A Survey on 3D Position-based Routing Algorithms for DANETS

Claudio Enrico Palazzi, *Member, IEEE,* Armir Bujari, and Daniele Ronzani,

***Abstract*—The abstract goes here.**

***Index Terms*—Wireless Communications, Routing 3D, Algorithm, MANET, Mobility, Forwarding.**

I. Introduction

- A brief introduction of evolution of wireless communications.
- Explain the coming of UAVs (drones), their network infrastructure and possible applications.
- Explain the issues of drone networks and deepen the problem of routing.
- Aims of this survey
- Illustrate the growth of ad-hoc network, due to the new frontier of the drone networks, that keeps growing. - Sections of this document.

Typical applications of MANETs range in military, civilian, rescue operations scenarios and more. A subclass of MANET is Wireless Sensor Network (WSN) that differs from the previous for several aspects, such as energy constraint, degree of mobility, type of devices and communication.

II. Routing in Mobile Ad-hoc Networks

A Mobile Ad-hoc Network (MANET) consists of a set of mobile nodes that communicate with each other over wireless links, capable of operating without centralized control or established infratructure. In this type of networks a *multi-hop* routing is needed, since a node needing to send a packet to another node can be not communicate directly with it. Since MANETs may change their topology frequently, routing is difficult.

- Classification of routing protocols into topologic and geographic.
- A summary of topologic routing protocols (AODV, DSDV, DSR, etc).
- Issues and disadvantages of topologic protocols.
- Short description and advantages of geographic-position based protocols, with reference of scalability.

A. Position-based Routing Protocols

- Explain the mechanism of position-based protocols (use fo GPS).

Name
name2

- Demonstrate the real gain using geographic position (local algorithms, no routing table, etc).
- Explain how the nodes known the neighbors position, and hint the difficulty of this (inconsistency of records).
- Put some example of routing protocol.

1) Taxonomy: In the years there was make several taxonomies to classificate position-based routing protocols. In this survey we classificate these according to two types of strategies: path strategy and forwarding strategy.

- *Path Strategy* represents how a packet traverses a network. The packet can uses either a single path to reach the destination, or multiple paths. Single path strategy requires that each node forwards the packet to only one of its neighbors. There is one copy of a packet in the whole network. Multipath strategy allows a node to forward multiple copies of the same packet to several neighbors. This strategy is also defined as *flooding*.
- *Forwarding Strategy* describes the forwarding criteria used to send a packet from the current node to a neighbor node in a single step. Every strategy exploits different methods and geometric models. The three main forwarding strategies are *deterministic progress-based*, *randomized progress-based* and *face-based*.

B. Routing in 3D Networks

- Explain the nature of MANET to become 3D
- Mention the fact that many papers have treated only 2d contexts
- Describe the difficults of 3D routing, and which 2D protocols have difficulty to be extended in 3D scenario.

III. Performance Metrics

In this section, we introduce the metrics of interest to evaluate the performances of the routing protocols described in this survey.

A. Delivery Rate

Delivery rate is the ratio of the number of packets received by the recipient (or recipients) to the number of packets sent by the source (or sources). The primary objective of a routing algorithm is to guarantee the delivery of all the packets, that this means a 100% delivery rate.

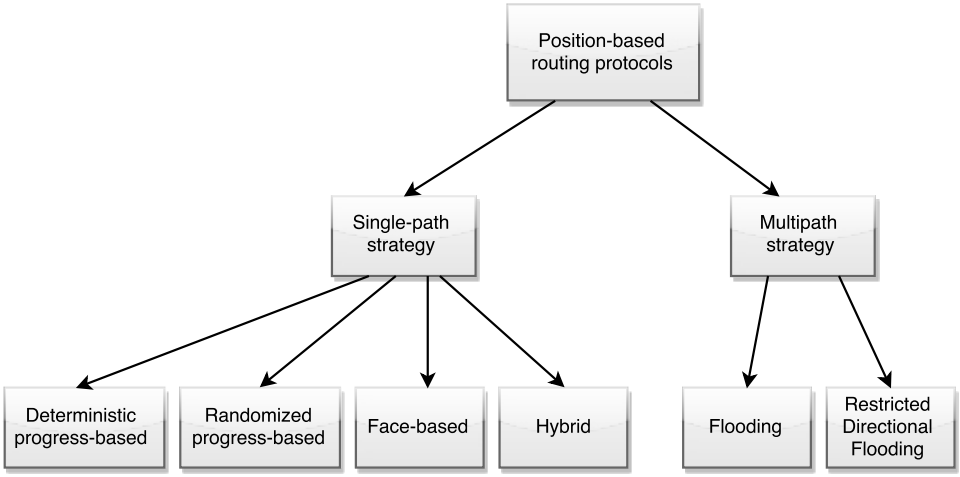


Fig. 1. Taxonomy of 3D routing protocols.

B. Path Dilation

Path dilation or *Stretch factor* is the ratio of the number of hops traversed by the packet to the number of hops of the minimum path.

IV. State of the Art

In this section, we explain the state-of-the-art of the position-based routing protocols in 3D MANET, in particular describe the forwarding algorithms employed in the routing protocols. The extension of progress-based algorithms (the first four) in 3D networks, proposed in [1], is relatively simple, as we need only the definition of the Euclidean distance between two points u and v in three dimensions and the definition of the sphere with center c and radius r.

A. 3D Greedy

Greedy is a simply progress-based forwarding strategy. The current node forwards the packet to the neighbor node that minimizes the distance to the destination node. With *Greedy*, the current node's distance does not compare against the distances of its neighbors, but it is used to select the forward nodes and the backward nodes: if there is no any neighbor node closer to the destination node than the current node, the algorithm fails. More precisely, there are two forwarding algorithms that use greedy strategy, and differ on which neighbors consider on the choice:

- *Greedy* [2]: the current node c forwards the packet to one of its neighbors that is closer to d than c and any other neighbors.
- *GEDIR* [3]: the current node c forwards the packet to one of its neighbors that is closer to d than any other neighbors, but not necessarily closer to d than node c itself.

In *GEDIR*, all neighbors are considered. So, also the nodes that are in backward direction can be chosen and the only kind of loop that may be formed using this algorithm is the local loop between c and the node $prev$ that sent the message to c in the previous step [3], but this case is avoided by the assumption mentioned follow. If the next node for forwarding is the node that sent message to c, it means that the packet has reached a local minimum c and then the algorithm fails. Instead, in *Greedy*, only the neighbors that are closer to the destination than c are considered. If no one is closer to d, algorithm fails. Fig. 2 shows an example of choosing of next node using progress-based algorithms. The current node c has five nodes that are in the direction of destination node d inside its range, but with *Greedy* the node that is closer to the destination is 2 (the closer to the dotted arc). In this survey we refer to *Greedy* algorithm and specify when we consider *GEDIR* (for example in *Random Walk*).

B. 3D Compass

Compass (or *Directional, DIR*) algorithm [4] uses the direction of nodes to selects the best forwarding node. The current node c uses the location information of d to calculate its direction. Then, forwards the packet to the neighbor node u such that the direction cu is closest to the direction cd, that is the neighbor node u that minimizes the angle between c and d, $min\{\angle u_1cd, ..., \angle u_ncd\}$. In Fig. 2 node c chooses node 6 as the next node, since the angle $\angle 6cd$ is the smallest among all. *Compass* is not a loop-free algorithm.

C. 3D Most Forward

Most Forward, a.k.a *MFR* algorithm [5] is very similar to *Greedy*, but, in this case, the current node c forwards the packet to the neighbor node u whose projection on the line between c and d is closer to d, $min\{d(u'_1, d), ..., dist(u'_n, d)\}$. If the packet reaches a local minimum, the algorithm fails. In most cases *MFR* choose the same path of *Greedy*. In Fig. 2 node c selects node 5 as next node, since the latter has the smallest projected distance to d, on the line sd. *Most Forward* is a *loop-free* algorithm.

D. 3D Ellipsoid

In *Ellipsoid* algorithm [6], the current node c forwards the packet to the neighbor node u that minimizes the

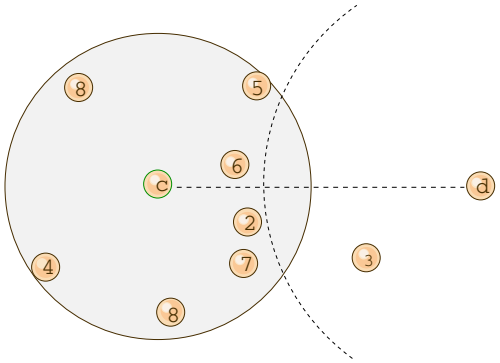


Fig. 2. Illustration of several next nodes chosen by c using the progress-based forwarding strategies.

