Restoration of lost connectivity of partitioned wireless sensor networks

Virender Ranga1*, Mayank Dave1 and Anil Kumar Verma2

Abstract: The lost connectivity due to failure of large-scale nodes plays major role to degrade the system performance by generating unnecessary overhead or sometimes totally collapse the active network. There are many issues and challenges to restore the lost connectivity in an unattended scenario, i.e. how many recovery nodes will be sufficient and on which locations these recovery nodes have to be placed. A very few centralized and distributed approaches have been proposed till now. The centralized approaches are good for a scenario where information about the disjoint network, i.e. number of disjoint segments and their locations are well known in advance. However, for a scenario where such information is unknown due to the unattended harsh environment, a distributed approach is a better solution to restore the partitioned network. In this paper, we have proposed and implemented a semi-distributed approach called Relay node Placement using Fermat Point. The proposed approach is capable of restoring lost connectivity with small number of recovery relay nodes and it works for any number of disjoint segments. The simulation experiment results show effectiveness of our approach as compared to existing benchmark approaches.

Subjects: Algorithms & Complexity; Computation; Computer Engineering

Keywords: Spider Web-1C; recovery relay node placements; connectivity restoration; large-scale node failure; convex hull; Fermat point

ABOUT THE AUTHOR
Virender Ranga received his PhD degree in Computer Engineering from National Institute of Technology, Kurukshetra in 2016 and MTech in Computer Science and Engineering from Guru Jambeshwar Govt. Technical University Hisar in 2004 respectively. He has done his BTech in Electronics and Communications from Kurukshetra University. Currently, he is an assistant professor in the Computer Engineering Department, NIT Kurukshetra Haryana, India with more than 15 years of experience in academic and administrative affairs in the institute. He has published more than 50 research papers in various International/ National referred journals and conferences like IEEE, Elsevier, Springer, T & F and ASP. He has guided more than 20 MTech dissertations and currently guiding one MTech dissertation inside as well as outside the institute. His areas of interest are pervasive computing, wireless sensor networks, wireless networks, routing algorithms, and mobile clouds. He has chaired various sessions in the International and National Conferences.

PUBLIC INTEREST STATEMENT
The lost connectivity due to failure of large-scale nodes plays major role to degrade the system performance by generating unnecessary overhead or sometimes totally collapse the active network. A very few centralized and distributed approaches have been proposed till now. The centralized approaches are good for a scenario where information about the disjoint network, i.e. number of disjoint segments and their locations are well known in advance. However, for a scenario where such information is unknown due to the unattended harsh environment, a distributed approach is a better solution to restore the partitioned network. In this paper, we have proposed and implemented a semi-distributed approach called Relay node Placement using Fermat Point. The proposed approach is capable of restoring lost connectivity with small number of recovery relay nodes and it works for any number of disjoint segments. The simulation experiment results show the effectiveness of our approach as compared to existing benchmark approaches.
1. Introduction

The use of wireless sensor networks (WSNs) has become very common in the real life in recent years due to tremendous advancement in its wireless and sensing technologies. One type of applications such as combat field reconnaissance, border protection, space exploration, etc. operates in the harshest environments, where sensor nodes reduce the risk of the human life (Ranga, Dave, & Verma, 2013; Stojmenovic, Simplot-Ryl, & Nayak, 2011; Younis, Sentruk, Akkaya, Lee, & Senel, 2013). Since, a sensor node is typically constrained in its on-board energy, computational power and communication resources, a large scale of sensor nodes is involved to ensure critical area coverage and increase the fidelity of the collected data (Younis et al., 2013). Due to small form factor and limited on-board energy supply, a sensor is very much susceptible to its failure. Due to hostile environments in which the network operates sometime result in large-scale damage of the sensor nodes that causes network partitioning in the network and converts into multiple disjoint segments. Figure 1 shows the scenario of partitioned network consisting of multiple disjoint segments. For example, some sensor nodes may be damaged under snow or sand after the storm or in the battlefield, a part of the deployment area may be attacked by the enemy explosives and, thus a set of sensor nodes in the vicinity would be destroyed (it is assumed that terrain is not destroyed). Therefore, rehabilitation of this large-scale partitioned network is the current research topic of WSN researchers in recent years. The deployment of recovery relay nodes (RRNs) in the disconnected network is the one way to connect the large-scale partitioned network. A RRN is more robust and capable node with significantly higher energy reserve and longer communication range \( R \) than sensor nodes. Although, RRNs can, in principle, be equipped with sensor circuitry, they mainly perform valuable data aggregation and data forwarding. Unlike sensor nodes, a RRN may be mobile and has some navigation capabilities. Thus, RRNs are favored in the partitioned recovery process, because these are easily accurately placed relative to the sensor nodes, and their radio range \( R \) is even larger, which facilitates and expedites the connectivity restoration among the disjoint segments effectively and efficiently. Intuitively, RRNs are more expensive and thus, a small number of RRNs should to be used for the recovery of the partitioned network. However, a small number of RRNs for network partitioning recovery can be found out using Steiner Minimum Tree (SMT) with Minimum number of Steiner Points and Bounded Edge Length (SMT-MSPBEL), but it is shown to be a NP-hard problem (Senel, Younis, & Akkaya, 2011) for the large-scale WSNs. Therefore, a polynomial time-distributed algorithm is required to deploy a small number of RRNs in the partitioned network for efficient recovery.

