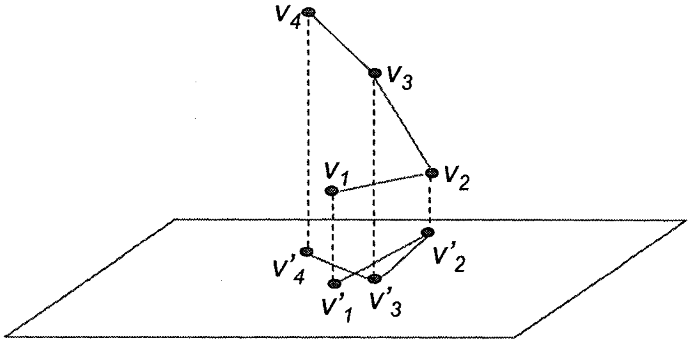


Figure 10: The 3-D nodes are projected onto a plane. The neighboring nodes of each node are preserved after projection.

3.2.2 Proposed Heuristics for Adapting to 3-D Environments

Projective FACE Routing on Two Orthogonal Planes

A plane is uniquely determined by three points not lying on a line. We compute the first projected plane by using the source, destination and an arbitrary third (other than the source and the destination) point such as a neighboring node of the source node or simply the point of the origin at $(0, 0, 0)$ as long as this point does not lie on the line formed by the source and destination nodes. Our proposed *Projective FACE* routing algorithm on two orthogonal planes is as follows. The points are first projected perpendicularly onto the first projected plane that contains the line $\overline{st}$. The *Face* routing algorithm is performed on this projected graph. If the routing fails, the points are then projected onto the second plane, which is orthogonal to the first plane and also contains the line $\overline{st}$. The *Face routing* is again performed. Although this heuristic does not guarantee delivery, our experiments show that the delivery rate is significantly better than other 3-D deterministic routing algorithms, such as 3-D *Greedy* or 3-D *Compass* routing algorithms. Figure 11 shows two planes that have the dihedral angle of $\frac{\pi}{2}$ degrees. The line of intersection between two planes passes through the source and the destination.

Least-Squares Projection Plane

In the *Projective FACE* routing algorithm, a third point is chosen arbitrarily, together with the source and the destination points, to compute the first projection plane. Instead of choosing a third point randomly, we adopt the mathematical optimization

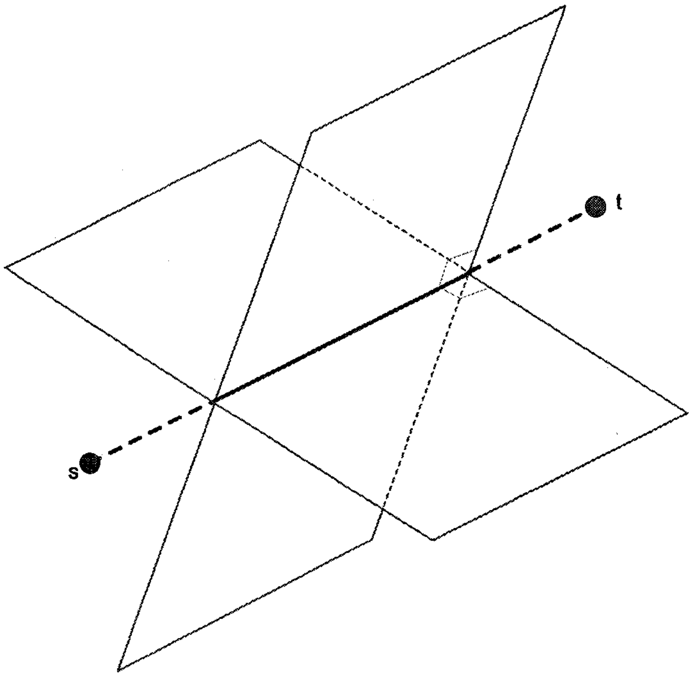


Figure 11: Two projection planes with dihedral angle of $\frac{\pi}{2}$ degrees.

technique that attempts to find the best fitting plane to a given set of data points. A plane with normal vector $\boldsymbol{n} = (a, b, c)$ through the point $p_0 = (x_0, y_0, z_0)$ has the equation, $\boldsymbol{n} \cdot (p - p_0) = 0$ or $a(x - x_0) + b(y - y_0) + c(z - z_0) = 0$. Assume there are m data points. The point p_0 of the least-squares plane is the centroid of these m data points, which is $\frac{1}{m}(\sum_{i=1}^{m} x_i, \sum_{i=1}^{m} y_i, \sum_{i=1}^{m} z_i)$. We define the ordinate difference (residuals), r_i, as the Euclidean distance from a data point i to its perpendicularly projected point on the least-squares plane. The sum of the squares of the ordinate differences between the fitted function of the least-squares plane and m data points has to be minimized. That is to say $\sum_{i=1}^{m}(r_i)^2$ is minimized. By applying the technique of singular value decomposition (SVD) [Sha98], the normal vector of the least-squares plane can be calculated, which with p_0 uniquely defines the plane.

To maintain the localized characteristic of the routing algorithm, we propose that only the source, destination and the neighboring nodes within the 2-hop distance of the current node are selected as the set of data points for computing the least-squares projection plane. The heuristic is aimed to have a less distorted projected graph so that the number of crossing edges can potentially be reduced.

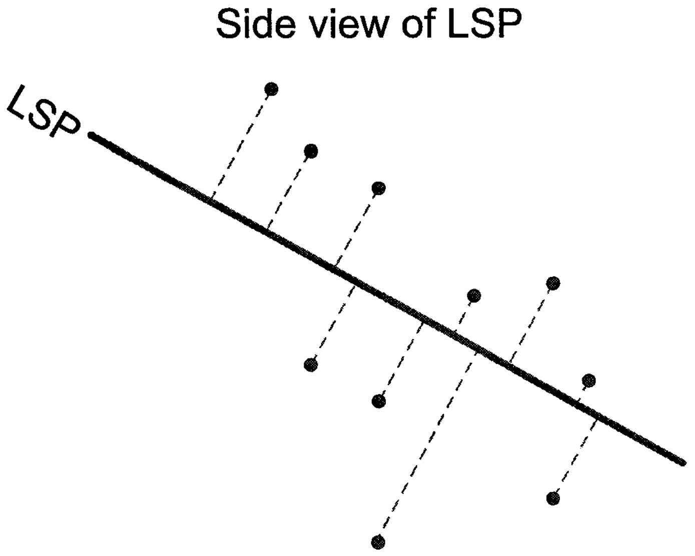


Figure 12: Least-squares projection plane as the first projection plane.

Adaptive Behavior Scale (ABS): An Adaptive Routing Approach

The current node can always be seen as the source node during the routing process since it is the node that is currently holding the message or the data packet to be delivered to the destination. We define a parameter called *Adaptive Behavior Scale (ABS)*. The *ABS* is used to determine when to recalculate the least-squares projection plane after a certain number of hops being traversed if the destination has not been successfully reached. This heuristic makes the *Projective FACE* routing algorithm more dynamic and robust as the looping caused by the crossing edges is possibly avoided after the recalculation of the least-squares projection plane. Our simulation results show that this heuristic leads to significant amelioration in terms of the hop stretch factor.

Multi-Projection-Plane Strategy: Increasing the Number of Available Projection Planes

By performing the *FACE* routing on the additional second plane that is orthogonal to the first plane, a significant increase in the delivery rate has been observed from our simulation results. We are interested in studying the effect on the delivery rate if we increase the number of the available projected planes. All projected planes have a common line of intersection. If we make a cross-section that is perpendicular to all the planes and look along their intersection, the dihedral angle between each pair of neighboring planes are identical. Let Ns be the number of planes. The dihedral angle between each pair of neighboring planes is thus $\frac{\Pi}{Ns}$ degrees. Figure 13 shows the projected view of eight planes.

3.2.3 Nearly Certain Delivery of *Adaptive Least-Squares Projective FACE* Routing Algorithm

We hybridize the three proposed heuristics, the *adaptive behavior scale*, the *least-squares projection plane* and the *multi-projection-plane strategy*, to develop our *Adaptive Least-Squares Projective FACE* routing algorithm. This algorithm with 32 projection planes available gives nearly certain delivery (100% shown by our simulation).

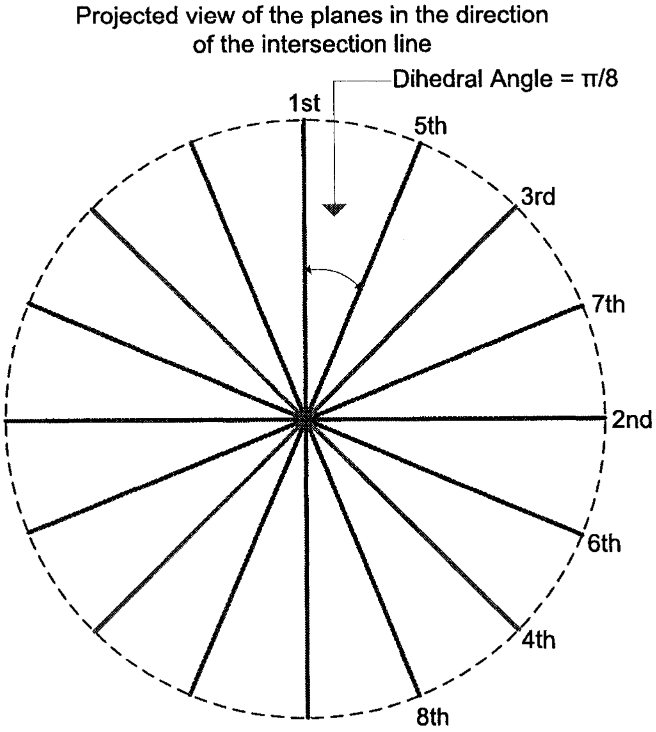

Projected view of the planes in the direction of the intersection line
Dihedral Angle = π/8
1st
5th
3rd
7th
2nd
6th
4th
8th

Figure 13: An example of the multi-projection-plane strategy (8 projection planes).

Algorithm Adaptive Least-Squares Projective FACE *routing*

Input: s, t and a 3-D geometric graph

1. $c \longleftarrow s$.

2. Compute the least-squares projection plane using the positions of c, t, and the 1-hop and 2-hop neighbors of c.

3. Compute the projected graph perpendicularly projected on the least-squares projection plane.

4. Perform 2-D *FACE* routing until t is reached, the pre-set ABS is reached or the

pre-set threshold value is reached.

5. If t is reached, exit. If the pre-set ABS is reached, go to step 2. If the pre-set threshold value is reached, continue to next step.
6. Compute the next projection plane if the pre-set number of projection planes is not reached else exit.
7. Compute the projected graph perpendicularly projected on the plane.
8. Perform 2-D *FACE* routing until t is reached or the pre-set threshold value is reached.
9. If t is reached, exit. If the pre-set threshold value is reached, go to step 6.

The heuristic of multi-projection-plane strategy strengthens our algorithm in terms of delivery rate and encourages the difficult routing cases, which fails to deliver using the two orthogonal planes, to succeed in delivery. The change of the projection plane results in the change of the geometric projected graph. The algorithm eventually chooses different faces to perform the *Face* routing algorithm.

3.3 Randomized Routing in 3-D

Randomized algorithms are considered in this paper since they are relatively simple algorithms for improving the performance of deterministic algorithms. Two randomized versions of certain deterministic routing algorithms to adapt to the 3-D environment are also proposed.

The phases of these two randomized versions of deterministic routing algorithms are described in detail below. They are applied at the current node during the routing process. In principle, one neighboring node is randomly chosen to forward the message or data packet according to the type of partition and the availability of the nodes.

3.3.1 Quadrant-Space Partition

The 3-D randomized algorithm based on the version of quadrant-space partition has the following phases.

1. One plane is first defined by using the current, destination and a third arbitrary point such as $(0, 0, 0)$. Another plane that also contains the current and the destination nodes is defined to be perpendicular to the first plane so that these two planes partition the space into four quadrants.

2. If there is at least one node in one of the four quadrant spaces, one node is selected deterministically, based on the chosen routing algorithm. For example, suppose the *Greedy* routing algorithm is used with this randomized version. In each of the four quadrant spaces separated by the two orthogonal planes defined in phase 1, one node from the subset of the neighboring nodes is selected if that node is closest to the destination. In the case that there is no node present in the quadrant space, no node is selected from this quadrant space.

3. One final node out of those nodes selected from phase 2 is randomly chosen as the next node to forward the message.

3.3.2 Half-Space Partition

The 3-D randomized algorithm based on the version of half-space partition has the following phases.

1. We define one plane that contains the line formed by the current and destination nodes. This plane bisects the space into two half spaces so that each half space contains an equal number of neighboring nodes.

2. One node is selected deterministically, based on the chosen routing algorithm, from each of the two half space defined by the two planes. If the current node has at least one neighboring node, one from each half space is selected in this phase. For example, suppose the *Greedy* routing algorithm is used with this randomized version, in each of the two half spaces separated by the plane defined in phase 1, one node from the subset of the neighboring nodes is selected if that node is closest to the destination.