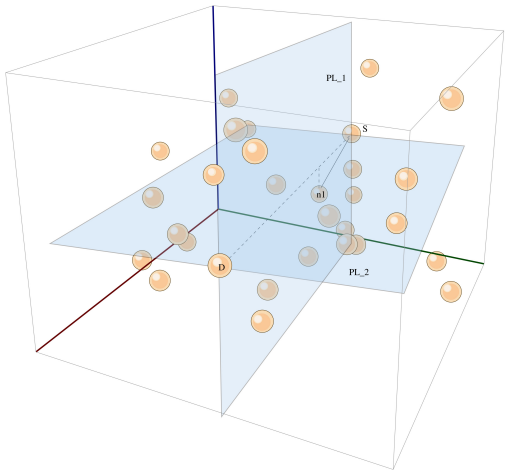


Fig. 3. In AB3D, plane PL_1 passes through s, d and n_1, plane PL_2 is orthogonal to PL_1. Both planes contain the line sd.

sum of the distance from node c to u and the distance from node u and the destination node d, $min\{dist(u_1, c) + dist(u_1, d), ..., dist(u_n, c) + d(u_n, d)\}$. If the packet reaches a local minimum, the algorithm fails. In Fig. 2 the next node chosen is 6, using *Ellipsoid*.

E. 3D Random Walk

Random Walk algorithm try to solve the local minimum problem described in previous section, by choosing the next node randomly from a subset of the current node's neighbors that makes progress to the destination. Already since 1984, some authors have tried to put the concepts of randomization algorithms in order to avoid the emergence of loops [7], [8], [9]. Tipically, randomization performs as follow: let $cw(u)$ be the neighbor node in $N(c)$, over the line cd, that minimizes one of the progress values defined in previous progress-based algorithms (distance, angle, etc.) and let $ccw(u)$ be the neighbor node in $N(c)$, under the line cd, that minimizes the same one; then, randomized algorithms moves the packet to one of $\{cw(u), cww(u)\}$ with equal probability or a probability weighted according to the progress values. Routing algorithms that use randomization are usually considered to fail when the number of hops in the path computed so far exceeds a *threshold* value, here called $TTLR$ (*Time To Live Random*).
Extension of randomized algorithm in 3D is not trivial, because it is not obvious to choose the best way to determine candidate neighbors. The reason is that in a three-dimensional graph there is no concept of above and below a line passing from source to destination. Therefore, in [10] authors propose an extension for the randomized-based algorithm from 2D to 3D, that uses the concept of 3D planes. This new algorithm is called *AB3D*. In this document we generalize the concept of randomized-based algorithm defining a parametric function that has four attributes, that are:

- m: are possible candidate neighbors to choose from $N(c)$.
- R: is the name of progress-based strategy used to choose the m candidates, and it is one of R (Random), CM (Compass), GR (Greedy) or MF (most Forward), as in *AB3D*.
- S: is used to represent the probability weighting when randomly choosing, and it is one of U (Uniform), D (Distance) , A (Angle), PD (Projection Distance). Probability weightings are defined as in *AB3D*.
- ab (above-below): is a boolean flag indicating whether to define the candidates over and below the Plane PL_1 (or even over and below the plane PL_2, if $m = 5$), or select candidates without considering the planes.

The steps of the algorithm are the following. The current node c selects a node n_1 from its neighbors, chosen according to the method defined in R (CM, GR or MF). With $ab = YES$, if m is 3, we define the plane PL_1 identified by the three nodes s, d and n_1. If m is 5, define also the plane PL_2 that is perpendicular to PL_1 and passes through c and d, such that the intersection line between the two planes is the line cd. Fig. 3 shows an example of network graph and its subdivision with the two planes. Then, if the parameter m was 3, *Random Walk* select another two nodes, in addition to n_1. One neighbor n_2 is chosen from the above of the plane PL_1 according to R and one neighbor n_3 is chosen from the below of the plane PL_1 according to R. If m was 5, in addition to n_1, the algorithm choose from $N(c)$ four neighbors n_2, n_3, n_4, n_5 each one in one side of the four regions that result from the intersection between PL_1 and PL_2. Once these candidates are determined, node c selects one of these nodes randomly, according to the probability weighting determined by S, and forwards the packet to the chosen node. Instead, with $ab = NO$, the m candidates are chosen without a plane subdivision.

F. Greedy-RW-Greedy

Greedy-Random-Greedy (*GRG*) algorithm belongs to *progress-based/randomized-based* class, and uses *Greedy* as the primary stage and a randomized algorithm, such as *Random Walk*, as a recovery strategy. The general procedure of this algorithm is the following: the algorithm starts with proceeding the greedy phase until it find a local minimum c. At this point, *GRG* stores the distance $dist(c, d)$, and switches to random phase, as recovery strategy, where the node c randomly selects one u of its neighboring nodes, using the steps defined in *Random Walk*. If $dist(u, d) < dist(c, d)$,

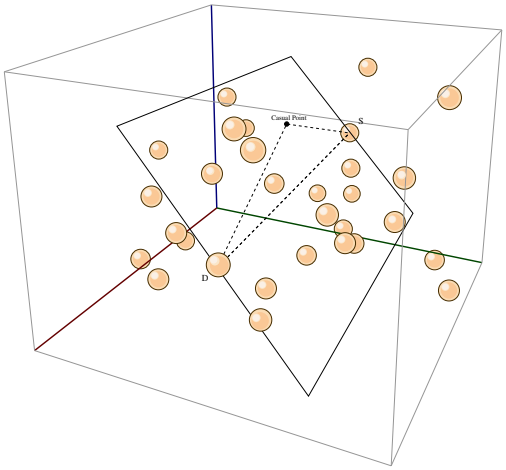


Fig. 4. Computing of a plane with *Projective face* algorithm

then the algorithm resumes the greedy forwarding, otherwise continues with *Random Walk*.

G. Projective Face

The first extension of face-based strategy in 3D space provides using two orthogonal planes intersected to the line connecting source and destination. In [1] authors propose *Projective Face*, that proceed as follows. The points of the networks are firstly projected onto one plane that contains the line sd, with third point chosen randomly. *Face* algorithm is performed on this projected graph. If the routing fails, the points are then projected onto the second plane, that is orthogonal to the first plane and also contains the line sd. *Face* algorithm is again performed. Fig. 4 shows an example of planes configuration in *Projective Face*. Note that, in this case (and in all the following algorithms), since the delivery rate is not guaranteed, the algorithm needs a *local threshold* value, $TTLF$ (*Time To Live Face*), in order to terminate the algorithm in case it not reached the destination. This is necessary because the algorithm can get stuck in a loop. More precisely, in this version of algorithm, $TTLF$ counter is started twice, once for the first plane and one for the orthogonal plane, obtaining a *global threshold* value ,TTL, at most $2 * TTLF$.

H. CFace(3)

CoordinateFace(3) (*CFace(3)*), proposed in [10], uses another set of projection planes, that is the planes xy, xz and yz for the projection of the nodes. *CFace(3)* works as follow. All nodes are projected on the first xy plane ($node.z = 0$) and then the face routing is started on the projected graph. If the packet does not arrive at the destination ($TTLF$ has expired) , original coordinate of all nodes are projected in xz plane ($node.y = 0$) and face routing is again performed. If again the packet does not arrive to the destination, original coordinate of all nodes are projected on yz plane ($node.x = 0$) and face routing is again performed. If the packet does not arrive with this last plane, algorithm fails. Fig. 5 shows all the three projections on the coordinate planes. Note that, in this algorithm, TTL is at most $3 * TTLF$. As we will see, the delivery rate is typically

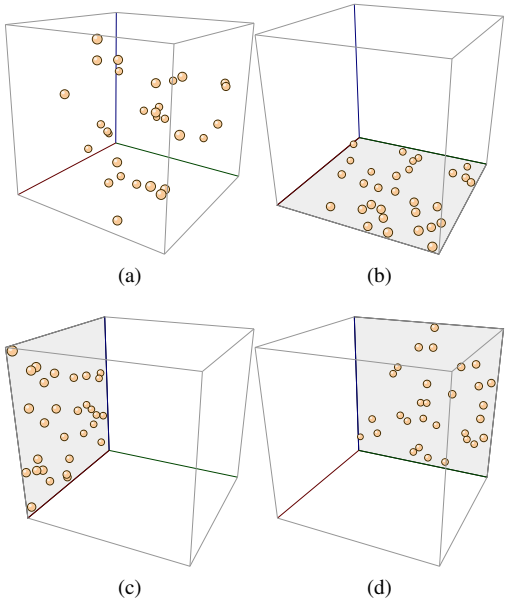


Fig. 5. Projection of graph nodes (a) on the three planes xy (b), xz (c) and yz (d) in *CFace(3)* algorithm.

greater greater than that of the *Projective face*, but with a greater path dilation.