Recently, two major RRNs placement heuristics are proposed and published in this research area. In the first approach, the authors have proposed bio-inspired heuristic and used a spider web-like RRNs placement technique known as Spider Web-1C approach (Senel et al., 2011) with the disjoint

Figure 1. Articulation of segmented area due to large-scale failure of nodes (Ranga et al., 2015).
segments situated at the perimeter of the network. The proposed key idea is to form the stronger connectivity (i.e., 2-connectivity among the nodes), achieve better coverage and also enable balanced distribution of traffic load on the employed RRNs. One of the main advantages of the Spider Web-1C approach is that it connects the segmented network efficiently and effectively, but the major issue is the large number of deployed RRNs for the recovery of partitioned network.

The second approach is based on SMT called ORC-SMT (Lee & Younis, 2012). The key idea proposed by the authors is the use of the SMT by considering three outer disjoint segments that are formed in the grid structure after applying the convex hull algorithm recursively in the cyclic fashion. The points thus obtained are then applied recursively to find more Steiner Minimum Point (SMP) for the RRNs’ placement. The multiple points that come in the radio range then become a single point for RRN’s placement. Indeed, the procedure repeats itself till all the outer disjoint segments are not less than three segments for which the run is made. Then, RRNs are placed on these calculated points by applying Minimum Spanning Tree (MST) algorithm such as Kruskal or Prim algorithm. The main advantage of ORC-SMT that it connects various segments fastly and efficiently with small number of RRNs placements.

In this research paper, we consider the similar type of problem of large-scale nodes’ failure with large number of partitions occurs in the network. The distinction of our proposed work as compared with the published research works is that in our proposed approach, a large-scale node failure issue is addressed and solved using global zero gradient point inside the triangle, (the point at which distance from all the vertices of the triangle is minimum called Fermat Point (FP) in the best case and center of triangle in the worst case) which is not yet seen in other published works. However, in Ghosh, Roy, and Das (2009), the authors have considered a global minima based geo-cast routing technique for data propagation to reduce data transmission distances among the nodes to enhance network lifetime and further reduce end-to-end delay. The main contribution of our proposed solution is shown below:

1. Two naive approaches and two state-of-the-art heuristics are implemented along with our proposed approach Relay node Placement using Fermat Point (RPFP) to rehabilitate the lost connectivity of the partitioned network due to failure of large-scale nodes in WSNs.
2. Our proposed approach can work perfectly for any number of disjoint segments, even for disjoint segments that do not form equilateral triangles which is a constraint in SMT-based approaches (i.e. SMT-ORC approach (Lee & Younis, 2012)).
3. The proposed protocol RPFP is different from other published partitioning recovery approaches (i.e. ORC-SMT, Spider Web-1C) due to use of global zero gradient point inside the equilateral triangle in the best case or center of triangle in the worst case which is calculated using Equations 1–9 or 33–35, respectively.
4. Finally, our proposed approach shows excellent results in terms of number of placed RRNs, recovery time (RT) and average path length (APL) as compared to benchmark approaches when number of disjoint segments is larger (which is more realistic scenario in real application).

The remainder of the paper is organized as follows: In the next section, related work is described. Sections 3 gives the problem statement of our proposed approach. In Section 4, the detail about our proposed novel approach, i.e. RPFP is given. In Section 5, detailed pseudo-code is depicted and explained. Section 6 shows various published state-of-the-art approaches for comparison. In Section 7, the performance evaluation of our proposed approach through simulation experiments is given and compared with traditional approaches to prove its effectiveness. In the next Section 8, the article is concluded with future scope.
2. Related works

Many approaches have been published till last five years to tolerate single-node failure in wireless sensor and actor networks (WSANs)/WSNs. A WSAN is an extension of WSN with more robust actor nodes for data communications. A very few large-scale node failure recovery approaches in WSNs have been proposed till last two years. The authors of Ranga et al. (2013) and Younis et al. (2013) proposed the comprehensive study of the network partitioning recovery approaches in WSANs based on the different criterion. All approaches are classified into two broad categories (Ranga et al., 2013): (a) centralized approaches, and (b) distributed or semi-distributed approaches. The classification is further divided into three different categories, i.e. proactive, reactive and hybrid approaches. For proactive schemes, many approaches have been pursued to tolerate node failure in this area. The similar type of method is applied to reactive and hybrid approaches. In all proposed approaches, controlled node mobility has been used to rehabilitate the lost network connectivity. For instance, in Kansal, Somasundara, Srivastava, and Estrin (2004), a small robot called Packbot has been used to serve as a mobile RRN. The use of small robot enables the recovery of partitioned network or break links. An algorithm is used to determine the trajectory of moving robot in the network. A similar type of work is presented by Wang, Srinivasan, and Chu (2005). They have used mobile RRNs within 2-hop of the sink in the network to restore the lost connectivity of the partitioned network. Unlike (Kansal et al., 2004), the idea is that the RRNs do not need to run the long distance in the network to save their energy for different work. However, the use of Packbot is inefficient due to unexpected delays in data delivery, even when multiple devices are used in delay tolerant applications. Another reason is the slow motion of these devices to cover every individual best point in the small-scale network. Moreover, the cost of these devices is more, consequently, it increases the application cost.