3. One final node out of those nodes selected from phase 2 is randomly chosen as the next node to forward the message.

Chapter 4

Empirical Results

4.1 Simulation Environment

We conduct experiments as follows. There are 75 nodes uniformly and randomly generated in a cube of side length 100. The transmission radius of each host is set to a fixed value. We first calculate all connected components in the graph so that we can identify the number of maximal connected subgraphs. We select the largest connected component (LCC) among all the connected components to evaluate the performance of the routing algorithms. The source and destination nodes are then randomly picked from the LCC. The statistics are obtained from the average of 10,000 runs. In each run, various 3-D routing algorithms are performed on the UDG, GG and RNG. The same simulation setting is conducted for five different maximum transmission radii, which are 15, 20, 25, 30 and 35.

4.2 Properties of 3-D Geometric Graphs

GG and RNG are the spanning subgraphs computed from UDG so that they both contain all the nodes of UDG. Thus, UDG, GG and RNG also have the same number of nodes in their LCCs. Figure 14 shows the average number of nodes in the LCC for different radii. If the radius is set to 30, the average number of nodes in the LCC is very close to the total number of nodes, 75, in the entire graph.

Figure 15 shows the average number of edges in the LCC of each graph for different radii. As expected, the average number of edges increases as the radius increases.

In Figures 16 and 17, we study the distribution of nodes in terms of the node degree. For the radius of 25, Figure 16 shows the average percentages of nodes with various degrees of the nodes in the LCC of each graph (only the nodes in the LCC are considered). Figure 17 shows the average percentages of nodes with various degrees of the nodes in the LCC of UDG for different radii. It is clearly seen that each node in the LCC is of at least degree 1.

4.3 Performance of Position-Based Routing Algorithms in 3-D

4.3.1 Performance of *Projective FACE* Routing on Two Orthogonal Planes

We study the performance of our first *Projective FACE* routing algorithm on two orthogonal planes and compare its performance with other 3-D deterministic routing algorithms (*Greedy*, *Compass*, *Ellipsoid* and *Most Forward* routing algorithms). We compare the performance of these five routing algorithms for different radii in Figure 18 and Figure 19.

Figure 18 shows the delivery rate, given that the underlying network topology is UDG. For the radius of 25, the *Projective FACE* routing algorithm performs significantly better than the other routing algorithms. Since the projected graphs on which the *Projective FACE* routing algorithm performs are not necessarily planar graphs, we use a threshold value to terminate the routing process if the number of hops traversed exceeds $2N$ (150 hops), where N is the number of nodes. Interestingly, we also found that the curve, for the delivery rate, of each routing algorithm for different radii is U-shaped (a parabola that opens upward). When the radius is small, the number of nodes in the LCC is small. The delivery rate decreases as the number of nodes in the LCC becomes larger. When the radius is 25, the number of nodes in the LCC almost reaches 80% of that of the entire graph. When we continue to increase the radius, the number of nodes in the LCC is nearly the same as the total number of nodes in the entire graph. However, the number of edges still increases (the average node degree increases) and this results in the increase of the delivery rate.

Figure 19 shows the hop stretch factor. The hop stretch factor is close to 1 for the *Compass*, *Greedy*, *Ellipsoid* and *Most Forward* routing algorithms even if the radius is set to different values. Therefore, the routing path traversed using these four algorithms is almost the same as the shortest path.

When the radius is set to 25, the delivery rate of *Projective FACE* routing on two orthogonal planes remains to be greater than 90%, which outperforms the other four routing algorithms. The drawback is, however, the huge stretch factor. In Figure 20 and Figure 21, the radius is set to 25. We compare the performance of the five routing algorithms on the three graphs, which are UDG, GG and RNG. Due to the fact that the RNG has the least number of edges among the three graphs, *Greedy*, *Compass*, *Ellipsoid* and *Most Forward* algorithms have worse delivery rate on that graph. These four routing algorithms also need to route longer paths to reach the destination, which results in slightly higher hop stretch factors. On the contrary, the *Projective FACE* routing on two orthogonal planes gives the lowest hop stretch factor on RNG. Since the number of edges are reduced for 3-D RNG, the crossing-edges on the projected graphs are potentially reduced.

4.3.2 Effect of Threshold Value

We are interested in seeing the effect on the performance of our *Projective FACE* routing on two planes. In Figure 22, the delivery rate is shown when we increase the threshold value. The increase in the delivery rate is very insignificant after the threshold value of $4N$, where N is the number of nodes (75 nodes in our cases). Figure 23 shows the corresponding hop stretch factor. The hop stretch factor continues to linearly increase even though the delivery rate stops increasing. This indicates that the looping occurs during the routing process.

4.3.3 Effect of Adaptive Behavior Scale (ABS)

Two more heuristics, least-squares projection (LSP) plane and adaptive routing approach, are introduced to our original *Projective FACE* routing algorithm. We name this algorithm the *Adaptive Least-Squares Projective FACE* routing algorithm. We initially set the ABS to N hops. The LSP plane is re-calculated once the number of hops traversed reaches the pre-set ABS. Figure 24 shows the delivery rate of the *Adaptive Least-Squares Projective FACE* routing algorithm on two orthogonal planes while the threshold value increases. The hop stretch factor has been improved, with our adaptive routing approach, compared to that of *Projective FACE* routing algorithm. The hop stretch factor stays relatively low even if the threshold value is large as shown in Figure 25. The delivery rate stops increasing after the threshold value of $12N$.

Algorithms	Number of Projection Planes Available	Dihedral Angle
Projective FACE	2	$\frac{\pi}{2}$
Projective4 FACE	4	$\frac{\pi}{4}$
Projective8 FACE	8	$\frac{\pi}{8}$
Projective16 FACE	16	$\frac{\pi}{16}$
Projective32 FACE	32	$\frac{\pi}{32}$
Adap. LS-Proj. FACE	2	$\frac{\pi}{2}$
Adap. LS-Proj4. FACE	4	$\frac{\pi}{4}$
Adap. LS-Proj8. FACE	8	$\frac{\pi}{8}$
Adap. LS-Proj16. FACE	18	$\frac{\pi}{16}$
Adap. LS-Proj32. FACE	32	$\frac{\pi}{32}$

Table 2: Various projective-based routing algorithms.

We are interested in finding the appropriate ABS that allows the *Adaptive Least-Squares Projective* routing algorithm to have optimal delivery rate. We fix the threshold value to $12N$ that is observed to give the best possible delivery rate from Figure 24. The ABS is varied and the delivery rate of *Adaptive Least-Squares Projective FACE* routing algorithm is plotted in Figure 26. The delivery rate gets increased by another 2% when we increase the ABS to 125 hops. The hop stretch factor is also improved as shown in Figure 27. Therefore, we will use 125 as the ABS value to conduct our various *Adaptive Least-Squares Projective FACE* routing algorithms.

4.3.4 *Projective FACE* vs *Adaptive Least-Squares Projective FACE* Routing

Our *Adaptive Least-Squares Projective FACE* routing algorithm nearly always gives 100% delivery rate if we have 16 or more projection planes available on UDG. Refer to Table 2 for the explanations of the names of the algorithms in the following figures.

Figure 28 shows the delivery rate of both *Projective FACE* and *Adaptive Least-Squares Projective FACE* routing algorithms with various numbers of projection planes available. The *Adaptive Least-Squares Projective FACE* routing algorithm with 32 planes gives 100% of delivery rate on all three graphs. The *Projective FACE* routing algorithm with 32 planes available achieves the delivery rate that is very close to 100%. However, the *Adaptive Least-Squares Projective FACE* routing algorithms give relatively better hop stretch factors as shown in Figure 29.

Figure 30 and 31 give the performance of the algorithms for different radii on UDG. When the radius is set to 25, the worst delivery rate is given by all the 3-D routing algorithms as observed in our previous simulation results. Since *Adaptive Least-Squares Projective FACE* routing algorithm with 16 or more planes gives 100% of delivery rate for the radius of 25, the delivery rates for other radii are also 100% as expected.

4.4 2-D Routing vs 3-D Routing

We compare the performance of the routing algorithms in 2-D and 3-D.

4.4.1 Deterministic Algorithms

We consider four deterministic algorithms (*Compass*, *Greedy*, *Ellipsoid* and *Most Forward* routing algorithms) and compare their performance in 2-D and 3-D. Figure 32 shows the delivery rate on UDG. It is observed that the curves for the 3-D routing algorithms are shifted to the right with respect to the increase of the radius. *Ellipsoid* routing algorithms performs the worst in terms of delivery rate among the four algorithms in both 2-D and 3-D. Figure 33 shows the hop stretch factor. These algorithms route the message or the packet with the path that is nearly the shortest in both 2-D and 3-D environments. The *Greedy* routing algorithm gives slightly better hop stretch factor than the other three in both 2-D and 3-D.

4.4.2 Randomized Algorithms

We refer to the randomized algorithms using the quadrant-space partition as *random1* and the half-space partition as *random2* in the following figures. We use three deterministic routing algorithms (*Compass*, *Greedy* and *Ellipsoid* routing algorithms) to hybridize with both of our 3-D randomized algorithms and compare them with the 2-D randomized algorithm. Figure 34 gives the delivery rate on UDG. The curves for 3-D are shifted to right with respect to the increase of radius. Figure 35 shows the hop stretch factors. The curves for 3-D are also shifted. The two 3-D randomized algorithms give similar delivery rates. However, the 3-D randomized algorithm using the half-space partition performs better in terms of the hop stretch factor.

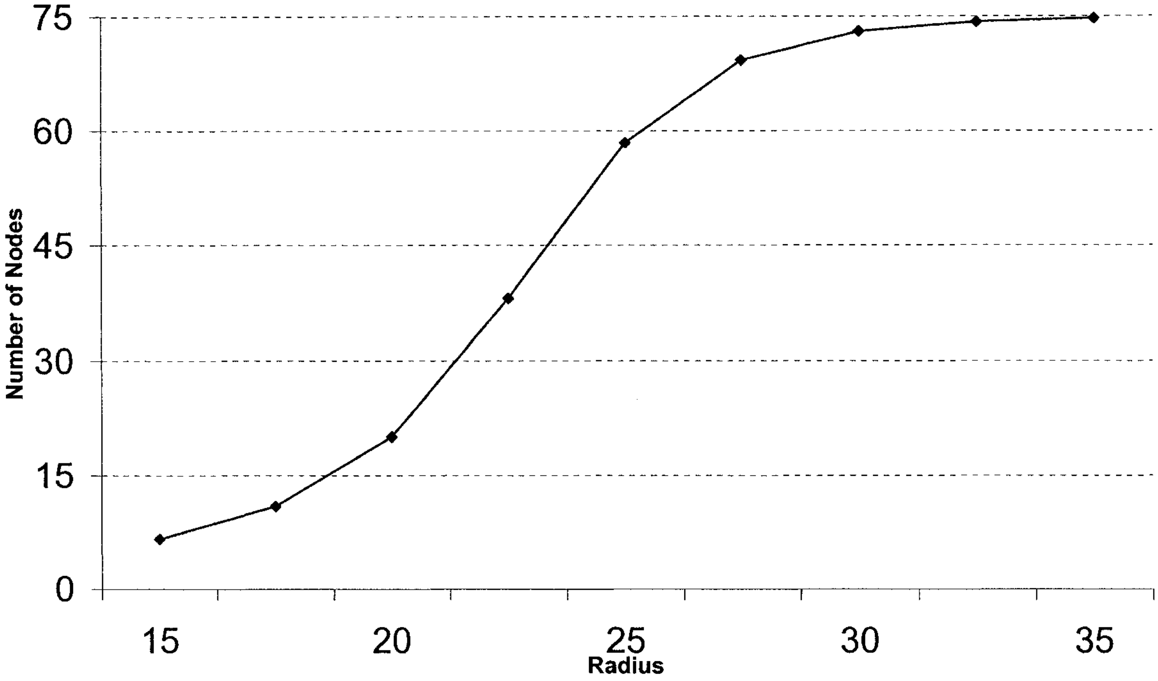


Figure 14: Average number of nodes in the LCC.

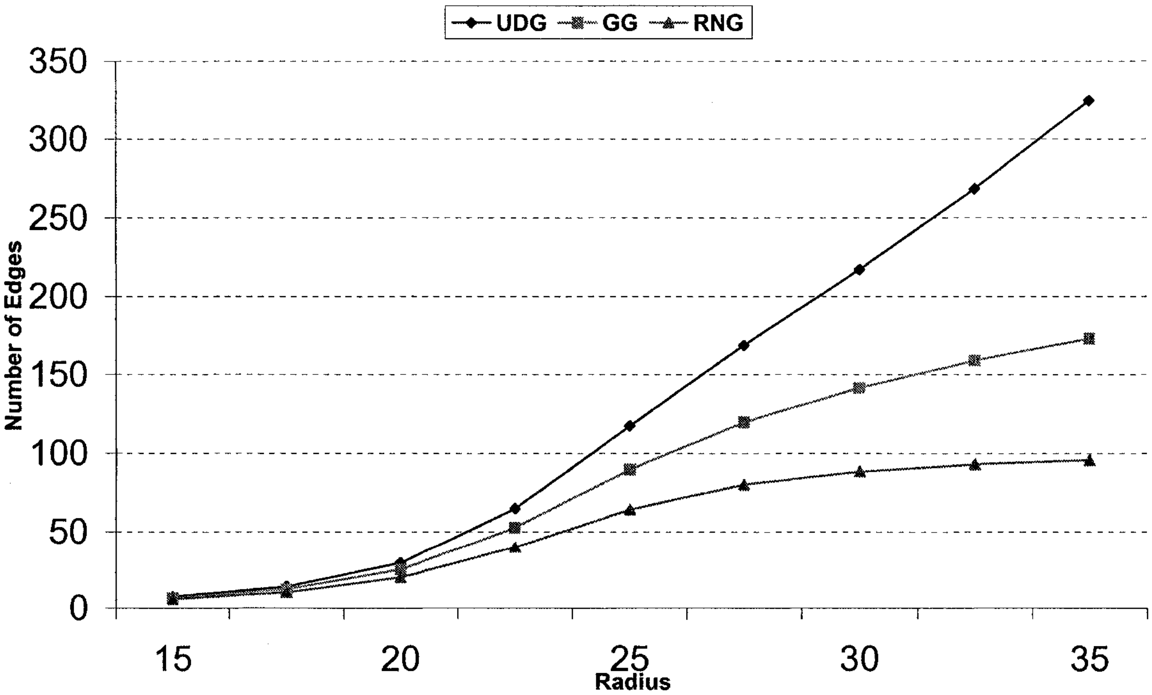


Figure 15: Average number of edges in the LCC of different graphs.