I. Adaptive Least-Square Projective Face

Authors of [11] proposes three heuristics to modify and improve *Projective Face* algorithm. The new obtained algorithm is called *Adaptive Least-Squares Projective Face*, (*ALSP Face*). The three heuristics are:

- Least-Squares Projection (LSP) Plane;
- Adaptive Behavior Scale (ABS);
- Multi-Projection-Plane Strategy.

In the *Projective Face*, a third point is chosen randomly, together with the source and the destination points, to compute the first projection plane. Instead, *ALSP Face* choose the third point adopting a mathematical optimization technique (first heuristic) for finding the best fitting plane to the set of neighbor nodes: using the current node, its neighbors up to two hops away, and the destination node, the initial projection plane is determined by using *least-squares error minimization* of the distance of the nodes to the plane, minimizing the sum

$$\sum_{i=1}^{m} (r_i)^2$$

where r_i is the Euclidean distance from a point i to its perpendicularly projected point in the least-squares projection plane (*LSP* plane), as seen in Fig. **??**. To maintain the local characteristic of the routing algorithm, authors propose that only the source, destination and the neighboring nodes within the 2-hops scope of the current node be selected as the set of points for computing the least projection plane [1] Then, nodes are projected to this plane and *Face* routing is performed. This

[1]for simplicity of implementation, in this work the 1-hop scope of current node was used for *LSP* plane computing

LSP plane aim to have a less distorted projected graph so that the number of crossing edges can potentially be reduces. The second heuristic defines a parameter called *Adaptive Behavior Scale* (*ABS*) that is used to determine when recalculate the *LSP* plane, in order to ensure that the plan is always appropriate for the current node. Third heuristic uses a set of N_s projection planes arranged in a fixed order about an axis. The algorithm switches between these planes, following the order, to disrupt any looping that may occur during routing. In [11] it is said that by performing the face routing on the additional projection plane, there is a significant increase in the delivery rate. Therefore, third heuristic tries to increase the number of projection planes. But this reasoning is true only up to a certain point: if it is true that the delivery rate is slightly increased, it is also true that the path followed by the packet becomes enormously long, because, for each projection plane, the threshold value (here, $TTLF$) is reset to the its pre-set value. Such a high traffic for only one packet might not be acceptable in real world. Then it is not clear So, this thesis consider the third heuristic using only two additional planes, chosen as in *Projective Face*.

J. Greedy-Face-Greedy

Greedy-Face-Greedy algorithm (*GFG*), also defined as *Greedy Perimeter Stateless Routing* algorithm (*GPSR*) for 2D networks, uses a combination of greedy method with face method. With *GFG*, a flag is stored in data packet. This flag can be set into greedy-mode and face-mode (or perimeter-mode), indicating whether the packet is forwarded with *Greedy* or *Face* algorithm. The algorithm starts from s with *Greedy*, setting the packet into greedy-mode and forwarding it. A node that receives a greedy-mode packet, searches from its neighbors the node that is closest to the destination. If this neighbor exists, the current node forwards the packet to this, otherwise marks the packet into perimeter-mode. *GFG* algorithm forwards the perimeter-mode packets performing the same planar graph as the *Face2* algorithm. Moreover, when a packet enters in perimeter-mode at node x, *GFG* set the line xd as a reference line to the crossing with the faces and records in the packet the location of x as the node when greedy forwarding mode failed. This information is used at next hops to determine whether and when the packet can be returned to greedy-mode: upon receiving a perimeter-mode packet, the current node c first compare the location of x stored in the packet with the forwarding node's location. *GFG* returns in greedy-mode if the distance from the forwarding node to d is less than the distance from x to d. In this case the packet is set into greedy-mode and algorithm continue greedy progress towards the destination. Fig. **??** shows an example, where the packet travels nodes 9, 6 and c in greedy-mode (the green path), stopping at node c, that is the local minimum. Then, from node c the face-mode is started, and forwards the packet on progressively closer faces of the planar graph, each of which is crossed by the line cd. The packet reaches nodes 7, 8 and 5 (the blue path). Node 5 is closest to the destination d than c, so the packet can be returned to greedy-mode and reaches destination d (the second green path).

K. RW-CFace(1)-RW

This algorithm, initially conceived in [10] as *AB3D-CFace(1)-AB3D*, starts with *RW* algorithm. Once the local threshold $TTLR$ is passed and the algorithm reaches a local minimum, it switches to *CFace(1)*. *CFace(1)* traverses one projective plane, which is chosen randomly from the xy, yz, or xz planes, starting from the node in which the algorithm is switched. At this point, $TTLF$ is initialized to 0 and *CFace(1)* restarts. If the destination is not reached during this phase and $TTLF$ is passed (a looping occurs), the algorithm goes back to *RW* and $TTLR$ restarts at 0.

L. RW-CFace(3)

This algorithm starts as *RW-CFace(1)-RW*, but instead of going back to *RW* if the phase in a projective plane fails, *RW-CFace(3)* tries the other two projective planes, define as in *CFace(3)*. This algorithm starts with *RW* stage and if the destination is not reached and $TTLR$ is passed, the algorithm switches to *CFace(3)* using the first xy plane. Again if a loop occurred ($TTLF$ is reached), it switch to yz plane, and finally the same process is make for xz plane.

M. LAR 3D

In the extension of *LAR* algorithm in 3D space, with three-dimensional coordinates of source and destination positions, the source node computes the expected zone for d, which is a sphere around d of radius equal to $v_max * (t_1 - t_0)$ where t_1 is the current time, t_0 is the time stamp of the position information that c has about d, and v_max is the maximum speed of the node in the network. This zone is used to define the 3D flooding area, which is the minimum size rectangular box with $ball(s, r)$ in one corner and the expected zone in the opposite corner. In our case, since we analyze static scenarios ($v_max = 0$), the expected zone is $ball(d, r)$.

N. RW-LAR

RW-LAR was proposed in [12] and hybridize *Random Walk* with *LAR*. This algorithm try to reduce the high path dilation of *LAR* algorithm. All the combinations in *RW-LAR* use the same partitions as in *Random Walk* algorithm. The difference is that, while *RW* select only one of the m candidates chosen from the neighborhood, *RW-LAR* sends the packet to all those selected candidates which are within the rectangular box defined as in *LAR*.

V. Performance Evaluation

In this section we provide a performance comparison of the above considered routing protocols. A 3D simulation environment is structured for the experiment and a common set of configurations is provided.

A. Simulation Environment

Our simulation scenario consists of n nodes placed in a random positions into a 3D cubic space. The cube has 500 meters of side length, which can be considered as a real typical MANET area size. Every scenario istance represents a connected networks, which means that each node can reach each other one. The node transmission range is set to 100 meters. To simplify the simulation, we assume that the nodes are not moving in the space, obtaining a static ad-hoc network. The motivation of this choice is twofold: (a) the need to consider initial essential parameters for the comparison (i.e. the minimum path) that would change if the nodes move, and (b) the assumption that, with the considered routing procotols, the route of each packet is traveled in a very little time, during which the nodes of the networks have a (very little movement? NON MI PIACE). First comparison aims to get a comprehensive look at the behavior of the forwarding algorithms under dinamyc number of nodes, ranging in {50, 100, 150, 200}. Second comparison analyzes the effect of varying threshold values *TTLR* and *TTLF* of respectively randomized-based and face-based forwarding algorithms. Third comparison applies all the forwarding algorithms into classes of graphs in which the length of the shortest path from source to destination ranges in {1-3, 4-6, 7-9, 10+} (the notation *a-b* indicates that the shortest path length in a-b class graph can assume values from *a* to *b* including).

B. Results

Now we discuss the outcome of the experimentation and analyze the protocols performances on different settings.