(Wang, Cao, Porta, and Zhang (2005) exploited the node controlled mobility in order to cover the coverage holes which are not covered by sensor nodes during their initial deployment. The idea behind their work is to identify some spare nodes from different parts of the network that can be relocated to coverage-hole places in the network. Since moving a node for longer distance can drain significant node power, a cascaded movement can be pursued if the sufficient number of spare sensor nodes is available on the way. Therefore, finding spare nodes in the network is another critical issue in this proposed approach.

Recently, some centralized/semi-centralized approaches have been proposed to handle large-scale node's failure in WSN using cascaded control mobility of the nodes. In Alfadhly, Baroudi, and Younis (2010), the recovery problem is formulated as an Integer Linear Program (ILP). The objective of ILP-based optimization model is to form a connected topology while minimizing the individual traveling distance of the nodes.

The authors of Sir, Senturk, Sisikoglu, and Akkaya (2011) strive to restore lost connectivity by using a Multi-Integer Linear Programming (MILP)-based transportation network flow model. The proposed idea is to rehabilitate the lost connectivity with a minimum traveling distance of the nodes with the assumption that every node should be able to go all destinations i.e. reaches all other node positions when lost network connectivity restoration is to be required. Due to centralized nature, this approach does not scale well.

Another approach is proposed by Senturk, Akkaya, and Senel (2012) to improve the scalability by reducing the number of candidate locations. A RRN placement algorithm is used to find the set of locations which can guarantee the connectivity if RRNs are to be deployed to these locations.

Vemulapalli and Akkaya (2010) described another distributed approach based on nodes' knowledge of full paths to sink. The pre-failure route information is used to determine the location of the failed nodes. The location of nodes is obtained when break paths are established. Thus, upon partitioning, nodes can attempt to re-establish the path towards sink node by moving to next hop location. However, many nodes do the same in the partition, therefore, recovery cost can be high. To limit the recovery cost, the recovery process elects only one node as a leader node based on its distance.
from failed node or sink. When the leader node moves towards the sink, a cascaded movement of
nodes within that partition is also required in order to sustain intra-segment connectivity.

Another approach based on game theory is proposed by Senturk, Akkaya, and Yilmaz (2012) by
assuming the complete knowledge of location of partitions, number of partitions, and failed nodes.
Each partition is used as a player in the game. The payoff function is based on nodes’ degree and
elects a partition representative \( P_a \). The elected \( P_a \) opts to maximize the payoff of its partition which
motivates the partitions to move each other. Due to the centralized nature of this approach, each
representative node must know the payoff function of the other partitions and eventually network
reaches to Nash equilibrium when all partitions are moved to each other.

3. System model and problem statement

We assume a WSN in which a large number of sensor nodes are deployed throughout an area of
interest and a sink node is located in the middle of deployment. Without losing the generality, this
assumption ensures that there is a balanced traffic load in the network. Due to harsh environment
of the application like sensor nodes deployed in a battlefield for field monitoring, where nodes could
be destroyed by enemy explosives (assume that terrain is not destroyed), thus causing a large-scale
node failure which leads to multiple disjoint partitions in the network. For e.g. Figure 1 shows the
partitioned WSN with 8 disjoint segments having sink node is in the middle of the network. Thus,
more robust RRNs are required to connect this disjoint network.

Our problem can be defined as follows: “N sensor nodes know their positions using some localizati-
on algorithm are randomly deployed in an unattended area of interest (AOI). Let us assume that \( j \) dis-
connected sub-networks are formed as a result of failure of large-scale nodes in the network. Each
sub-network \( G_i \) has \( n_i \) sensor nodes, where \( 0 < n_i \ll N \). Our goal is to implement a polynomial time
algorithm that will rehabilitate the lost connectivity among the disconnected sub-networks \( G_i \) and
heal the partitioned network by deploying minimum number of RRN’s placement with their small
deployment time, and thus, create a new temporary connected network.”

4. Our proposed RPFP approach

Our proposed approach, unlike ORC-SMT, considers triangles of disjoint segments and finds a point
\( P \), where distance from each vertices of that \( \Delta \) is minimum i.e. gradient of all vertices become zero
inside \( \Delta \), instead of calculating individual SMT using three non-collinear segments (Lee & Younis,
2012). Our proposed approach exploits Fermat Point (FP) property of equilateral triangle to calculate
zero gradient point \( F_p \) inside \( \Delta \) for deploying RRNs towards the calculated point. If no equilateral
triangle exists in the network, then the center of the triangle is taken as FP. There are few questions
which need to be addressed to find the optimal solution for the recovery of lost connectivity due to
failure of large-scale nodes in WSNs:

(1) How disjoint segments after the failure of large-scale sensor nodes in the network can dis-
cover each other to connect again?