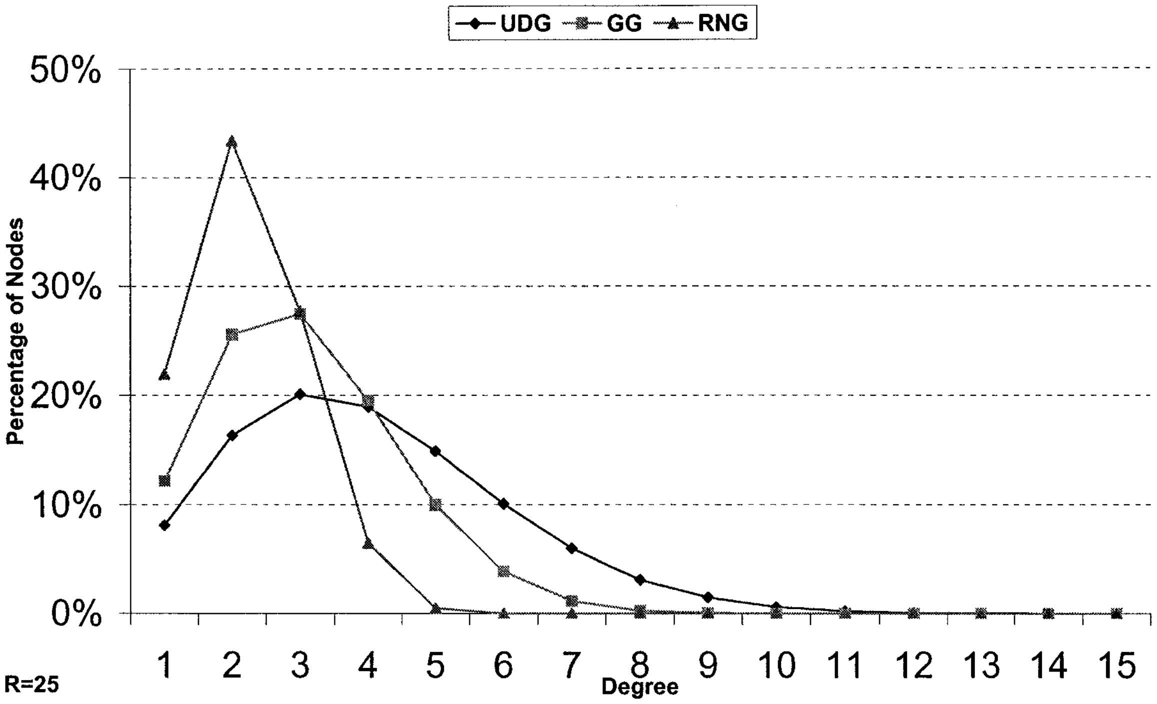


Figure 16: Distribution of nodes with various degrees in the LCC of different graphs for the radius of 25.

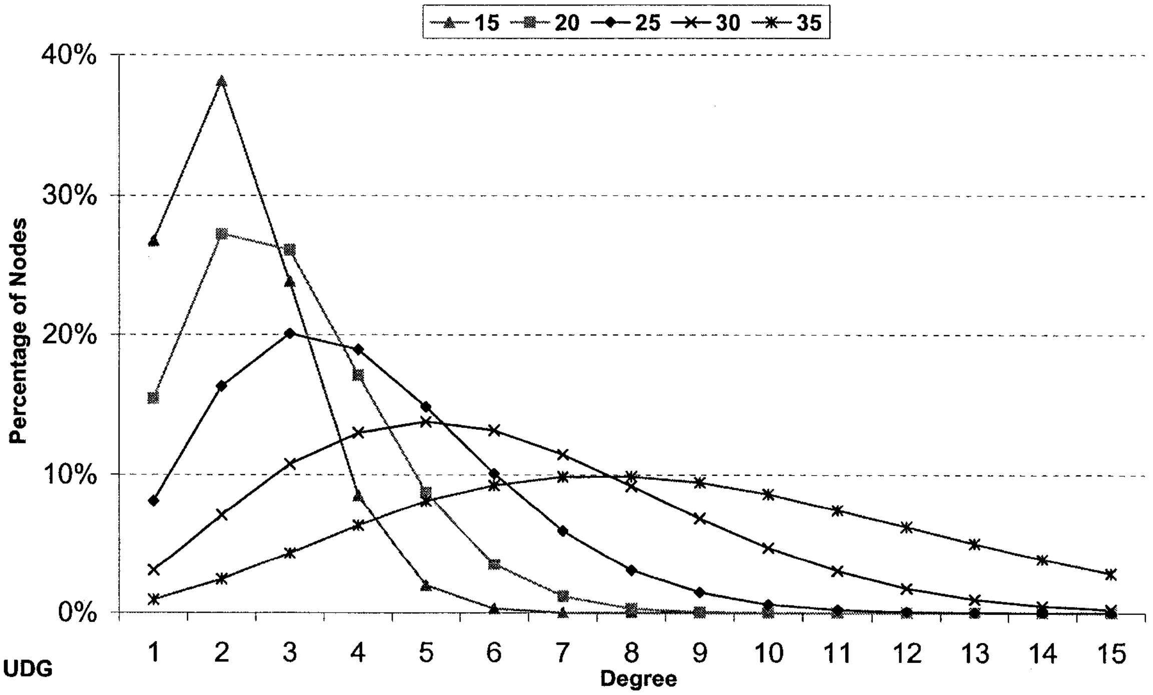


Figure 17: Distribution of nodes with various degrees in the LCC of UDG for different radii.

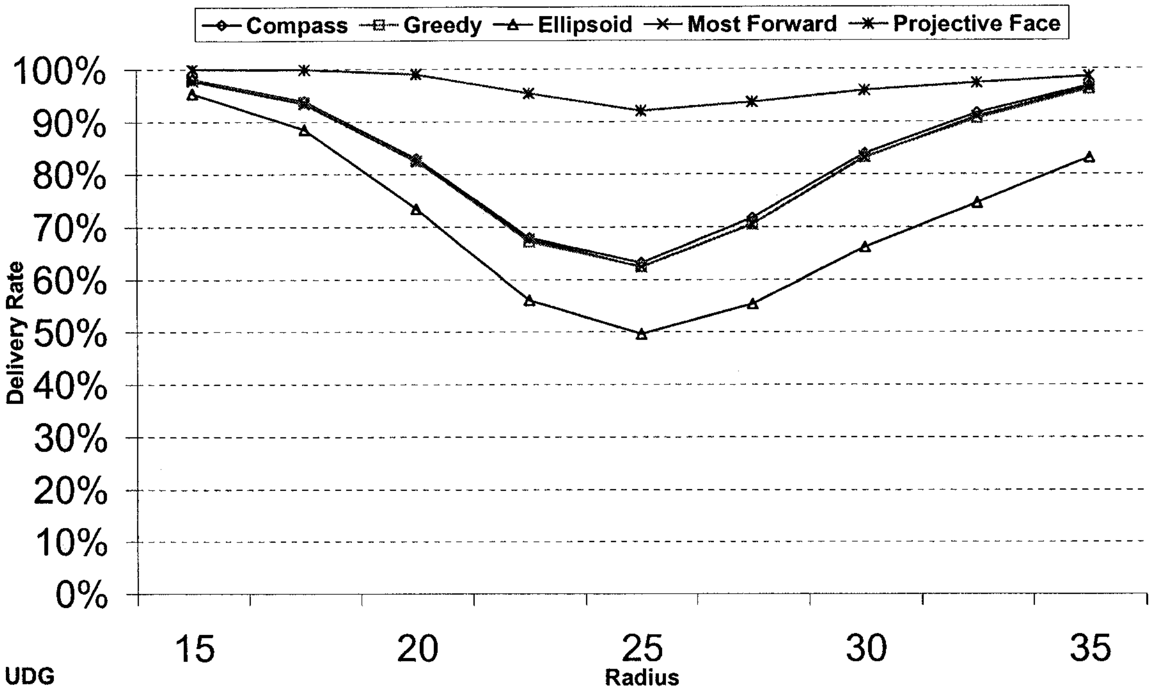


Figure 18: The delivery rate on UDG.

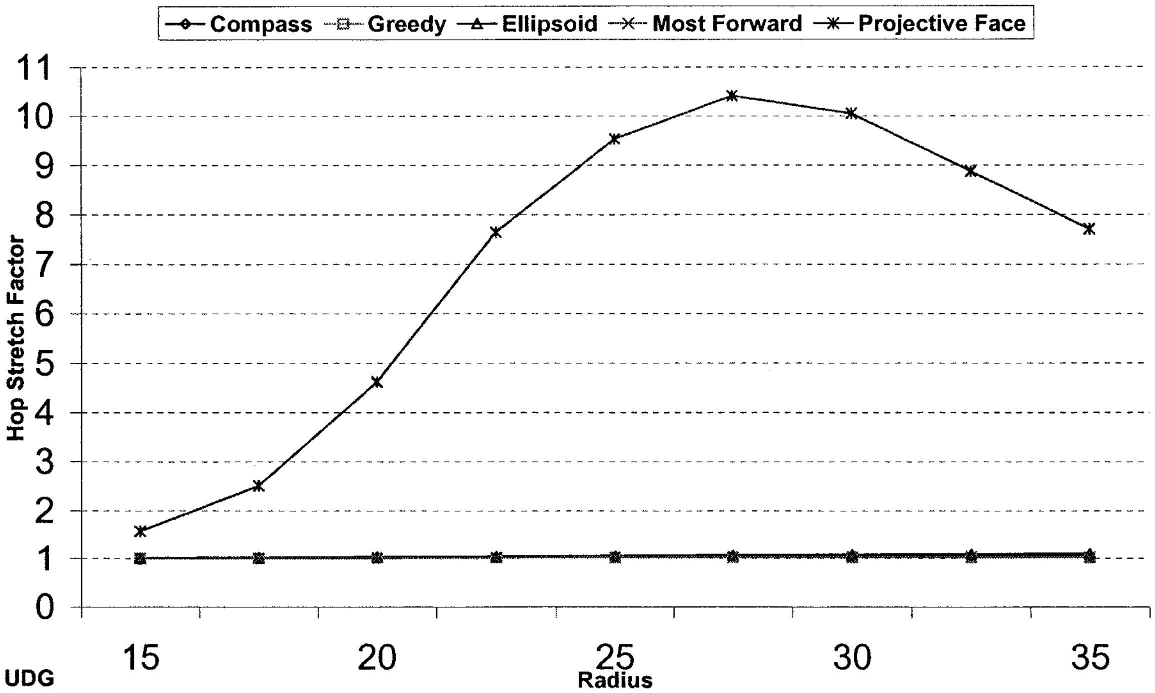


Figure 19: The hop stretch factor on UDG.

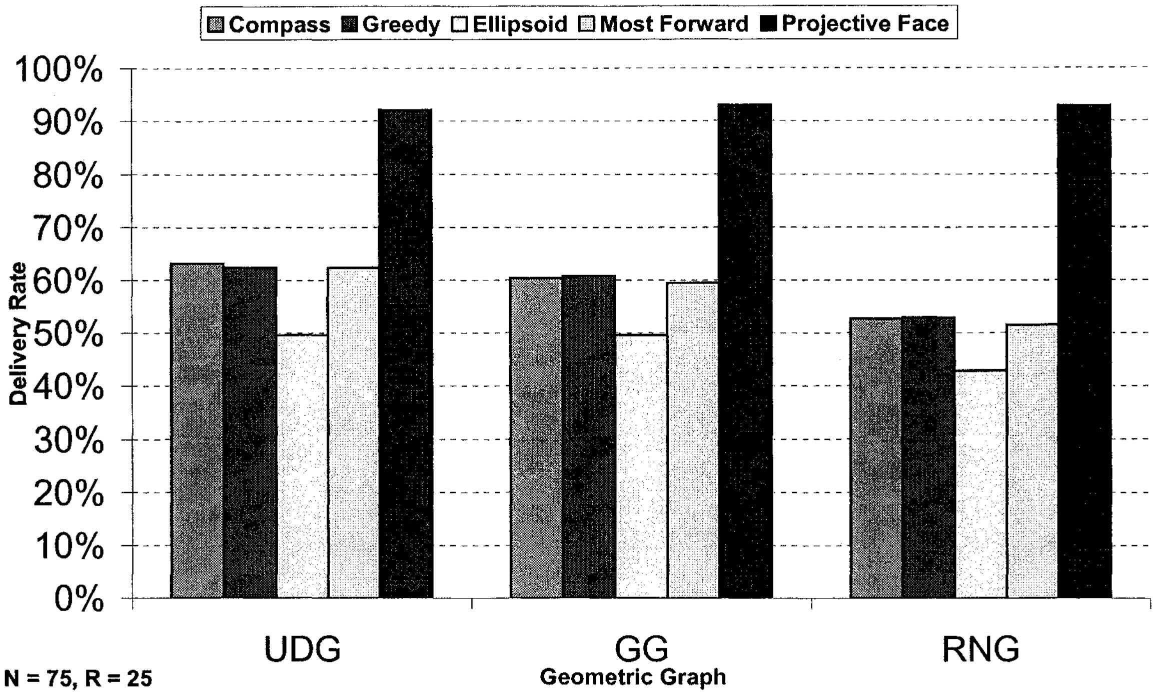


Figure 20: The delivery rate for the radius of 25 on different graphs.