1) Comparison with dynamic number of nodes: First comparison performs considered routing protocols in a set of network istances consisting of 50, 100, 150, 200 hosts. The results are shown in Figs. 10 and . We see immediately that, from the results in Fig. 10, deterministic progress-based forwarding algorithms have the lowest delivery rate, less than 60% in the case of 150 nodes. This means that in this scenario (150 nodes) there is more chance of finding local minima. Instead, in the case of 200 nodes, we can see an increment of the delivery rate. This suggests a reduction of local minima with a growth of nodes' number starting from 150, in a cube of 500 meters of side length. *Random Walk* and *Greedy Random Greedy* algorithms obtain better results in delivery rate, with a peak in the case of 50 nodes (90%), as seen in Fig. 10a. Face-based routing algorithms (*Projective Face*, *CFace(3)*, and *ALSP Face*) have generally better results than the previous ones, in terms of delivery rate. This overcoming depends on the fact that these algoriths have more possibility to find the destination node, since the *TTLF* resetting. *CFace(3)* can be defined as *Projective Face*, only choosing two plans that are *xy* plane and *xz* plane, and with the addition of the *xz* plane. So, since the algorithm have three chances to find the destination instead of two as in *Projective Face*, the delivery rate is a little higher. The hybrid algorithm, *Greedy Face Greedy*, reaches better results compared to *ALSP Face*, regarding delivery rate. This is caused probably by the greedy method phase that, at first, closes the packet to the destination and then, if a local minimum is found, the face method phase starts, but with less search space. *LAR* and *RW-LAR* are partial flooding-based algorithms. *LAR* has relevant traffic and does not show a very high delivery rate. Not even *RW-LAR* presents a good delivery value, but has almost half of path dilation than *LAR*. From what we can see in Figs. **??**, the transition from 150 nodes networks to 200 nodes ones, brings an increase of delivery rate, due to the increase of the node density and the restricion area includes many more nodes.

2) Comparison with dynamic threshold values: This performance test has the purpose of assessing the effect of the threshold values respectively for randomized-based and face-based algorithms, that is to see how effective is to increase the *TTLR* and *TTLF* values to increase delivery and path performance. In this test *TTLR* and *TTLF* are dynamic, while the number of nodes n is not. Therefore, to get a fair treatment of the various instances and to allow each algorithm reach all their thresholds values (i.e., *CFace(3)* fails after three times *TTLF*), the *TTL* value is calculated equal to *2 * TTLR* for randomized case and to *3 * TTLF* for face case.

3) Comparison with dynamic min path length: This section shows the results relating to the application of all the routing algorithms considered in classes of graphs in which the length of the shortest path source-destination is respectively 1-3, 4-6, 7-9, and >10 hops. These results can be seen in Figs. 5.11, 5.12, 5.13, 5.14. Note that for the first class (1-3 hops), all the algorithms have a high delivery rate and a small path dilation. This happens because there is very little chance that in a so few hop distance there are holes (local minima). When it start to move to longer paths, deterministic progress-based strategies are the first to degenerate, because, as the destination is more distant, there is a greater probability to finding holes (localm minima). As seen in Fig. 5.14a, progress-based algorithms reach about 10% of delivery rate. RW and GRG algorithms perform well up to 4-6 length of minimum path, but from this point the delivery rate decreases. This is because the increasing of number of link to choose from source to destination reduce the probability of choosing the righ path. However, the path length of the delivered packet remain short (see Fig. 5.14b, RW and GRG histogram). Performance of face-based and hybridized algorithms are also good for high lengths, due to the fact that they succeed in reach the destination within the T T LF or T T L, despite the increasing of the distance from source to destination. However, the length of path performed and delivery time are too high, due to an increase in the number of crossing links during the travel. [1]

4) Comparison with dynamic z coordinate:

VI. Conclusion

The conclusion goes here.

Appendix A
Appendix 1

Appendix one text goes here.

REFERENCES

[1] J. O. G. Kao, T. Fevens, "Position-based routing on 3d geometric graphs in mobile ad hoc networks," pp. 88–91, 2005.
[2] G. G. Finn, "Routing and addressing problems in large metropolitan-scale internetworks," 1987.
[3] X. L. I. Stojmenovic, "Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks," vol. 12 (10), pp. 1023–1032, 2001.
[4] H. S. E. Kranakis and J. Urrutia, "Compass routing on geometric networks," pp. 51–54, 1999.
[5] L. K. H. Takagi, "Optimal transmission ranges for randomly distributed packet radio terminals," vol. 32 (3), pp. 246–257, 1984.
[6] K. S. K. Yamazaki, "The proposal of geographical routing protocols for location-aware services," vol. 87(4), 2004.
[7] L. K. R. Nelson, "The spatial capacity of a slotted aloha multihop packet radio network with capture," vol. COM-32, pp. 684–694, 1984.
[8] P. M. P. Bose, "Online routing in triangulations," pp. 113–122, 1999.
[9] B. N. B. T. Fevens, A. E. Abdallah, "Randomized ab-face-ab routing algorithms in mobile ad hoc networks," 2005.
[10] A. Abdallah, T. Fevens, and J. Opatrny, "Randomized 3d position-based routing algorithms for ad-hoc networks," pp. 1–8, 2006.
[11] J. O. G. Kao, T. Fevens, "3d localized position-based routing with nearly certain delivery in mobile ad-hoc networks," 2007.
[12] J. O. A.E Abdallah, T. Fevens, "High delivery rate position-based routing algorithms for 3d ad-hoc networks," vol. 31 (4), pp. 807–817, 2007.

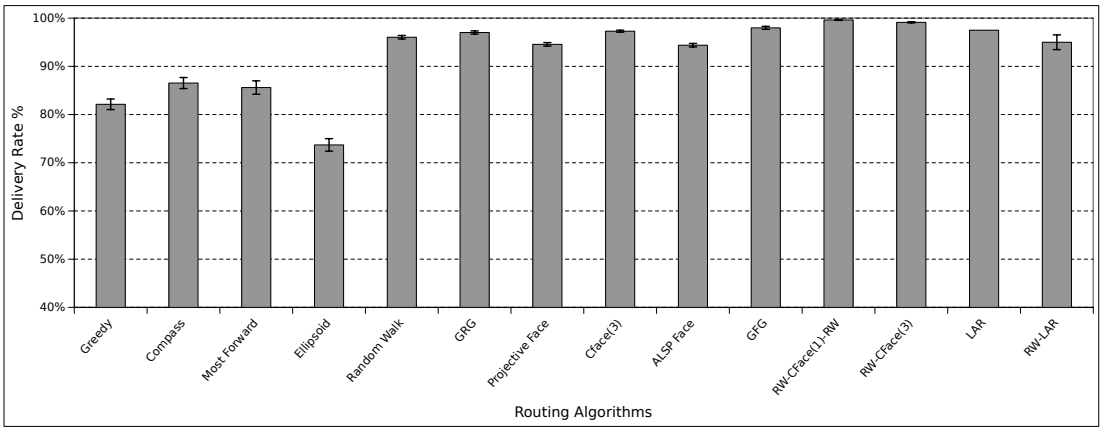
REFERENCES

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

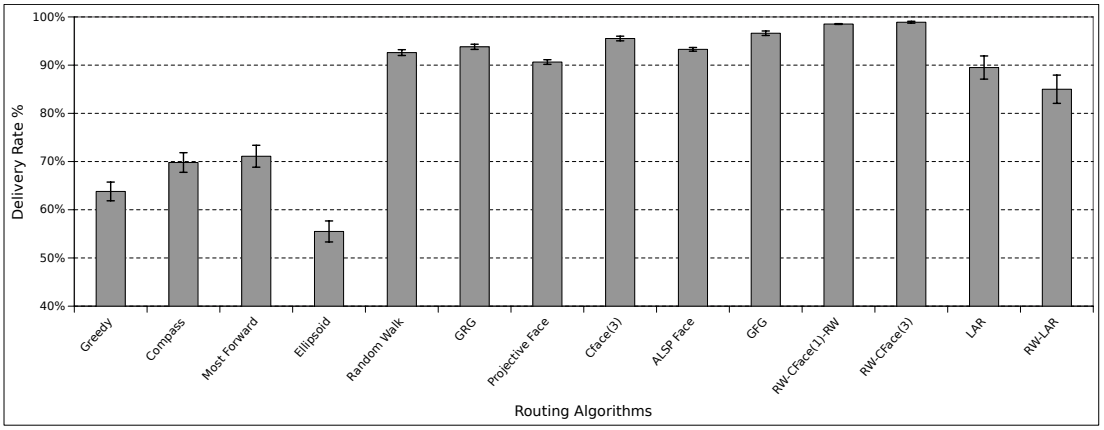
Claudio Enrico Palazzi Biography text here.