(2) How RRNs should be placed on their proper locations?

(3) How many RRNs is to be required to connect the partitioned network?

To address the first question in the challenged environment, we use a small size robot-like Packbot
initially to find the information about the disjoint segments and deliver this information to the sink
node for further recovery action. To further improve the recovery action in the network, RPFP uses
representative nodes (later explained) to deliver the information about its connected segment, in-
stead of taking information from all the sensor nodes of the disjoint segments.

To address the second question, we use Fermat Point (FP) based technique or center of triangle to
find the locations between the disjoint segments for placing RRNs.
The third question is addressed by placing RRNs by sink node without observing how many live sensor nodes are available in any disjoint segment. It is important to note here that the information of bigger segment is more crucial than smaller segment. Therefore, to give the priority to the bigger segments to connect first in the network, a convex hull algorithm is applied by each individual segment independently and calculates its representative nodes sits on the boundary. Further, each representative node calculates how many nodes \( N_i \) are attached in its segment by applying depth first search which is expressed as:

\[
N_i = \sum_{s_i} n_{s_i},\quad \text{where} \quad S_n = \{s_1, s_2, s_3, \ldots, s_m\} \quad \text{are the disjoint segments in the network}
\]

Based on \( N_i \), a node density function (NDF) for each segment is computed as follows:

\[
NDF = \frac{N}{\sum_m N_i} \quad \text{where} \quad m = \text{number of segments occurring in the network}
\]

After finding NDF, for each segment, these are sorted into ascending order. Now, zero gradient point i.e. \( FP \) is calculated by taking representative nodes \( (R_{ps}) \) of first three disjoint segments at a time. The next \( FP \) is calculated by taking next three \( R_{ps} \) of disjoint segments from the sorted list. In the first round, level first \( FP \) are computed using \( R_{ps} \). In the next round \( FP \) are calculated using previously calculated \( FP \) called level second Fermat Points and so on till network is connected. Figure 2 shows the actual methodology of calculation of \( FP \) in the network. It is assumed that the segments are sorted in ascending order i.e. \( s_1, s_2, \ldots, s_n \) etc.

**Definition 1**  
Let \( \Delta \) be a triangle with vertices \((x, y, z)\). The zero gradient point or Fermat Point (FP) is the point that minimizes the sum of Euclidean distance from the vertices of \( \Delta \) to \( FP \) i.e. minimize \( (d(x, FP) + d(y, FP) + d(z, FP)) \).

**Definition 2**  
Let \( \Delta \) be a triangle with vertices \((x, y, z)\) and an arbitrary point \( k \) lies in + - side or on any vertex position of \( \Delta \). The number of recovery RRNs requires to connect the vertices to point \( k \) is computed by:

\[
N_{RN}(\Delta, k) = (N_{RN}(x, w_1) + N_{RN}(y, w_1) + N_{RN}(z, w_1) + 1)
\]

\[
= \left( \frac{|d(x, w_1)|}{R} - 1 \right) + \left( \frac{|d(y, w_1)|}{R} - 1 \right) + \left( \frac{|d(z, w_1)|}{R} - 1 \right)
\]

\[
= \left( \frac{|d(x, w_1)|}{R} \right) - \left( \frac{|d(y, w_1)|}{R} \right) + \left( \frac{|d(z, w_1)|}{R} \right) - 2
\]

where \( w_1, w_2, w_3 \) are the weights along \( x, y, z \) vertices and \( R \) is the communication range of RRNs.

Figure 2. Example of calculation of chain of \( FP \).
In our system model, the following assumptions are taken in the simulation experiments without losing generality:

1. All RRNs are under the direct control of sink node and also assume that these are sufficient in number for network partitioning recovery.
2. The network is obstacle free for the movement of RRNs during the recovery process.
3. The intermittent failure of RRN does not occur during the recovery process.

Our proposed approach RPFP contains four main steps for the recovery of the partitioned WSN network:

1. Identification of failure of large-scale nodes in the partitioned network.
2. Calculate the representative node \( R_{P_s} \) of each disjoint segment in the construction of outer convex hull polygon.
3. Identify global zero gradient point inside the triangle of disjoint segments for RRNs’ placement.
4. Deployment of RRNs to rehabilitate lost connectivity of the partitioned network.

4.1. Large-scale node failure identification

In the first step, our proposed approach senses large-scale nodes failures in the network. The detection of large-scale node failure is illustrated in procedure Failure_detection() in the pseudocode, where each node sited on the boundary detects the failure of its neighbor nodes in the network. When the large-scale failure of nodes occurs in the network, the boundary nodes as calculated by the convex hull algorithm send a small Hello message to their 1-hop neighbor nodes and, then wait for reply till duration \( T_w \) before judging that their neighbor nodes failed or not. The \( T_w \) is calculated and equal to the sum of the time period of sending and receiving messages plus the node processing time \( (T_1 + T_r) + T_p \). In case the boundary node does not receive an answer in the stipulated period, then it considers the failure of its neighbor nodes and broadcasts Failure message for all the nodes connected to it. This process is repeated by all the different boundary nodes that are sited on the different disjoint segments as shown in Figure 3.