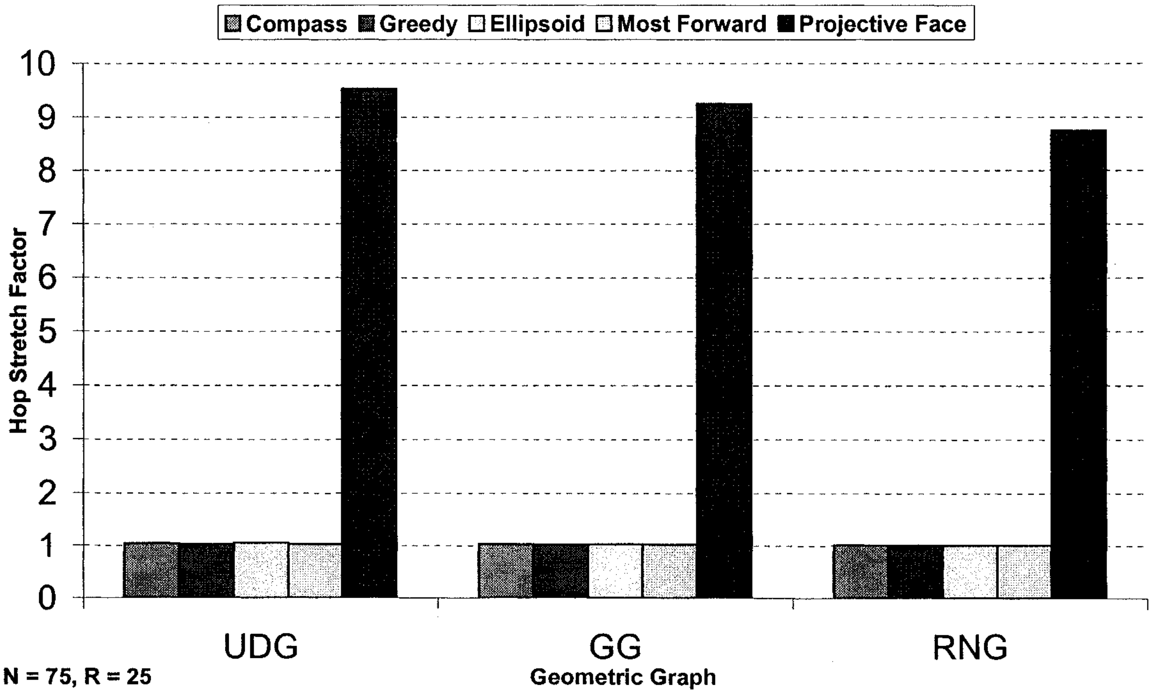


Figure 21: The hop stretch factor for the radius of 25 on different graphs.

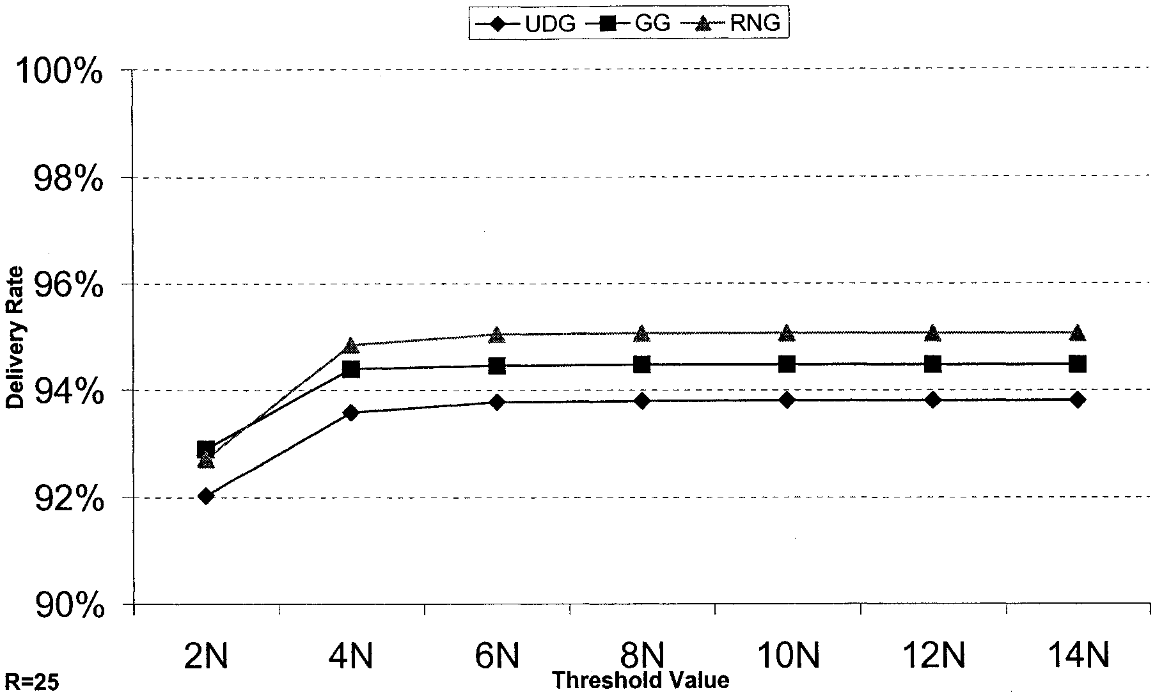


Figure 22: The delivery rate for *Projective FACE* routing on two orthogonal planes while the threshold value is increased.

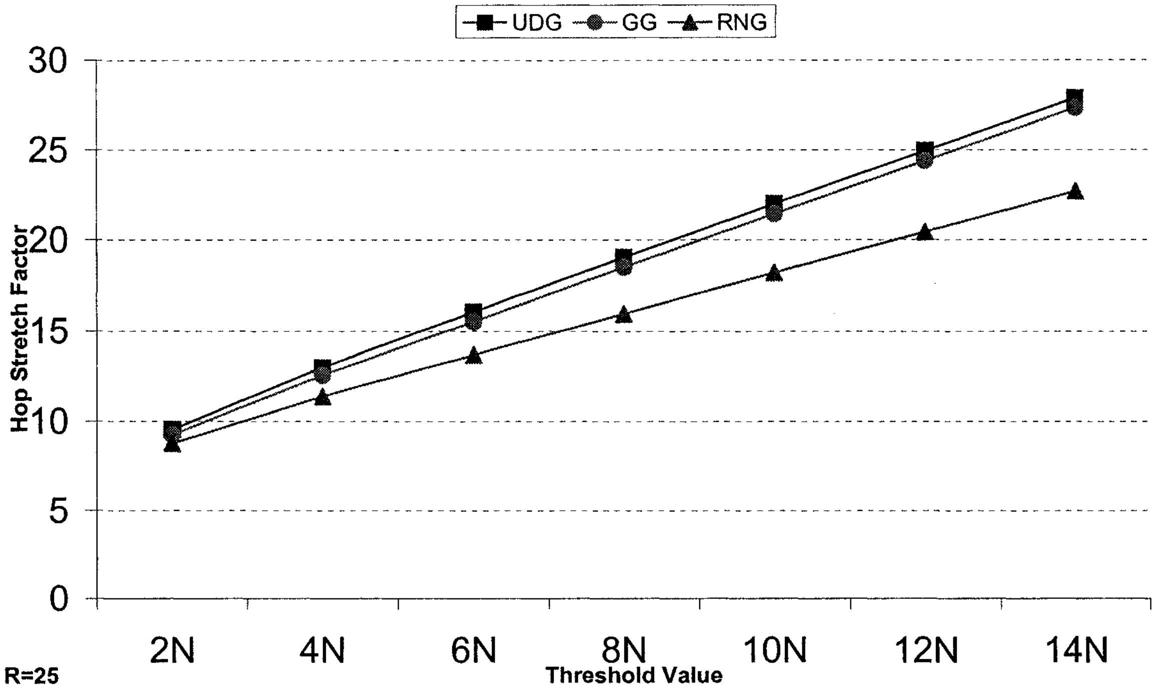


Figure 23: The hop stretch factor for *Projective FACE* routing on two orthogonal planes while the threshold value is increased.

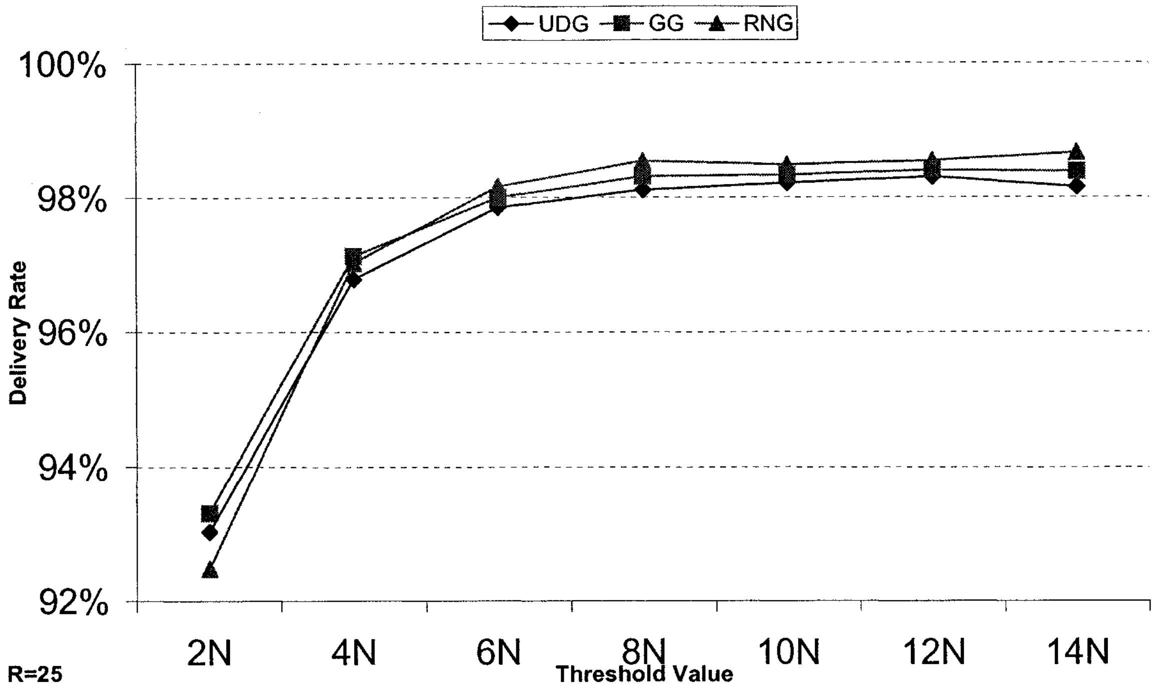


Figure 24: The delivery rate for *Adaptive Least-Squares Projective FACE* routing on two orthogonal planes while the threshold value is increased. The ABS is set to N.

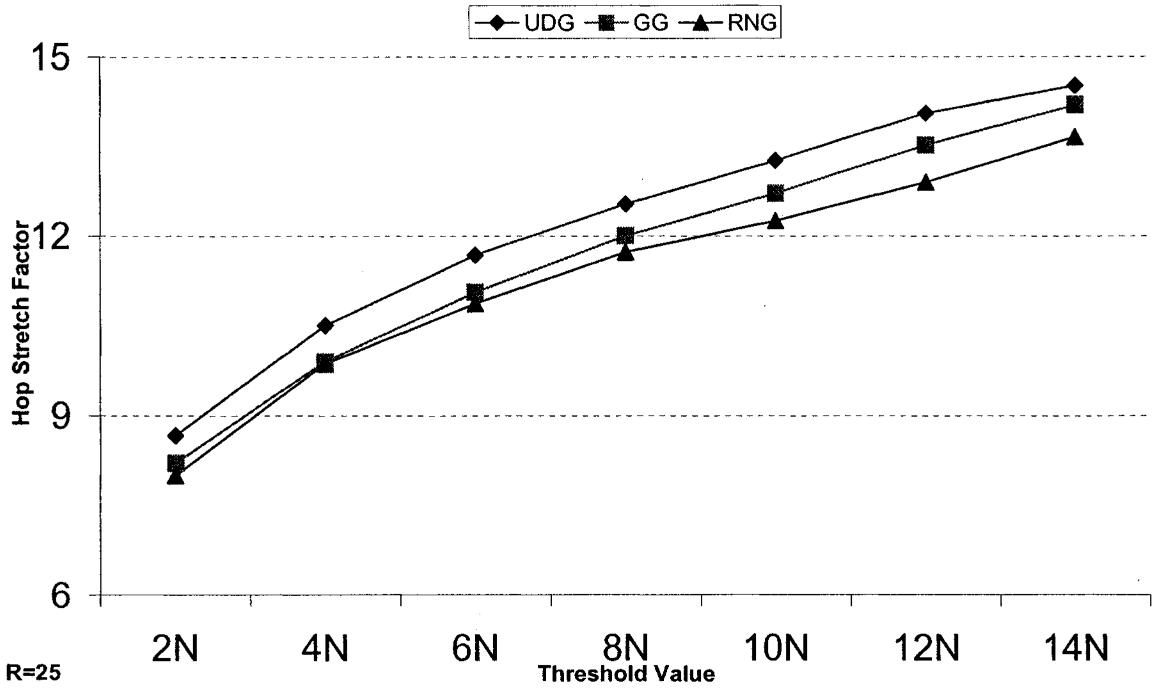


Figure 25: The hop stretch factor for *Adaptive Least-Squares Projective FACE* routing on two orthogonal planes while the threshold value is increased. The ABS is set to N.