Armir Bujari Biography text here.

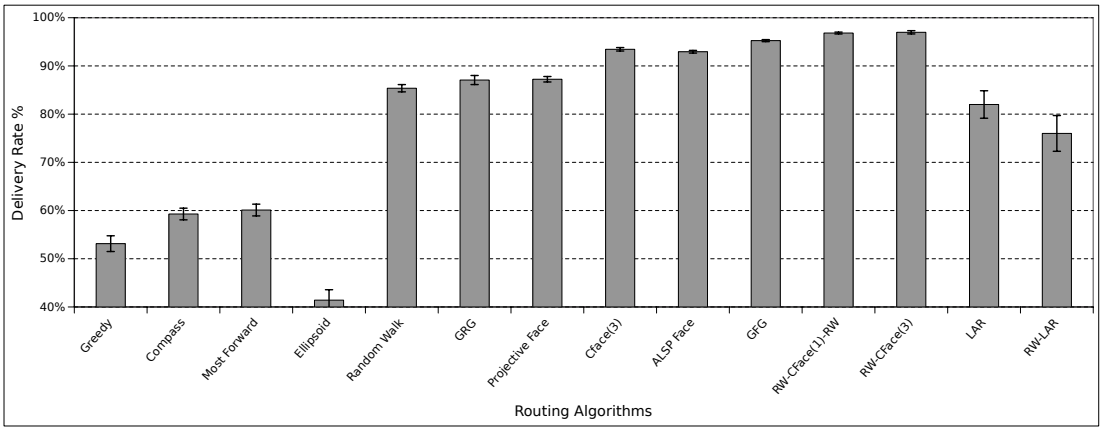
Daniele Ronzani Biography text here.



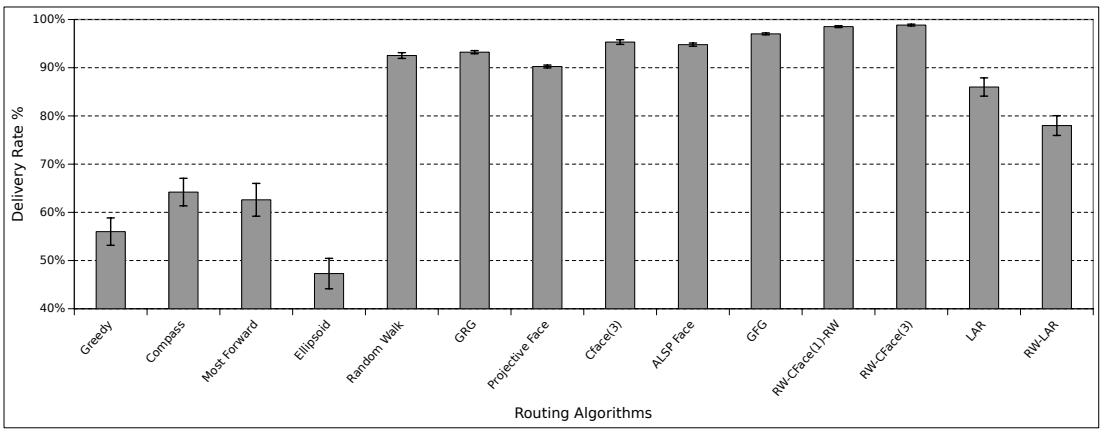
(a)



(b)

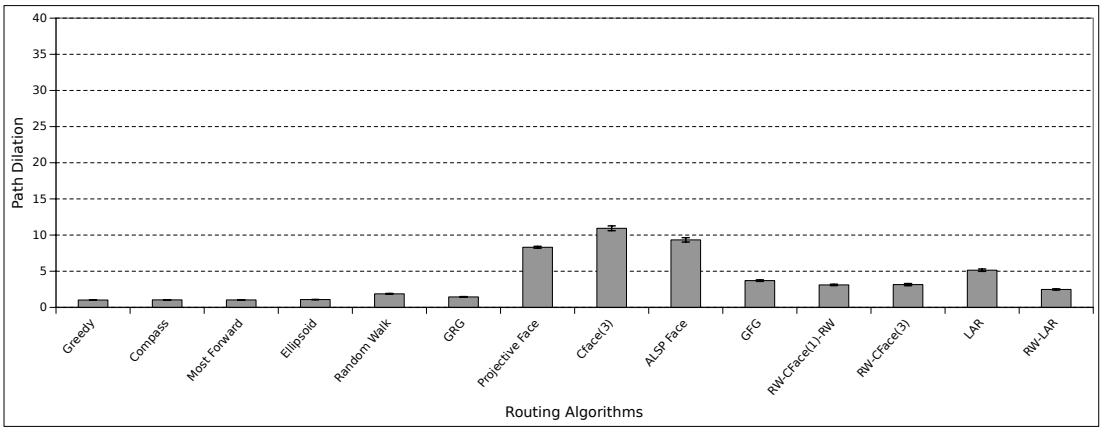


(c)

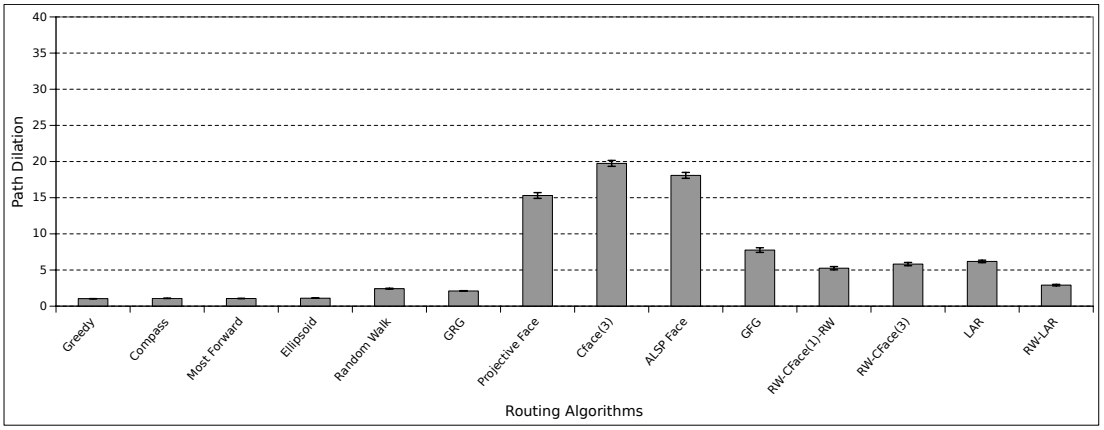


(d)