4.2. Calculation of \( R_{P_s} \)

Our second step is similar to that proposed in the approaches of (Ranga, Dave, & Verma, 2015) to find \( R_{P_s} \) of each disjoint segment. In this, our proposed approach finds out the point sets from each of the subnetworks by randomly selecting any point inside the disjoint segments and then determining its corresponding convex hull polygon. Indeed, the obtained convex hull polygons are smaller size polygons which consist of every point of its subnetworks. Hence, the nodes on the convex hull polygon are closer to other subnetworks than their corresponding convex hull polygon as illustrated in Figure 3. We considered these nodes as \( R_{P_s} \) in our proposed approach.
4.3. Calculation of FP inside the triangle (mathematical analysis)

To find the zero gradient point i.e. FP inside the triangle Δ, let us consider a point \( P \), which has coordinates \((x_i, y_i)\), where \( i = 1, 2, \ldots, 3 \) and another point \( P \) that has coordinates \((x, y)\) inside that triangle. Let \( d_i \) be the distance between \( P \) and \( P_i \), \( \forall i \in [1 \ldots 3] \). Then, our problem is to minimize the function as given below (Ranga et al., 2015; Shen & Tolosa, 2008):

\[
f(P) = \min(f(x, y)) = w_1d_1 + w_2d_2 + w_3d_3 = \min \sum_{i=1}^{3} w_id_i
\]  

(1)

It is a continuous function on 2-D plane \( \mathbb{R}^2 \) such that it must gain an absolute minimum value inside the closed \( \Delta \), i.e. \( P_1, P_2, \ldots, P_n \) where \( w_1, w_2, w_3 \) are the weights assigned to the edges of \( \Delta \), connected to point \( P \).

Let the distance between \( P(x, y) \) and \( \forall P_i \) is

\[
d_i^2 = (x - x_i^2) + (y - y_i^2), \quad \forall i \in [1 \ldots 3]
\]  

(2)

Then, gradient of \( f(P) \) is given by:

\[
g_d(P) = \frac{\partial}{\partial x}(d_i^2) = 2(x - x_i)
\]  

(3)

Therefore, if \( P \neq P_i \) then \( d_i \) is itself is differentiable i.e.

\[
2d_i \frac{\partial}{\partial x}d_i = 2(x - x_i), \quad \text{if } P \neq P_i
\]  

(4)

\[
\frac{\partial}{\partial x}d_i = \frac{x - x_i}{d_i}, \quad \text{if } P \neq P_i
\]  

(5)

Similarly, we can obtain for \( y_i \):

\[
\frac{\partial}{\partial x}d_i = \frac{y - y_i}{d_i}, \quad \text{if } P \neq P_i
\]  

(6)

Therefore, the gradient of \( f(d) \) is given by

\[
g_d(d_i) = \frac{1}{d_i}(x - x_i, y - y_i), \quad \text{if } P \neq P_i, \quad \forall i \in [1 \ldots 3]
\]  

(7)

Let \( u_i = \frac{1}{d_i}(P_i) \)

It is a unit vector defined for every \( P \neq P_i, \forall i \in [1 \ldots 3] \). Therefore, we can conclude that \( f(P) \) is differentiable on domain \( \mathbb{R}^2 \) i.e. \( \Omega = \mathbb{R}^2 \setminus \{P_1 \ldots P_3\} \) and its gradient is equal to:

\[
g_d(P) = w_1u_1 + w_2u_2 + w_3u_3 = \sum_{i=1}^{3} w_iu_i
\]  

(8)

Since the function \( f(x, y) \) has a minimum global value at this optimal point \( P \), therefore, all the trajectories of system will

\[
(x, y) = -g_d(x, y)
\]  

(9)

converge to the asymptotically stable equilibrium point \( P \). It means that \( G(x, y) = -\| g_d(x, y) \| ^2 = 0 \) is a global Lyapunov function for the system on \( \mathbb{R}^2 \) (proof is given in Appendix A). Moreover, the trajectories of Equation (9) are orthogonal and converged to a point called zero gradient point \( P \). Figure 4 shows the situation when \( \forall w_i \) converges to a point \( P \). Furthermore, closed lines are the level curve of \( f(x, y) \).
Lemma 1  A necessary and sufficient condition for $g_d(P)$ to be zero at some point i.e. $P$ on $\mathbb{R}^2$ is

\begin{align*}
  w_1 &< w_2 + w_3 \\
  w_2 &< w_1 + w_3 \\
  w_3 &< w_1 + w_2
\end{align*}

Proof  Geometrically, we can construct a non-degenerate triangle with its sides $w_1, w_2, w_3$. Indeed, $g_d(P) = 0$ which is equivalent to $w_1u_1 + w_2u_2 + w_3u_3 = 0$. Thus, it means that triangle curve with sides $w_u, w_v, w_w$ must be converged to point $P$ (as proved in Equations 1 to 9). Therefore, a necessary and sufficient condition for $g_d(P)$ is zero at this point $P$.