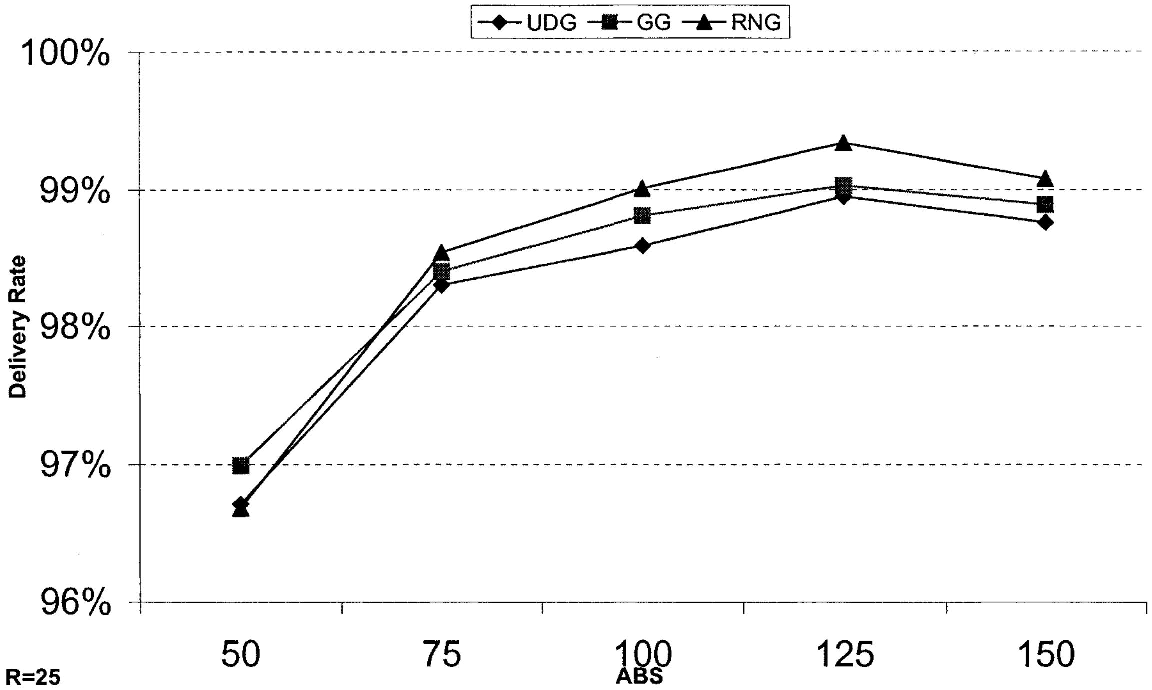


Figure 26: The delivery rate for *Adaptive Least-Squares Projective FACE* routing on two orthogonal planes while the ABS varies. The threshold value is set to $12N$.

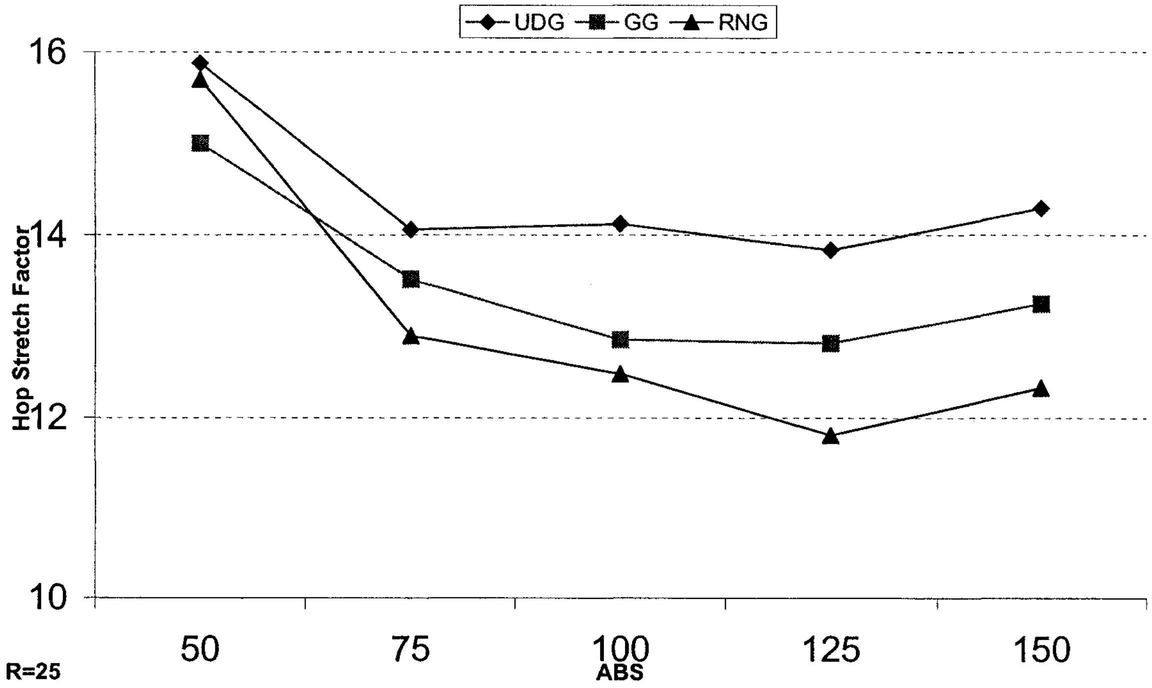


Figure 27: Hop stretch factor for *Adaptive Least-Squares Projective FACE* routing on two orthogonal planes while the ABS varies. The threshold value is set to $12N$.

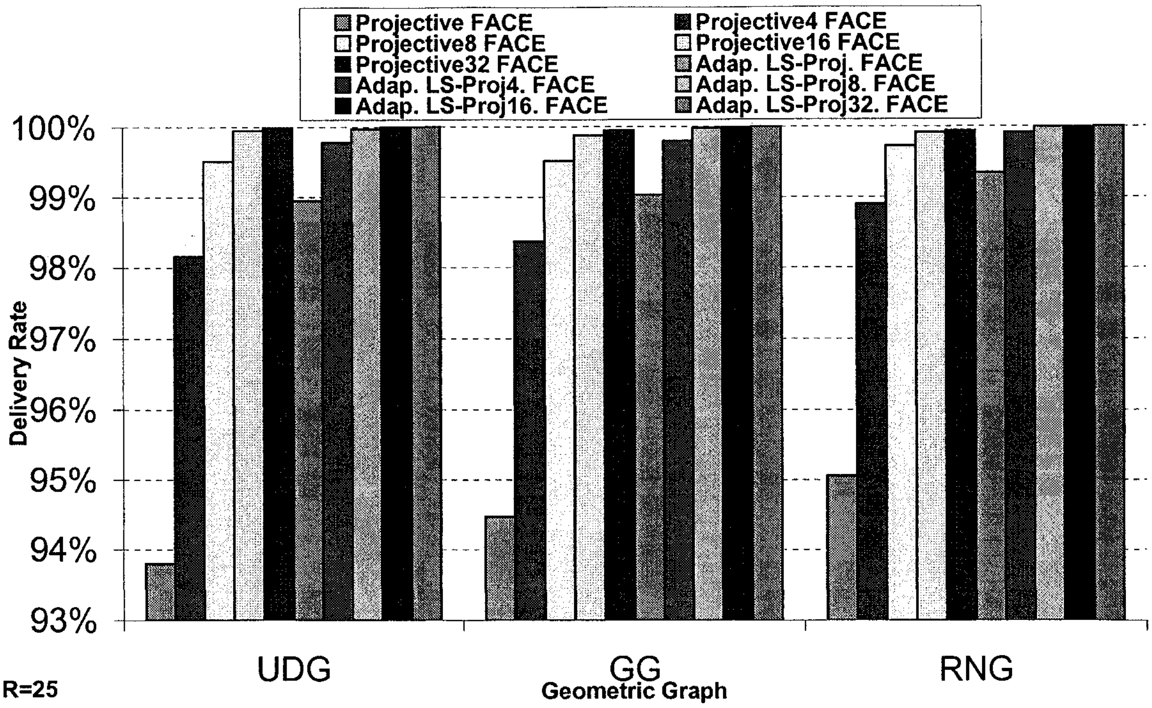


Figure 28: The delivery rate of our projective-based routing algorithms with various numbers of projection planes available for the radius of 25.

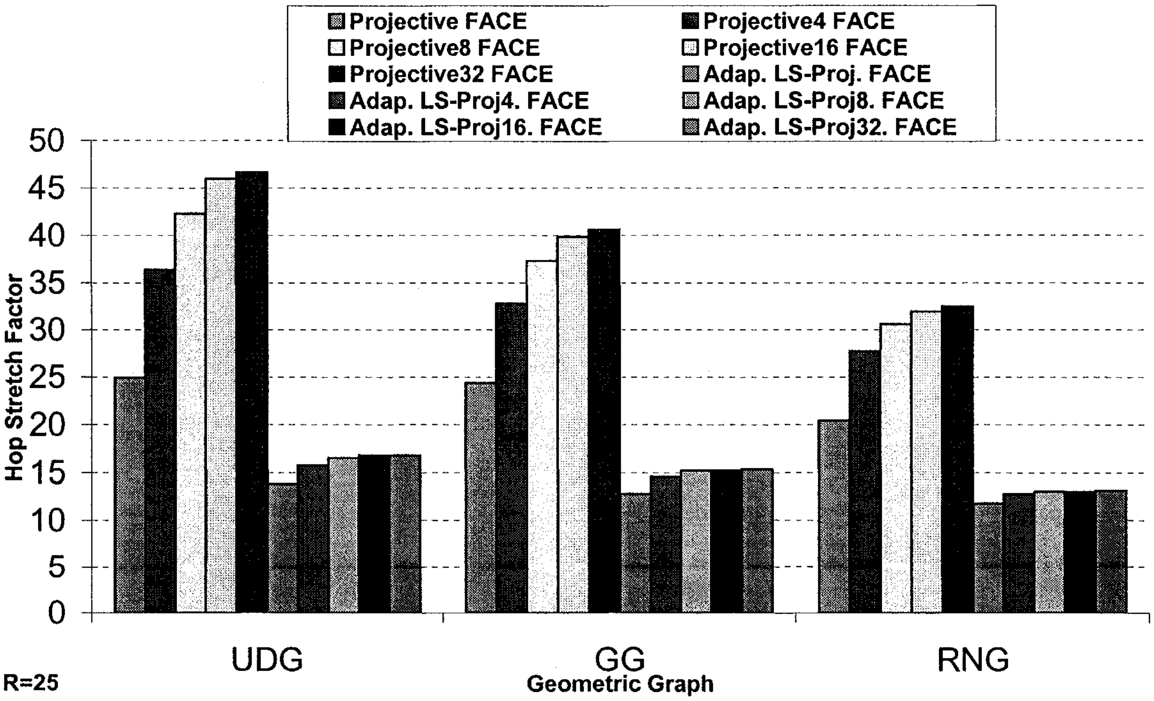


Figure 29: The hop stretch factor of our projective-based routing algorithms with various numbers of projection planes available for the radius of 25.