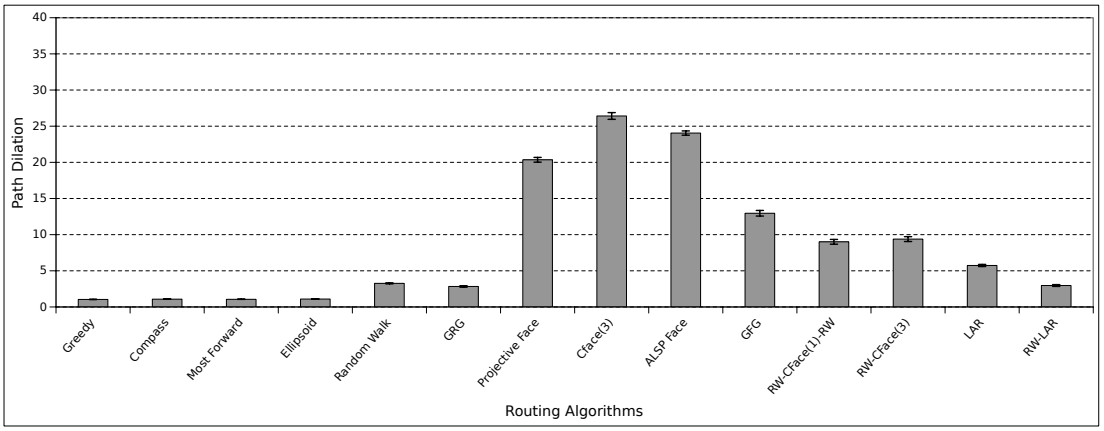
Fig. 6. Results of delivery rate



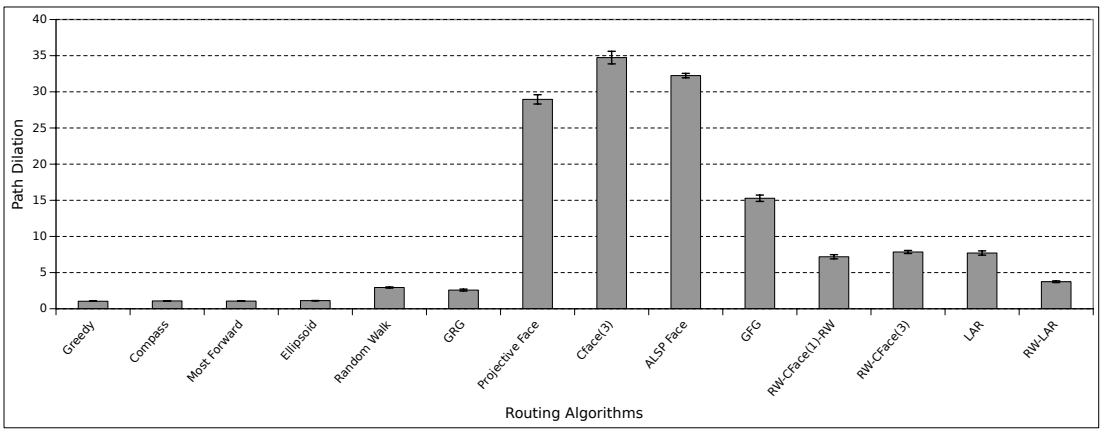
(a)



(b)

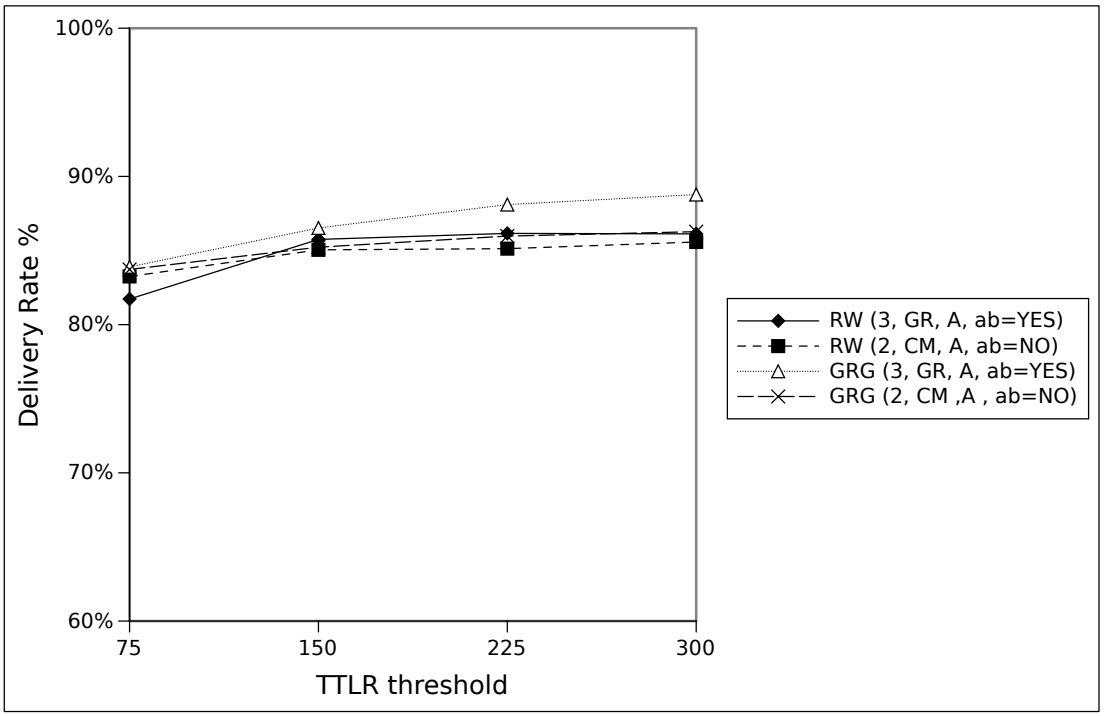


(c)

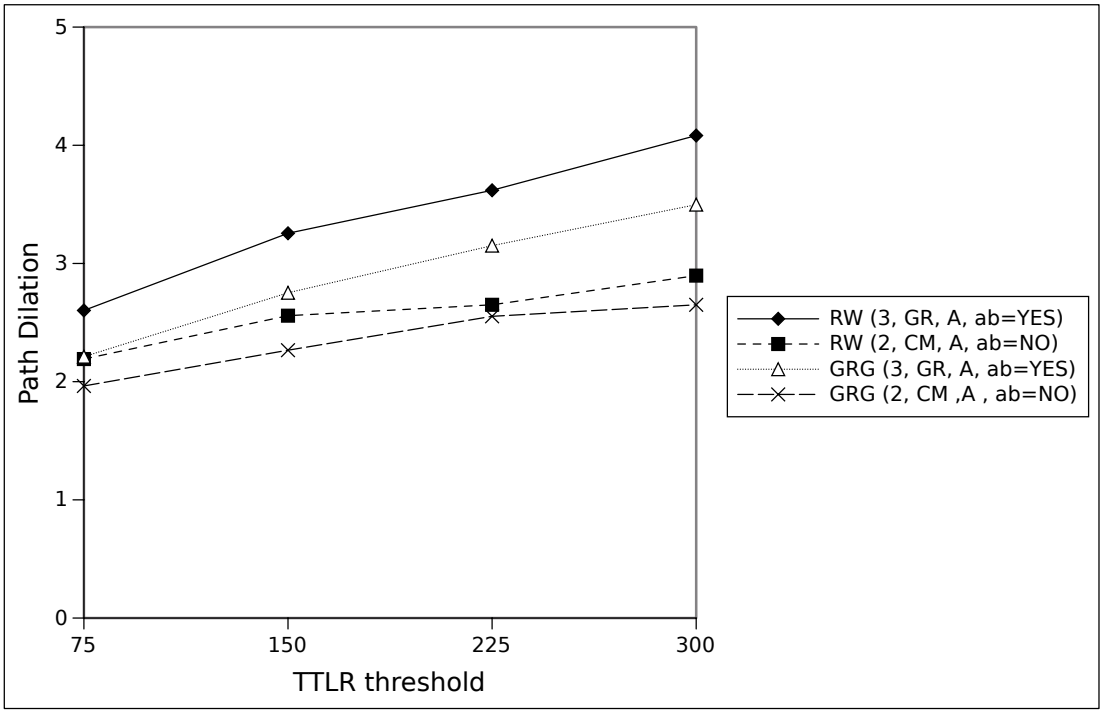


(d)

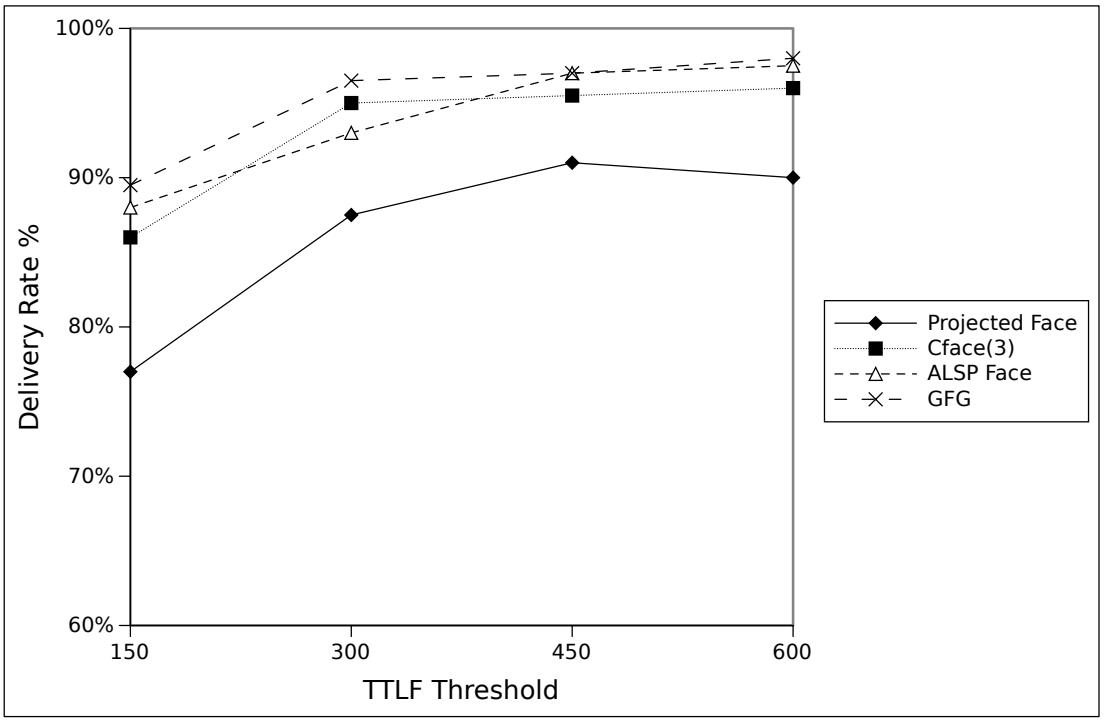
Fig. 7. Results of path dilation


100%
90%
80%
70%
60%
Delivery Rate %
75
150
225
300
TTLR threshold
RW (3, GR, A, ab=YES)
RW (2, CM, A, ab=NO)
GRG (3, GR, A, ab=YES)
GRG (2, CM ,A , ab=NO)