Lemma 2  If $g_d(P) = 0$ at some point $P \in \mathbb{R}^2$, then we have

\begin{align*}
  u_1, u_2 &= \left( \frac{w_3^2 - w_2^2 - w_1^2}{2w_1w_2} \right) \\
  u_2, u_3 &= \left( \frac{w_1^2 - w_2^2 - w_3^2}{2w_2w_3} \right) \\
  u_3, u_1 &= \left( \frac{w_2^2 - w_3^2 - w_1^2}{2w_3w_1} \right)
\end{align*}

Proof  Indeed, if $g_d(P) = 0$, then at $P$ we have $w_1u_1 + w_2u_2 + w_3u_3 = 0$. Let us take the dot multiplication of Equation $w_1u_1 + w_2u_2 + w_3u_3 = 0$ successively by $u_1, u_2$ and $u_3$ with the fact that $\| u_i \| = 1$. Hence, we can get

\begin{align*}
  w_1u_1u_1 + w_2u_1u_2 + w_3u_1u_3 &= 0 \\
  w_1u_1^2 + w_2u_1u_2 + w_3u_1u_3 &= 0 \\
  w_1 + w_2u_1u_3 + w_3u_1u_3 &= 0 \\
\end{align*}

Similarly, we can obtain the Equations for $u_2$ and $u_3$.
To simplify the Equations (18), (19), and (20), let us assume that:

\[ t_1 = u_2 u_3, \quad t_2 = u_3 u_1, \quad t_3 = u_1 u_2 \]  

(21)

Therefore, Equations (18), (19), and (20) look like:

\[ w_1 t_1 + w_2 t_2 + w_3 t_3 = 0 \]  

(22)

\[ w_1 t_3 + w_2 t_1 + w_3 = 0 \]  

(23)

\[ w_1 t_2 + w_2 t_1 + w_3 = 0 \]  

(24)

Let us multiply the Equation (22) by \( w_1 \), Equation (23) by \( w_2 \), and subtract Equation (23) from Equation (22). We get:

\[ w_1^2 - w_2^2 + w_1 w_3 t_2 - w_2 w_1 t_1 = 0 \]  

(25)

Take Equation (24) after multiply by \( w_3 \) and subtract it from Equation (25). Therefore, the resultant Equation looks like:

\[ u_2 u_3 = t_1 = \frac{w_1^2 - w_2^2 - w_3}{2 w_1 w_3} \]  

(26)

Similarly we can find the following resultant Equations:

\[ u_1 u_2 = t_3 = \frac{w_2^2 - w_1^2 - w_2}{2 w_1 w_2} \]  

(27)

\[ u_1 u_3 = t_2 = \frac{w_2^2 - w_1^2 - w_3}{2 w_1 w_3} \]  

(28)

**Lemma 3** If a point \( P \) lies on one of the sides of \( \Delta \), i.e. \( P_1, P_2, P_n \), as shown in Figure 5 and also it does not lie on the triangle vertices, then \( g_d(P) \neq 0 \).

**Proof** Let us assume that \( P \) lie on the side of \( P_1 P_2 \) as shown in Figure 5. Then unit vectors i.e. \( u_1 \) and \( u_2 \) are parallel. Therefore, \( w_1 u_1 + w_2 U_2 + w_3 U_3 \) are linearly independent and at least one is non-zero. Thus, \( g_d(P) = w_1 u_1 + w_2 u_2 + w_3 u_3 = \sum_i w_i u_i \neq 0 \).

**Lemma 4** If point \( P \) lies on the outside of the triangle, then \( g_d(P) \neq 0 \).

**Proof** Let us assume that the point \( P \) lies on the outside of \( \Delta \) as shown in Figure 6. Then, it must lie on one half plane of the \( \Delta \), whose boundary is the line joining between two vertices i.e. \( P_1 P_2 \) and it does not contain the remaining point i.e. \( P_3 \). Therefore, we draw the vectors \( w_1 U_1, w_2 U_2, w_3 U_3 \) terminate to common point \( P \) with the fact that all points lie on the same half plane with the boundary being parallel to \( P_1 P_2 \) through \( P \). Since all the vectors are non zero, therefore their sum is \( w_1 U_1 + w_2 U_2 + w_3 U_3 = g_d(P) \). Hence, it is contradicted with our assumption that \( P \) lies outside the triangle.