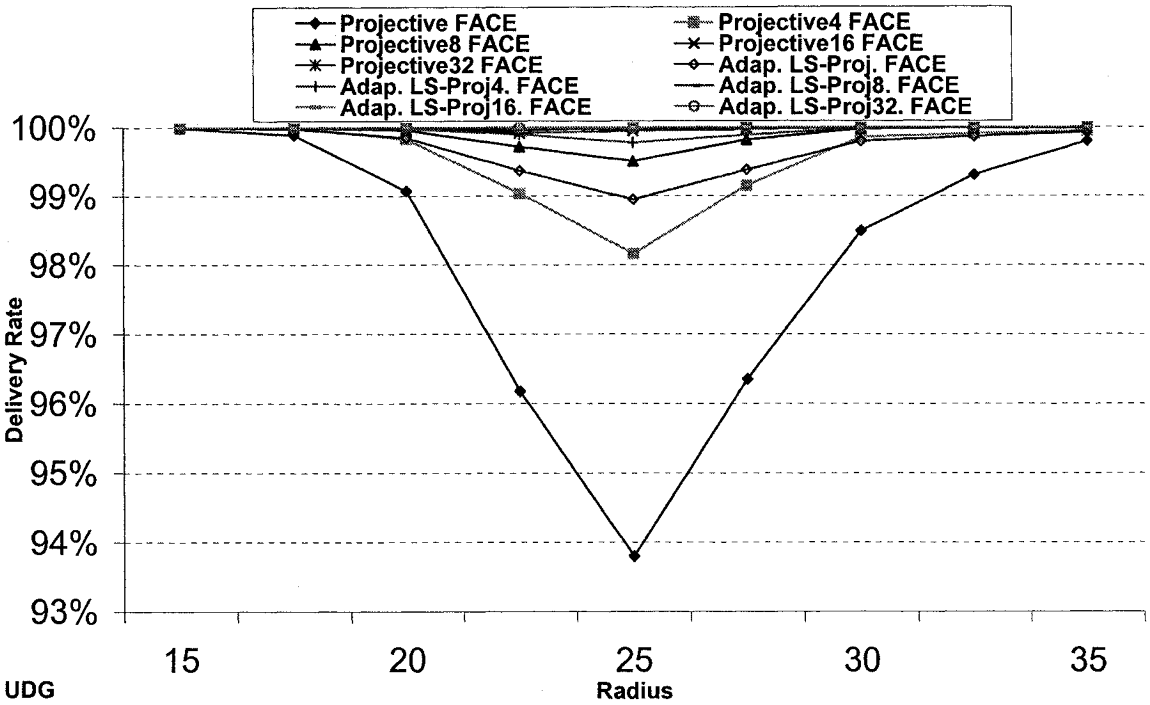


Figure 30: The delivery rate of our projective-based routing algorithms with various numbers of projection planes available for different radii.

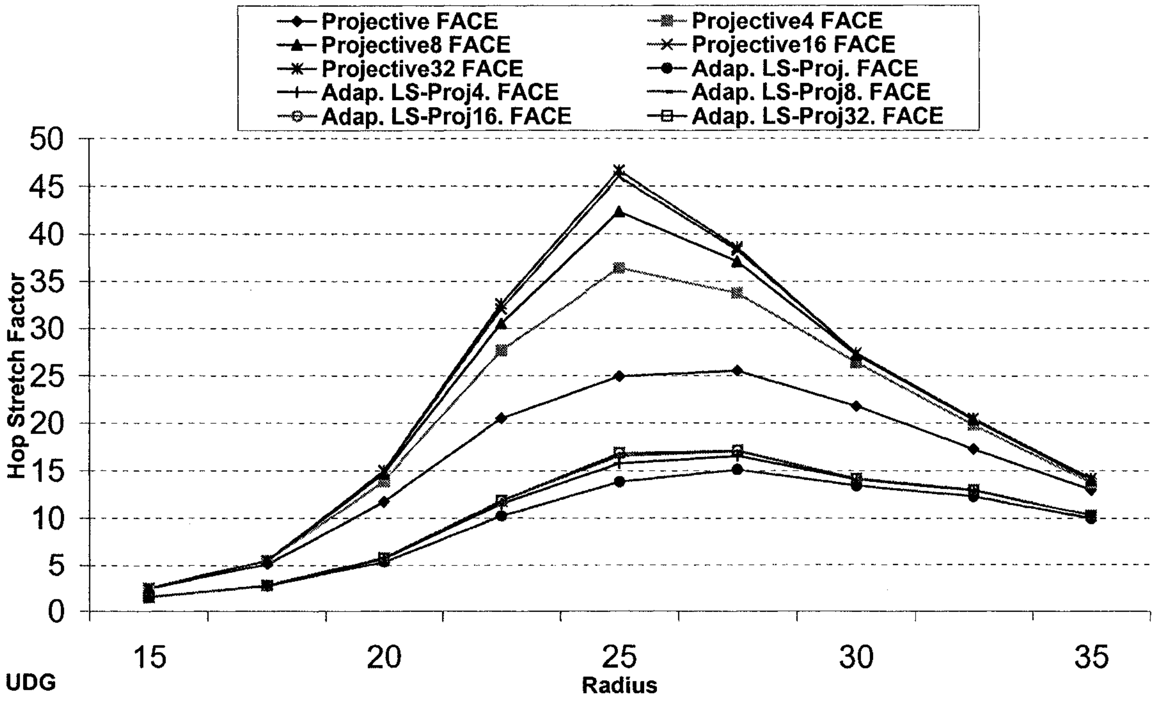


Figure 31: The hop stretch factor of our projective-based routing algorithms with various numbers of projection planes available for different radii.

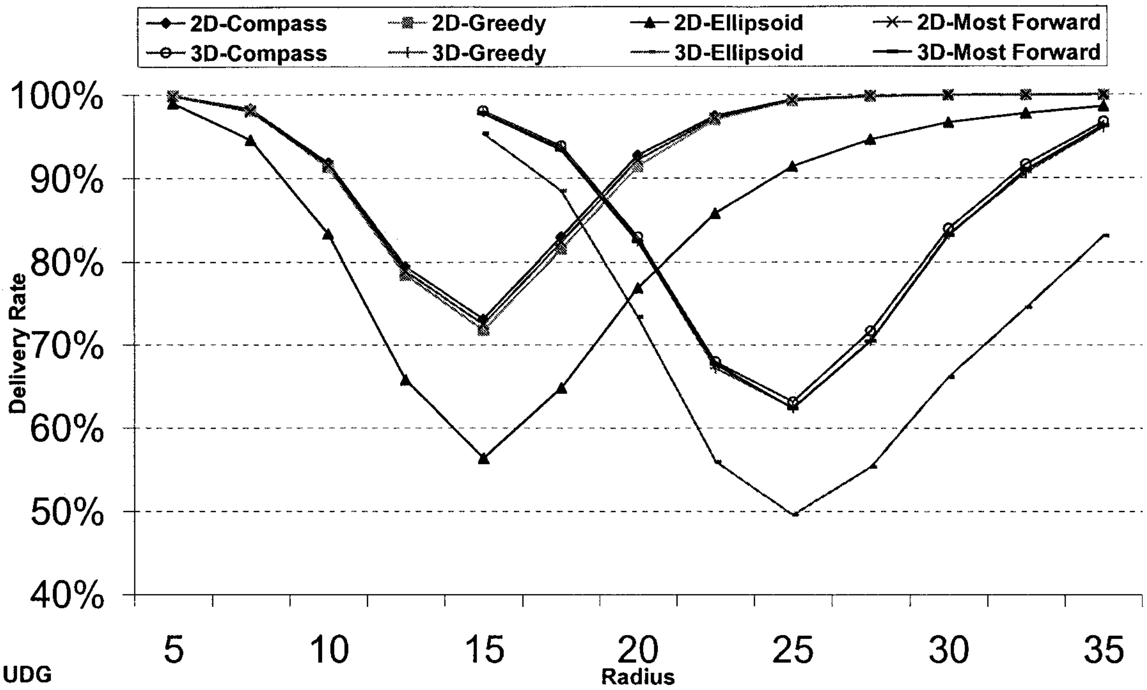


Figure 32: Comparison of the delivery rate for the deterministic routing algorithms in 2-D and 3-D.

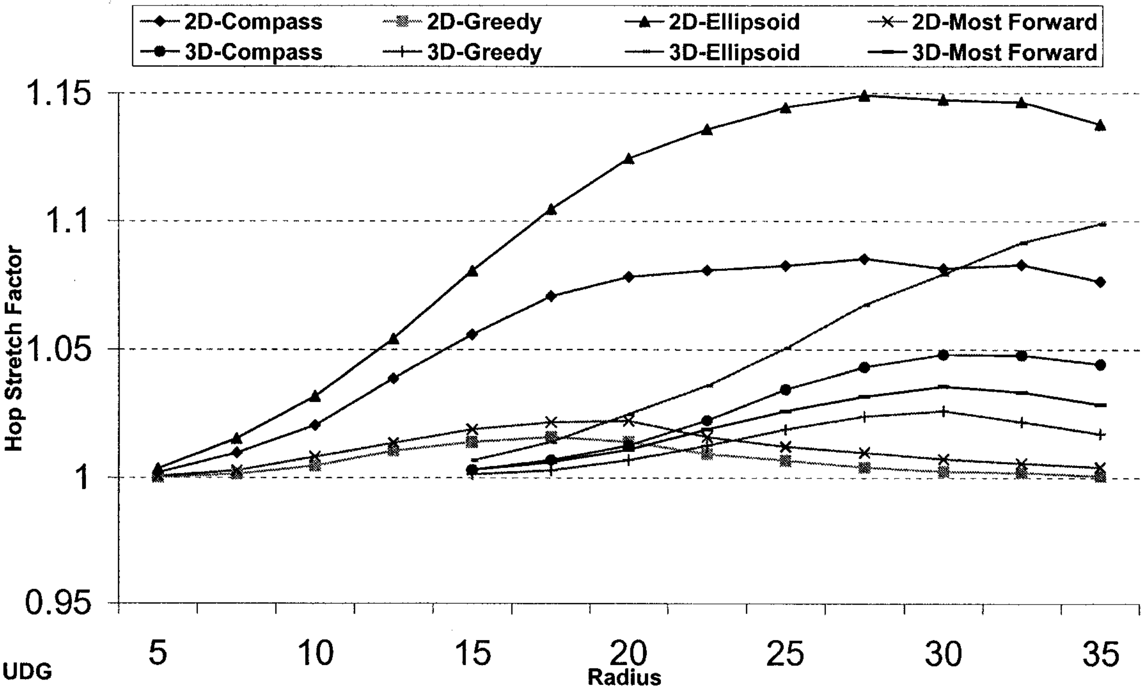


Figure 33: Comparison of the hop stretch factor for the deterministic routing algorithms in 2-D and 3-D.

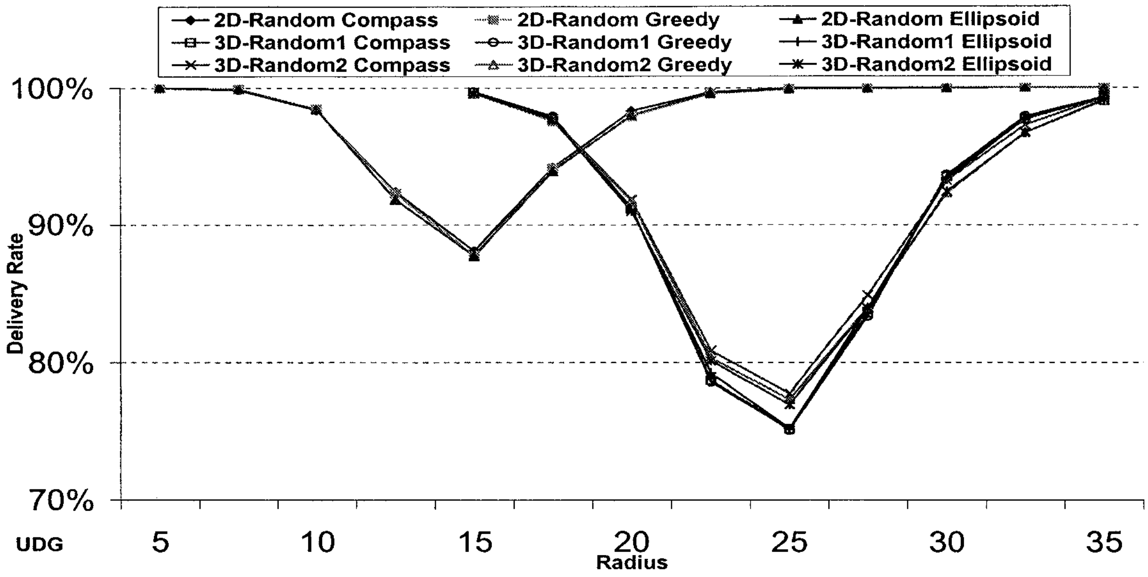


Figure 34: Comparison of the delivery rate for the randomized routing algorithms in 2-D and 3-D.

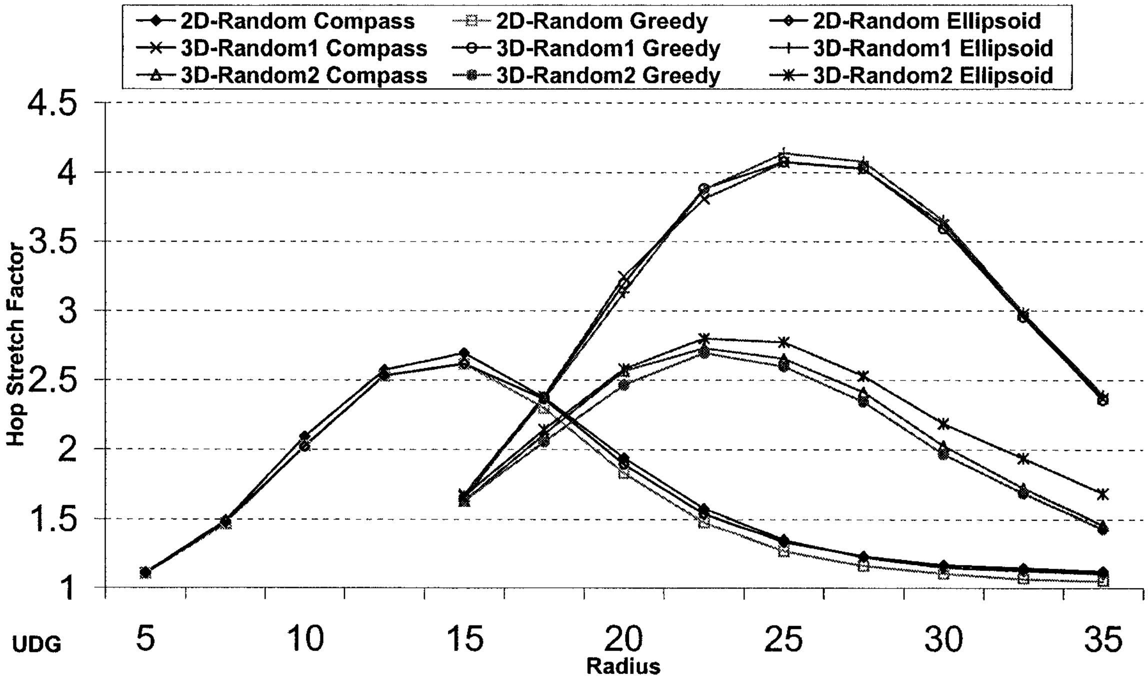


Figure 35: Comparison of the hop stretch factor for the randomized routing algorithms in 2-D and 3-D.