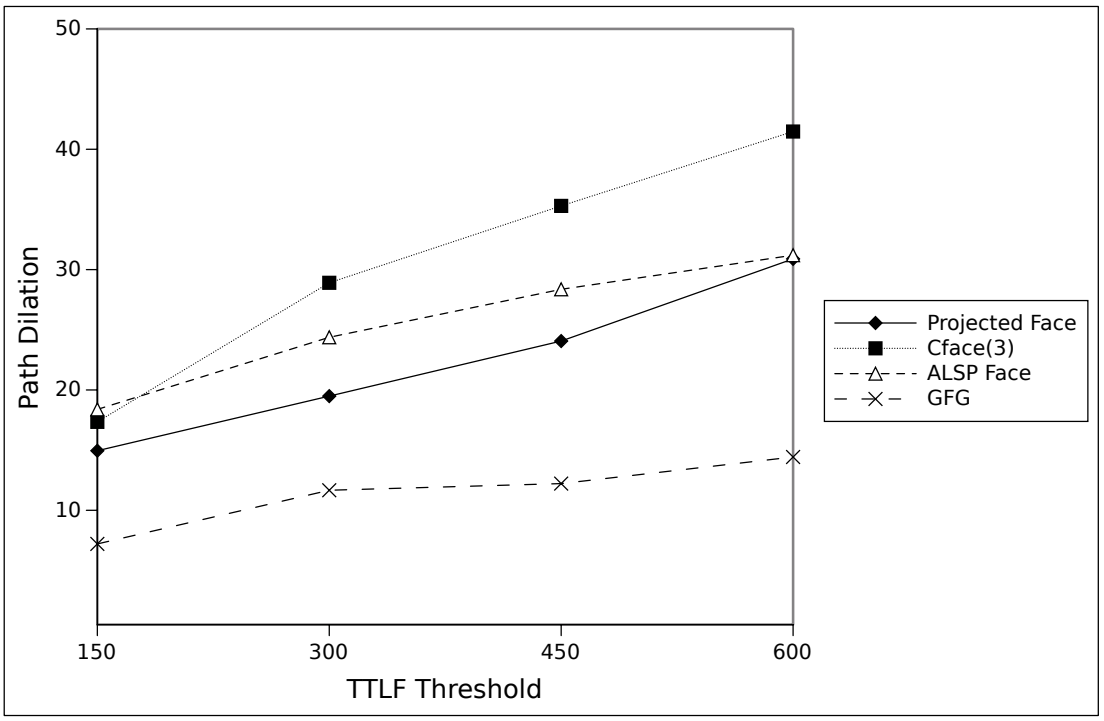
(a)


5
4
3
2
1
0
Path Dilation
75
150
225
300
TTLR threshold
RW (3, GR, A, ab=YES)
RW (2, CM, A, ab=NO)
GRG (3, GR, A, ab=YES)
GRG (2, CM ,A , ab=NO)

(b)

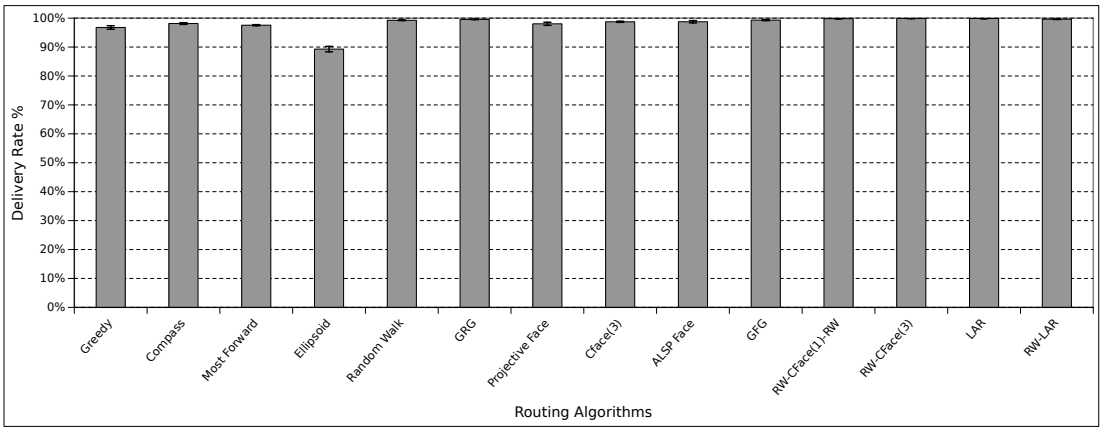


(a)

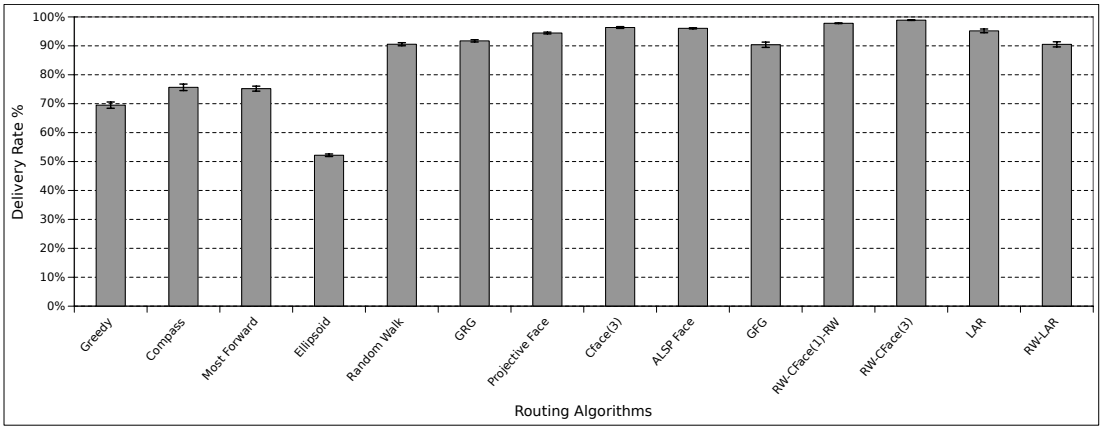


(b)