**Lemma 5** If a point \( P \) lies inside the triangle i.e. \( P_1, P_2, P_n \), then \( g_d(P) \) must gain its absolute minimum value in the interior of the triangle at exactly one point.
Proof: we have already proved in Lemma 1 that the gradient $g_d(P)$ of all the points vanishes at some point $P$. Let us consider a contradiction assumption that the minimum value is achieved at some other point $P'$ other than point $P$ inside the triangle. Then point $P$ lies on one of the triangles $\angle P, P_2, P_3, \angle P, P_1$ as shown in Figure 7. Since we have both $g_d(P) = 0$ and $g_d(P') = 0$, therefore we must have

$$u_1u_2 = \frac{w_1^2 - w_2^2 - w_3^2}{2w_1w_2}$$  \hspace{1cm} (29)$$

On the other hand, from Figure 7, we have

$$u_1u_2 = \cos \angle P_1PP_2$$  \hspace{1cm} (30)$$

$$u_1u_2 = \cos \angle P_1PP_2$$  \hspace{1cm} (31)$$

But, $\angle P_4PP_5$ is strictly bigger than $\angle P_1PP_2$, therefore, we must have $u_1u_2 < u_4u_5$, a contradiction assumption.

4.4. Placement of RRNs to rehabilitate the lost connectivity of partitioned network

After the identification of a point $P$ on which the gradient vanishes inside $\Delta$, our proposed approach RPFP will deploy RRNs towards that point to restore lost connectivity. However, the number of nodes to connect the segmented subnetworks depends upon the distance of any vertex to point $P$. Let $R$ be the range of a RRN node; therefore, our proposed approach deploys RRNs from each vertex towards $P$ at a distance $\left( \frac{d}{R} \right)$. Since, our approach is distributed in nature; therefore, we use greedy strategy to deploy RRNs towards $P$. To accomplish RRNs deployment task, the following substeps are given below:

1. The sink node executes our algorithm as given in Figure 8 on area of interest (AOI) to obtain a list of locations (i.e. $L$) towards $P$ on which the RRNs have to be placed. After that the sink node then broadcasts these calculated locations to all RRNs (it is assumed that all RRNs are in the vicinity of the sink node).

2. After receiving $L$, by all RRNs, they will keep it and mark each calculated location (i.e. $x, y$) as unoccupied, where $i = \{1, \ldots, m\}$. 

![Figure 5. Example of a point lies on the edge of $P_1P_2$](image1)

![Figure 6. Example of point lies outside the edge of $P_1P_2$](image2)
(3) The nearby RRN chooses an unoccupied location from \( L \) as its target position. The choice of the position is dependent on our objective function as given in Equation (1).

(4) The selected RRN then moves to the calculated location \((x_i, y_i)\) and marks this position as occupied location. After landing to the target position, RRN will periodically broadcast the status of \( L \) to sink for its connectivity. Moreover, while moving towards the calculated position, it is quite obvious that more than one RRN may move to same location, but these RRNs will compete with their cost (i.e. distance and their residual power). The cost function is calculated by each RRN:

\[
\text{cost} = \frac{\text{Remaining residual power}}{\text{Distance}}
\]  

(32)

Therefore, one with higher cost will win and keep moving towards the calculated position. Indeed, the nodes having lowest cost will reselect the new destination in the network after refresh their \( L_r \). In case of tie-break node ID is used for the same.

(5) Once a RRN is placed to its marked location it becomes the part of \( i \)th segment. Now, by taking each relocated RRN as reference node, the next node is placed until the distance between reference RRN and next node is less than \( R \). The same procedure is repeated for each disjoint segment according to the sorting order of node density function \( (NDF_i) \).

(6) Each RRN will repeat steps 2–5 until it reaches its calculated destination and marks its position as occupied.

In this way, the segmented subnetworks are contracted into a new connected network as shown in Figure 8. As aforementioned, our approach uses greedy strategy from all the disjoint segments to deploy RRNs, therefore, recovery time is lesser as compared with other state-of-the-art proposed approaches.

**Theorem 1**  The worst case time complexity of RPFP is \( O(m^3) \), where \( m \) is the number of disjoint segments that occurs after the large-scale node failure in the network and which is mapped on a given graph \( G = (V, E) \).

**Proof**  We use depth first search (DFS) to find the connected components in our proposed RPFP (Ranga, Dave, & Verma, 2014). Therefore, DFS takes \( O(m + e) \) time to find the connected components, where \( m \) is the number of segments in the network and \( e \) is the edges between the neighboring segments. The convex hull polygon can be found using convex hull algorithm (Graham, 1972) and it takes \( O(m \log m) \) time complexity. Since we apply convex hull algorithm on \( m \) disjoint segments as explained in the first step of our implementation, therefore, the total worst case time complexity is \( O(m^2 \log m) \). The minimum distance between two non-intersecting convex polygons can be found in \( O(m^3) \) time, because there are at most \( O\left(m^2\right) \) pairs of convex hull polygons in the configuration (Toussaint, 1984). The gradient point can be found in \( O(m) \) by applying Equations 4 to 12. Thus, the total worst case time complexity of our proposed RPFP is given by \( O(m + e) + O\left(m^2 \log m\right) + O\left(m^3\right) + O(m) \), which is equal to \( O\left(m^3\right) \).
Theorem 2  Our proposed approach RPFP connects the partitioned topology successfully and terminates guaranteed.