Chapter 5

Conclusion and Future Work

We have studied the UDG and its associated spanning subgraphs, GG and RNG, in 3-D and extended the position-based routing algorithms to adapt to the context of 3-D. Our simulation showed that the *Ellipsoid* routing algorithm does not give better delivery rate than the *Greedy* routing algorithm as the number of nodes increases. This conclusion differs from what is claimed in [YS04].

The primary interest of the thesis is the 3-D localized position-based routing algorithm. Before proposing our projective-based algorithms, we started by a new routing approach in 3-D and named it *ICR* (Intersection Count Routing) algorithm. The algorithm first defines a number of planes using the source and destination nodes and their neighboring nodes as third point of each plane. The number of planes is then equal to the sum of neighboring nodes of the source and destination. The routing decision of a current node is determined by the greatest number of intersections between the segment formed by the source and its neighboring node and the planes defined earlier. This 3-D routing algorithm, however, gives worse performance than *Greedy* routing algorithm. The result of this algorithm is then excluded from the thesis work.

Our first proposed algorithm, *Projective FACE* routing algorithm on two orthogonal planes, performs significantly better in terms of delivery rate than the other deterministic routing algorithms. However, the projected graphs may have crossing edges that cannot be eliminated by using only the local information. The delivery is thus not guaranteed. We further attempted to explore different approaches and proposed three heuristics. The heuristics of adaptive behavior scale (ABS) and the least-squares projection (LSP) plane make our projective-based algorithm more efficient by improving the hop stretch factor. By having the heuristic of multi-projection-plane strategy, our *Adaptive Least-Squares Projective FACE* routing algorithm on 16 or more projection planes gives nearly certain delivery rate on UDG (100% generated by our simulations).

The projected graphs on the planes are not necessarily to be planar graphs and a planar graphs cannot be extracted from these projected graphs using a localized method. We have to use a threshold value to terminate the routing process if the number of hops traversed exceed that value. However, we have no heuristics to detect the looping of our projective-based algorithms on the projected graphs that are possibly non-planar. If the occurrence of looping can be detected as soon as possible, the efficiency of our algorithms will be potentially improved. One interesting approach to deal with this problem is to introduce a constant memory, $O(1)$, for memorizing a node and an edge. Once a certain number of hops have been traversed, the algorithm will pick a node and store the position information of that node as well as the edge that leads to that node as the part of the packet or message being delivered. For the subsequent routing process, the verification of the node and the edge is required. If the same node and the edge are detected, this indicates the occurrence of looping. Once the looping is detected, we can use the heuristics of our adaptive approach and multi-projection-plane strategy proposed in this thesis. Hence, the hop stretch factor will be potentially reduced.

Our simulation results showed that the 3-D deterministic routing algorithms (*Greedy*, *Compass*, *Ellipsoid* and *Most Forward* routing algorithms) give nearly the shortest path although the delivery rate of these algorithms are relatively low. By hybridizing one of the four algorithms with our *Projective FACE* or *Adaptive Least-Squares Projective FACE* routing algorithms, we are able to benefit from both algorithms in which the former can route with shorter path and the latter can give much better delivery rate. This also remains as our future work.

Bibliography

[ANO05] S. Ansari, L. Narayanan, and J. Opatrny. A generalization of the face routing algorithm to a class of non-planar networks. In *Proceedings of Mobiquitous*, pp. 213–225, San Diego, 2005.

[BDEK02] P. Bose, L. Devroye, W. S. Evans, and D. G. Kirkpatrick. On the spanning ratio of gabriel graphs and beta-skeletons. In *Proceedings of the 5th Latin American Symposium on Theoretical Informatics*, pp. 479–493. Springer-Verlag, 2002.

[BFNO03] L. Barrière, P. Fraigniaud, L. Narayanan, and J. Opatrny. Robust position-based routing in wireless ad hoc networks with irregular transmission ranges. *Wireless Communications and Mobile Computing Journal*, vol. 3/2, pp. 141–153, 2003.

[BMSU01] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, vol. 7/6, pp. 609–616, 2001.

[CJL$^+$01] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, A. Qayyum, and L. Viennot. Optimized link state routing protocol. In *IEEE INMIC*, 2001.

[Dij59] E. Dijkstra. A note on two problems in connection with graphs. *Numerical Mathematics*, pp. 269–271, 1959.

[ECU05] E. Kranakis J. Opatrny L. Stacho H. Tejeda E. Chavez, S. Dobrev and J. Urrutia. Half-space proximal: A new local test for extracting a bounded dilation spanner. In *proceedings of OPODIS*, 2005.

[FF62] L. Ford and D. Fulkerson. Princeton, NJ: Princeton University Press. *Numerical Mathematics*, 1962.

[FHN04] T. Fevens, I. Haque, and L. Narayanan. A class of randomized routing algorithms in mobile ad hoc networks. In *Proceedings of the 1st Algorithms for Wireless and Ad-hoc Networks (A-SWAN)*, Boston, 2004.

[Fin87] G.G. Finn. Routing and addressing problems in large metropolitan-scale internetworks. Technical report, ISI Research Report ISU/RR-87-180, 1987.

[GGH$^+$01a] J. Gao, L. J. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Discrete mobile centers. In *Proceedings of the seventeenth annual symposium on Computational geometry*, pp. 188–196. ACM Press, 2001.

[GGH$^+$01b] J. Gao, L. J. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Geometric spanner for routing in mobile networks. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, pp. 45–55. ACM Press, 2001.

[GS69] K. Gabriel and R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, vol. 18, pp. 259–278, 1969.

[GSB03] S. Giordano, I. Stojmenovic, and L. Blazevic. Position based routing algorithms for ad hoc networks: A taxonomy. In *Ad Hoc Wireless Networking.* Norwell, MA: Kluwer, 2003.

[HL99] Z. J. Haas and B. Liang. Ad hoc mobility management with uniform quorum systems. *IEEE/ACM Transactions on Networking*, vol. 7/2, pp. 228–240, 1999.

[JT92] J. Jaromczyk and G. Toussaint. Relative neighborhood graphs and their relatives. In *Proceedings of the IEEE*, vol. 80, pp. 1502–1517, 1992.

[KK00] B. Karp and H. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pp. 243–254. ACM Press, 2000.

[KSU99] E. Kranakis, H. Singh, and J. Urrutia. Compass routing on geometric networks. In *Proceedings of the 11th Canadian Conference on Computational Geometry*, pp. 51–54, Vancouver, 1999.

[KV98] Y.-B. Ko and N. H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. In *Mobile Computing and Networking*, pp. 66–75, 1998.

[KWZ02] F. Kuhn, R. Wattenhofer, and A. Zollinger. Asymptotically optimal geometric mobile ad-hoc routing. In *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM)*, pp. 213–225, Atlanta, Georgia, 2002.

[LCW02] X.-Y. Li, G. Calinescu, and P.-J. Wan. Distributed construction of planar spanner and routing for ad hoc wireless networks, 2002.

[LJD$^+$00] J. Li, J. Jannotti, D. De Couto, D. Karger, and R. Morris. A scalable location service for geographic ad-hoc routing. In *Proceedings of the 6th ACM International Conference on Mobile Computing and Networking (MobiCom '00)*, pp. 120–130, 2000.

[MWH01] M. Mauve, J. Widmer, and H. Hartenstein. A survey on position-based routing in mobile ad hoc networks. *IEEE Network Magazine*, vol. 15/6, pp. 30–39, Novermber 2001.

[NK84] R. Nelson and L. Kleinrock. The spatial capacity of a slotted aloha multihop packet radio network with capture. *IEEE Transactions on Communications*, vol. COM-32/6, pp. 684–694, 1984.

[PB94] E. C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance vector (dsdv) for mobile computers. In *Proceedings of the SIGCOM'94 Conference on Communication Architectures, Protocols and Applications*, 1994.

[PH97] M. Pearlman and Z. Haas. Determining the optimal configuration for zone routing protocol. *IEEE Journal on Selected Areas in Communications*, vol. 17/8, pp. 1395–1414, 1997.

[Sha98] C. M. Shakarji. Least-squares fitting algorithms of the nist algorithm testing system. *Research of the National Institute of Standards and Technology*, vol. 103/6, pp. 633–641, 1998.

[SL01] I. Stojmenovic and X. Lin. Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, vol. 12/10, pp. 1023–1032, 2001.

[TK84] H. Takagi and L. Kleinrock. Optimal transmission ranges for randomly distributed packet radio terminals. *IEEE Transactions on Communications*, vol. COM-32/3, pp. 246–257, 1984.

[Yao77] A. C.-C. Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. Technical report, 1977.

[YS04] K. Yamazaki and K. Sezaki. The proposal of geographical routing protocols for location-aware services. *Electronics and Communications in Japan*, vol. 87/4, 2004.

Appendix A

Appendix

	R = 15		R = 20		R = 25		R = 30		R = 35	
Algorithm	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ
Compass	98.06%	1.37	82.98%	4.03	63.15%	4.43	83.96%	3.35	96.73%	1.74
Greedy	97.97%	1.47	82.52%	4.03	62.40%	4.21	83.32%	3.45	96.09%	1.97
Ellipsoid	95.37%	2.09	73.42%	4.05	49.57%	4.28	66.18%	4.17	83.09%	3.50
Most Forward	97.75%	1.47	82.41%	4.13	62.34%	4.58	83.16%	3.17	96.30%	1.85
Projective FACE	99.99%	0.10	99.00%	0.97	92.03%	2.50	95.91%	1.82	98.51%	1.25

Table 3: The delivery rate on UDG.

	R = 15		R = 20		R = 25		R = 30		R = 35	
Algorithm	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ
Compass	1.00	0.03	1.01	0.06	1.03	0.10	1.05	0.12	1.04	0.12
Greedy	1.00	0.02	1.01	0.04	1.02	0.07	1.03	0.09	1.02	0.07
Ellipsoid	1.01	0.05	1.02	0.09	1.05	0.12	1.08	0.16	1.10	0.19
Most Forward	1.00	0.04	1.01	0.06	1.03	0.09	1.04	0.11	1.03	0.10
Projective FACE	1.58	3.09	4.62	7.98	9.54	12.30	10.05	14.58	7.71	13.83

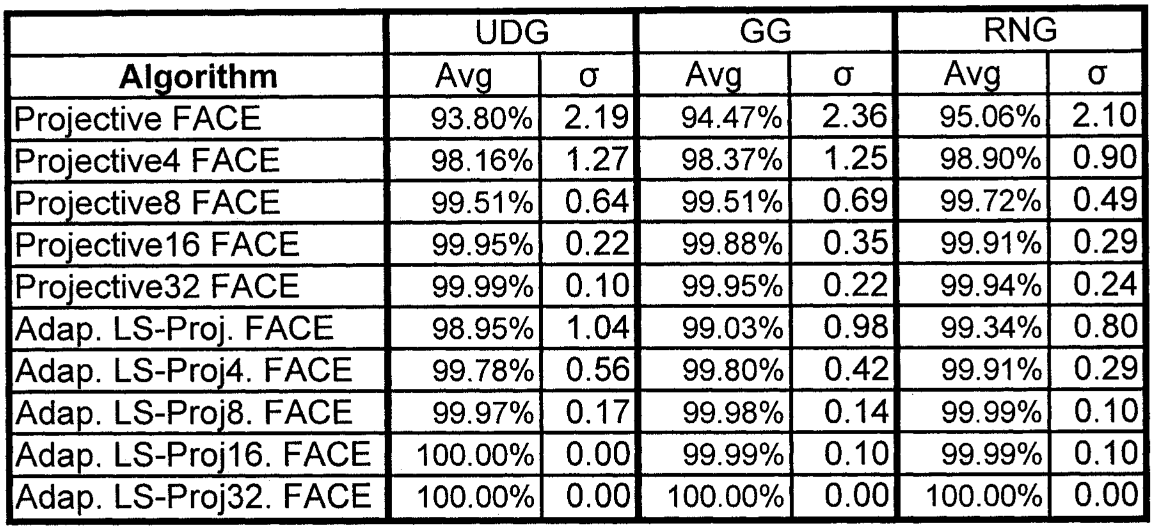
Table 4: The hop stretch factor on UDG.

	UDG		GG		RNG	
Algorithm	Avg	σ	Avg	σ	Avg	σ
Compass	63.15%	4.43	60.41%	4.93	52.63%	5.05
Greedy	62.40%	4.21	60.76%	5.38	52.88%	5.00
Ellipsoid	49.57%	4.28	49.53%	5.24	42.77%	5.01
Most Forward	62.34%	4.58	59.44%	5.05	51.43%	4.81
Projective FACE	92.03%	2.50	92.90%	2.70	92.72%	2.63

Table 5: The delivery rate for the radius of 25 on different graphs.