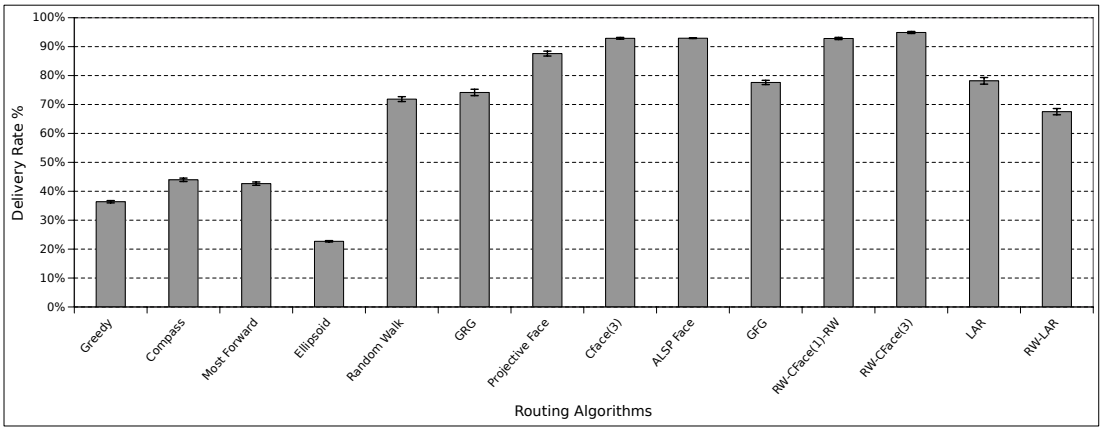
Fig. 8. Results of path dilation



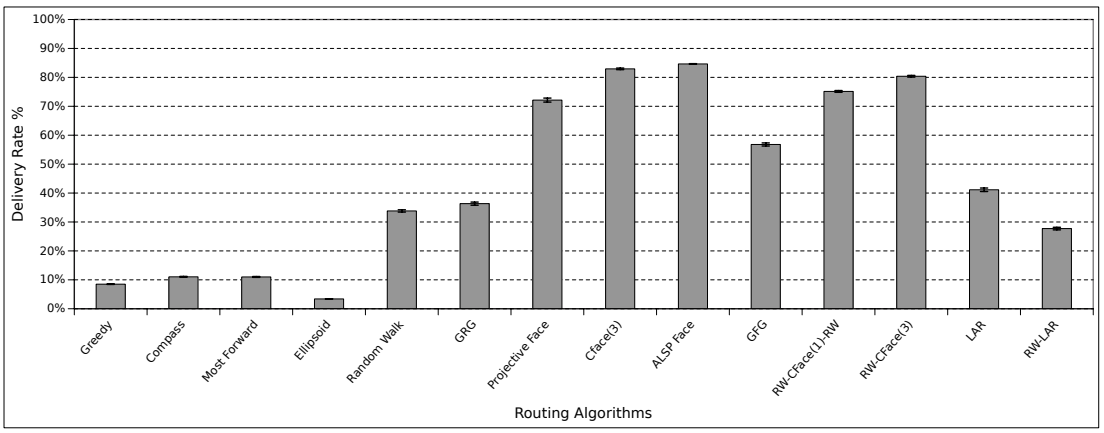
(a)



(b)

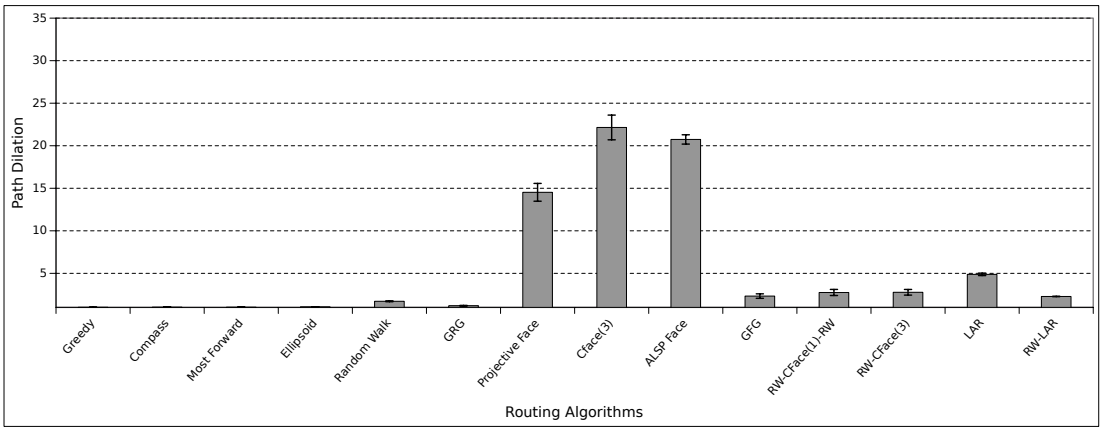


(c)

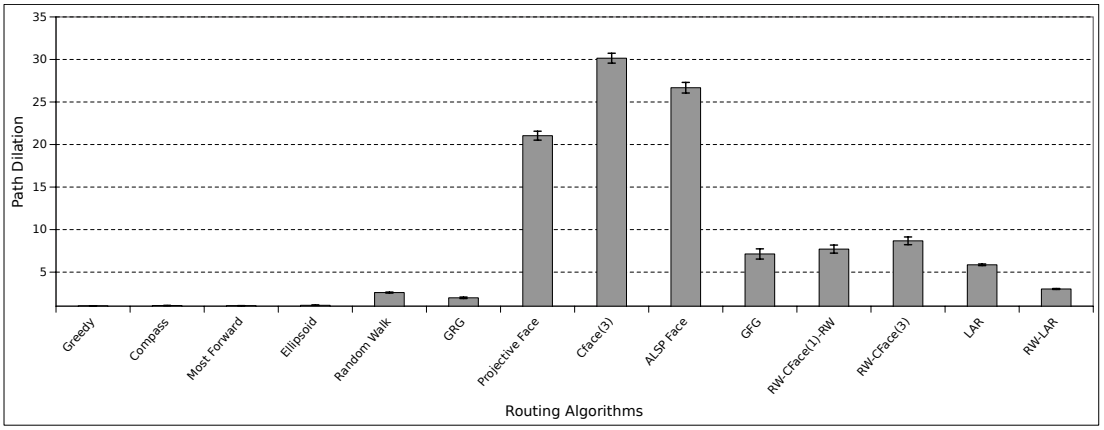


(d)

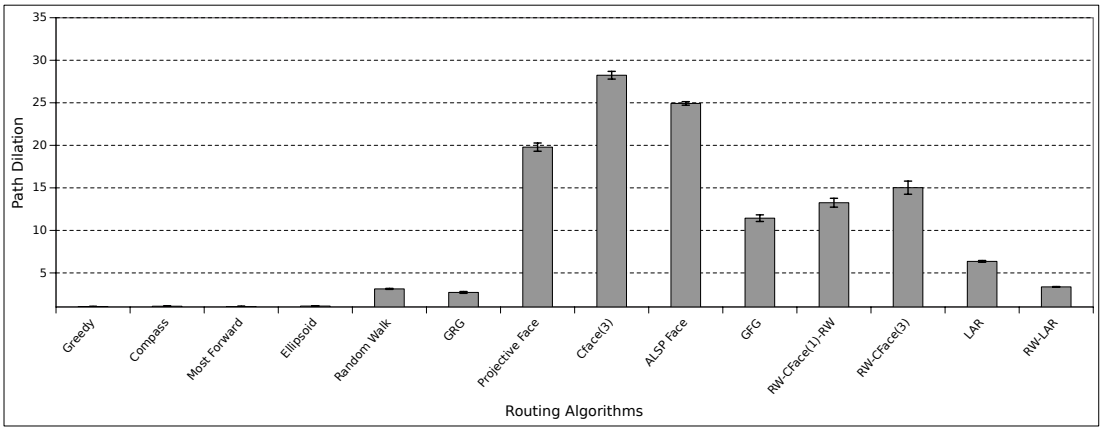
Fig. 9. Results of delivery rate



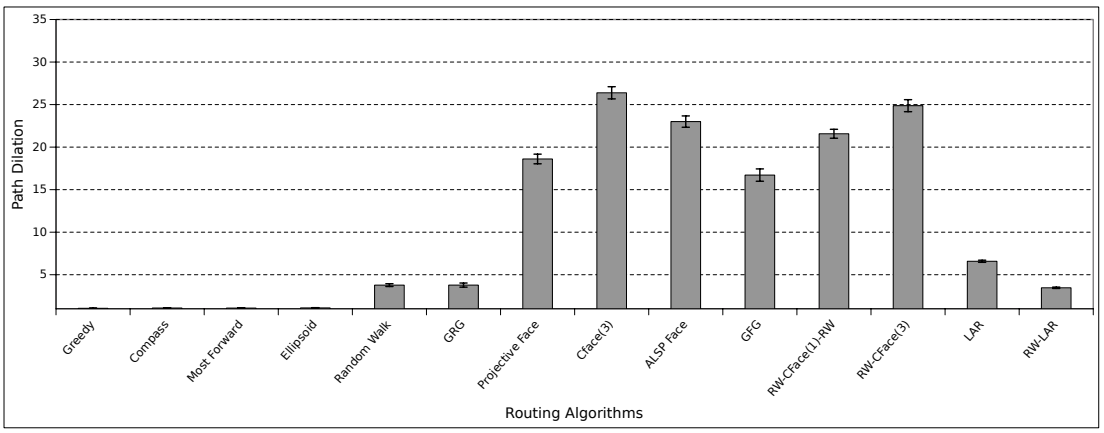
(a)



(b)



(c)



(d)

Fig. 10. Results of delivery rate