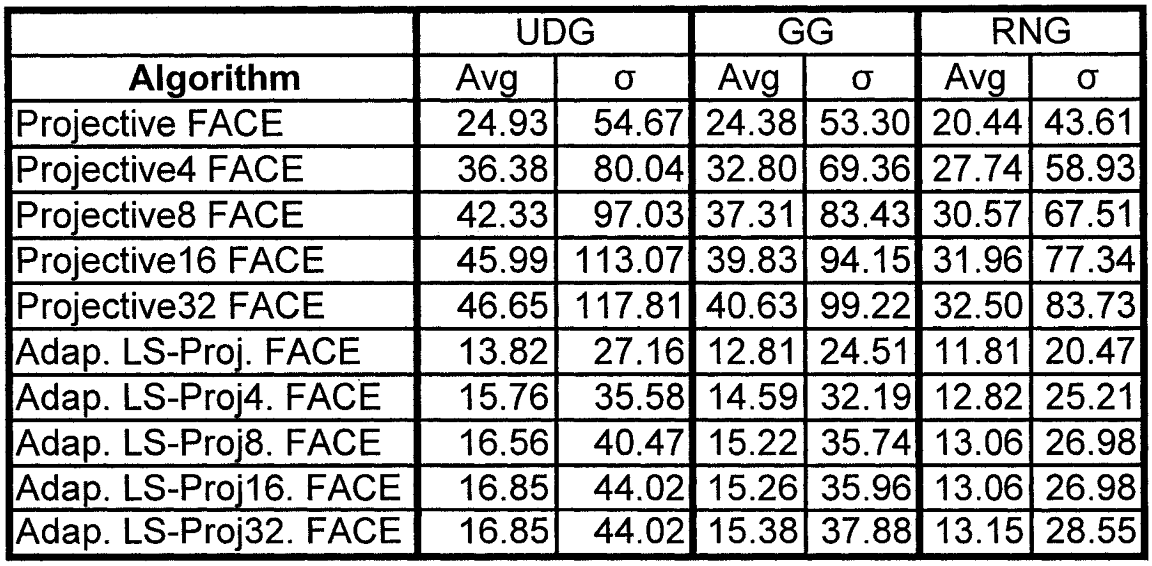
	UDG		GG		RNG	
Algorithm	Avg	σ	Avg	σ	Avg	σ
Compass	1.03	0.10	1.03	0.08	1.02	0.06
Greedy	1.02	0.07	1.02	0.07	1.01	0.05
Ellipsoid	1.05	0.12	1.03	0.09	1.01	0.05
Most Forward	1.03	0.09	1.02	0.07	1.01	0.05
Projective FACE	9.54	12.30	9.25	11.77	8.74	10.32

Table 6: The hop stretch factor for the radius of 25 on different graphs.



	UDG		GG		RNG	
Algorithm	Avg	σ	Avg	σ	Avg	σ
Projective FACE	93.80%	2.19	94.47%	2.36	95.06%	2.10
Projective4 FACE	98.16%	1.27	98.37%	1.25	98.90%	0.90
Projective8 FACE	99.51%	0.64	99.51%	0.69	99.72%	0.49
Projective16 FACE	99.95%	0.22	99.88%	0.35	99.91%	0.29
Projective32 FACE	99.99%	0.10	99.95%	0.22	99.94%	0.24
Adap. LS-Proj. FACE	98.95%	1.04	99.03%	0.98	99.34%	0.80
Adap. LS-Proj4. FACE	99.78%	0.56	99.80%	0.42	99.91%	0.29
Adap. LS-Proj8. FACE	99.97%	0.17	99.98%	0.14	99.99%	0.10
Adap. LS-Proj16. FACE	100.00%	0.00	99.99%	0.10	99.99%	0.10
Adap. LS-Proj32. FACE	100.00%	0.00	100.00%	0.00	100.00%	0.00

Table 7: The delivery rate of our projective-based routing algorithms for the radius of 25.



	UDG		GG		RNG	
Algorithm	Avg	σ	Avg	σ	Avg	σ
Projective FACE	24.93	54.67	24.38	53.30	20.44	43.61
Projective4 FACE	36.38	80.04	32.80	69.36	27.74	58.93
Projective8 FACE	42.33	97.03	37.31	83.43	30.57	67.51
Projective16 FACE	45.99	113.07	39.83	94.15	31.96	77.34
Projective32 FACE	46.65	117.81	40.63	99.22	32.50	83.73
Adap. LS-Proj. FACE	13.82	27.16	12.81	24.51	11.81	20.47
Adap. LS-Proj4. FACE	15.76	35.58	14.59	32.19	12.82	25.21
Adap. LS-Proj8. FACE	16.56	40.47	15.22	35.74	13.06	26.98
Adap. LS-Proj16. FACE	16.85	44.02	15.26	35.96	13.06	26.98
Adap. LS-Proj32. FACE	16.85	44.02	15.38	37.88	13.15	28.55

Table 8: The hop stretch factor of our projective-based routing algorithms for the radius of 25.

	R = 15		R = 20		R = 25		R = 30		R = 35	
Algorithm	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ
Projective FACE	99.99%	0.10	99.07%	0.97	93.80%	2.19	98.49%	1.14	99.80%	0.45
Projective4 FACE	100.00%	0.00	99.83%	0.38	98.16%	1.27	99.86%	0.37	99.95%	0.22
Projective8 FACE	100.00%	0.00	99.96%	0.20	99.51%	0.64	99.98%	0.14	99.99%	0.10
Projective16 FACE	100.00%	0.00	99.99%	0.10	99.95%	0.22	100.00%	0.00	100.00%	0.00
Projective32 FACE	100.00%	0.00	99.99%	0.10	99.99%	0.10	100.00%	0.00	100.00%	0.00
Adap. LS-Proj. FACE	100.00%	0.00	99.85%	0.38	98.95%	1.04	99.80%	0.45	99.93%	0.29
Adap. LS-Proj4. FACE	100.00%	0.00	99.98%	0.14	99.78%	0.56	99.99%	0.10	100.00%	0.00
Adap. LS-Proj8. FACE	100.00%	0.00	100.00%	0.00	99.97%	0.17	100.00%	0.00	100.00%	0.00
Adap. LS-Proj16. FACE	100.00%	0.00	100.00%	0.00	100.00%	0.00	100.00%	0.00	100.00%	0.00
Adap. LS-Proj32. FACE	100.00%	0.00	100.00%	0.00	100.00%	0.00	100.00%	0.00	100.00%	0.00

Table 9: The delivery Rate of our projective-based routing algorithms for different radii.

	R = 15		R = 20		R = 25		R = 30		R = 35	
Algorithm	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ
Projective FACE	2.40	17.13	11.76	41.42	24.93	54.67	21.83	54.01	12.98	40.60
Projective4 FACE	2.44	17.71	13.88	49.19	36.38	80.04	26.39	67.63	13.73	45.67
Projective8 FACE	2.44	17.71	14.73	55.73	42.33	97.03	27.20	71.70	14.03	48.01
Projective16 FACE	2.44	17.71	15.01	58.08	45.99	113.07	27.38	72.80	14.18	50.46
Projective32 FACE	2.44	17.71	15.01	58.08	46.65	117.81	27.38	72.80	14.18	50.46
Adap. LS-Proj. FACE	1.52	5.66	5.35	16.48	13.82	27.16	13.44	27.82	9.96	23.85
Adap. LS-Proj4. FACE	1.48	3.42	5.76	20.22	15.76	35.58	14.09	32.68	10.31	26.87
Adap. LS-Proj8. FACE	1.48	3.42	5.85	21.35	16.56	40.47	14.17	33.55	10.31	26.87
Adap. LS-Proj16. FACE	1.48	3.42	5.85	21.35	16.85	44.02	14.17	33.55	10.31	26.87
Adap. LS-Proj32. FACE	1.48	3.42	5.85	21.35	16.85	44.02	14.17	33.55	10.31	26.87

Table 10: The hop stretch factor of our projective-based routing algorithms for different radii.

	R = 5		R = 10		R = 15		R = 20		R = 25		R = 30		R = 35	
Algorithm	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ
2D-Compass	99.90%	0.30	91.83%	2.70	73.11%	4.44	92.73%	2.53	99.43%	0.82				
2D-Greedy	99.88%	0.35	91.29%	2.70	71.78%	4.41	91.41%	2.65	99.26%	0.88				
2D-Ellipsoid	98.97%	0.99	83.35%	3.62	56.39%	5.43	76.85%	3.80	91.40%	2.87				
2D-Most Forward	99.84%	0.39	91.52%	2.62	72.48%	4.38	92.17%	2.69	99.32%	0.92				
3D-Compass					98.06%	1.37	82.98%	4.03	63.15%	4.43	83.96%	3.35	96.73%	1.74
3D-Greedy					97.97%	1.47	82.52%	4.03	62.40%	4.21	83.32%	3.45	96.09%	1.97
3D-Ellipsoid					95.37%	2.09	73.42%	4.05	49.57%	4.28	66.18%	4.17	83.09%	3.50
3D-Most Forward					97.75%	1.47	82.41%	4.13	62.34%	4.58	83.16%	3.17	96.30%	1.85

Table 11: The delivery rate for the deterministic routing algorithms in 2-D and 3-D.

	R = 5		R = 10		R = 15		R = 20		R = 25		R = 30		R = 35	
Algorithm	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ
2D-Compass	1.00	0.03	1.02	0.08	1.06	0.11	1.08	0.14	1.08	0.15				
2D-Greedy	1.00	0.01	1.00	0.03	1.01	0.05	1.01	0.06	1.01	0.04				
2D-Ellipsoid	1.00	0.04	1.03	0.10	1.08	0.15	1.12	0.19	1.14	0.22				
2D-Most Forward	1.00	0.02	1.01	0.05	1.02	0.07	1.02	0.08	1.01	0.06				
3D-Compass					1.00	0.03	1.01	0.06	1.03	0.10	1.05	0.12	1.04	0.12
3D-Greedy					1.00	0.02	1.01	0.04	1.02	0.07	1.03	0.09	1.02	0.07
3D-Ellipsoid					1.01	0.05	1.02	0.09	1.05	0.12	1.08	0.16	1.10	0.19
3D-Most Forward					1.00	0.04	1.01	0.06	1.03	0.09	1.04	0.11	1.03	0.10

Table 12: The hop stretch factor for the deterministic routing algorithms in 2-D and 3-D

	R = 5		R = 10		R = 15		R = 20		R = 25		R = 30		R = 35	
Algorithm	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ
2D-Random Compass	99.99%	0.10	98.38%	1.16	88.10%	3.47	98.30%	1.44	99.98%	0.14				
2D-Random Greedy	100.00%	0.00	98.43%	1.21	87.77%	3.52	97.90%	1.45	99.89%	0.34				
2D-Random Ellipsoid	100.00%	0.00	98.43%	1.12	87.72%	2.89	98.00%	1.28	99.92%	0.27				
3D-Random1 Compass					99.69%	0.58	91.18%	2.77	75.23%	4.57	93.51%	2.65	99.30%	0.79
3D-Random1 Greedy					99.73%	0.49	91.23%	2.86	75.14%	4.32	93.66%	2.22	99.32%	0.82
3D-Random1 Ellipsoid					99.70%	0.54	91.10%	2.65	75.22%	3.89	93.30%	2.36	99.29%	0.78
3D-Random2 Compass					99.58%	0.62	91.84%	2.44	77.77%	4.76	93.22%	2.46	99.26%	0.91
3D-Random2 Greedy					99.59%	0.62	91.71%	2.70	77.34%	4.78	92.32%	2.55	99.04%	0.99
3D-Random2 Ellipsoid					99.59%	0.60	90.98%	2.46	76.96%	4.45	92.44%	2.47	99.11%	0.87

Table 13: The delivery rate for the randomized routing algorithms in 2-D and 3-D.

	R = 5		R = 10		R = 15		R = 20		R = 25		R = 30		R = 35	
Algorithm	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ	Avg	σ
2D-Random Compass	1.12	0.66	2.09	2.20	2.70	2.44	1.94	1.75	1.36	0.88				
2D-Random Greedy	1.11	0.61	2.03	2.11	2.62	2.40	1.83	1.62	1.28	0.75				
2D-Random Ellipsoid	1.12	0.60	2.02	2.03	2.62	2.33	1.90	1.67	1.34	0.77				
3D-Random1 Compass					1.68	1.90	3.25	3.55	4.07	3.70	3.63	3.45	2.37	2.22
3D-Random1 Greedy					1.65	1.84	3.20	3.48	4.08	3.73	3.59	3.48	2.36	2.23
3D-Random1 Ellipsoid					1.66	1.86	3.13	3.40	4.14	3.76	3.65	3.48	2.39	2.28
3D-Random2 Compass					1.63	1.71	2.56	2.67	2.66	2.36	2.03	1.80	1.46	1.02
3D-Random2 Greedy					1.63	1.73	2.47	2.51	2.60	2.32	1.97	1.73	1.43	0.97
3D-Random2 Ellipsoid					1.67	1.74	2.58	2.58	2.77	2.38	2.19	1.80	1.69	1.16

Table 14: The hop stretch factor for the randomized routing algorithms in 2-D and 3-D.