Proof  Since our proposed approach is distributed in nature; therefore, in step 4 of the RRNs’ placement guarantees that a RRN will eventually occupy its position \((x_i, y_i)\) if it wins the competition based on cost function. In case, RRN loses the competition, it will not move from its original position and reselect another position calculated by the sink node. Therefore, every time a selected RRN will move until it occupies its calculated location in the partitioned network. Thus, all calculated locations will eventually be occupied by RRNs, if sufficient RRNs are available with sink node. Therefore, our algorithm restores the partitioned topology successfully and terminates guaranteed.

5. Pseudo-code of our proposed RPFP
This section describes the pseudo-code of our proposed approach RPFP to rehabilitate the lost connectivity due to failure of large-scale nodes in the network.

The pseudo-code for RPFP() is shown in Figure 9. Before start of actual deployment of RRNs in the partitioned network, our proposed approach forms a convex hull polygon of the outer disjoint segments and computes \(R_p\) as shown in procedure Failure_detection(). Line 2 shows how to compute \(R_p\) sited on the boundary of the subnetworks. After detection of nodes’ failure, then, zero gradient point (P) i.e. FP inside the triangle \(\Delta\) is calculated based on these \(R_p\) and NDF in the procedure RPFP(). Line 3 to 13 in the same procedure is used to calculate P inside the triangle. Further, we draw the virtual lines (i.e. \(L\)) for the deployment of RRNs from each \(R_p\) to P. Line 19 is used in the procedure RPFP() to perform this operation. Line 10 to 13 is used to check that formed triangle is equilateral triangle or not. If it is the case, then FP is calculated otherwise center of \(\Delta\) is taken as FP for convergence. Further line 15 to 19 is used to check the status of the path i.e. path is connected or not and compute the location for RRNs’ placement. Our approach skips the virtual line if network is already connected by RRNs. Lines 1 to 3 is used to deploy the virtual lines from \(R_p\) to \(L\) as mentioned in the procedure deploy_virtualline(). Furthermore, the status of RRN’s deployment is checked and listed in the procedure deploy_RRNs () from lines 1 to 7. After that, we compute the location of RRN’s along the virtual line (\(L\)) from \(R_p\) to P and RRNs are added in the network based on their node radio range (i.e. R). Line 8 is used in the procedure RPFP() to perform this operation. We call our procedure RPFP() iteratively until all RRNs are not deployed up to P.

6. Various relay node placement state-of-the-art approaches
The following approaches are used to compare with our proposed novel approach RPFP:

1. Basic Deployment (BD): It is a very basic approach in which the isolated segments apply the Graham Scan i.e. convex hull algorithm in the partitioned network to form closed polygon. The outer segments then deploy the RRNs along the borders of the convex hull.
in the circular fashion. Similarly, the inner segments deploy the RRNs towards the nearest placed RRN as shown in Figure 10. In this approach, it is assumed that all nodes know the complete topology of the network before the partition occurs in the network. The main issue of this proposed approach is the number of required RRNs for healing the disjoint network.

2. Center Deployment with Convex Hull (CDCH): In this approach, first we calculate the out segment nodes like RPFP. The obtained outer segments then calculate representative nodes \( R_p \) and compute the center of deployment (CoD) for RRN’s deployment. Then, all \( R_p \) apply the convex hull algorithm to form closed polygon of the outer segments. Further, each representative node of the outer segment which is closer to CoD deploys the RRNs towards CoD. Moreover, the inner representative node deploys the RRNs towards the nearest deployed RRNs as shown in Figure 11. The main advantage of CDCH is that it requires small number of RRN placements as compared with the BD approach, although CDCH is also a centralized approach. The main problem in this approach is the scalability issue for large-scale network. The CoD can be computed as given below:

\[
\text{Area}(A) = \frac{1}{2} \sum_{i=0}^{n-1} (x_{i} y_{i+1} - x_{i+1} y_{i}) \tag{33}
\]

\[
X = \frac{1}{mA} \sum_{i=0}^{m} \{(x_{i} + x_{i+1})(x_{i} y_{i+1} - x_{i+1} y_{i})\} \tag{34}
\]

\[
Y = \frac{1}{mA} \sum_{i=0}^{m} \{(y_{i} + y_{i+1})(y_{i} x_{i+1} - y_{i+1} x_{i})\} \tag{35}
\]

where \( X \) and \( Y \) are the coordinates of the CoD and \( m \) is the number of disjoint segments occurs in the network.
3. **Spider Web-1C heuristic**: The key idea behind Spider Web-1C deployment strategy as explained by the authors (Senel et al., 2011) is to place the RRNs inward of the damaged area to yield better network connectivity and coverage. To balance the inter-segment path length in terms of the number of hops, RRNs are placed toward the estimated center of mass (CoM) of the segments. Basically, from each partition to CoM, Spider Web-1C gradually deploys nodes until all the partitions are connected effectively. In this way, Spider Web-1C not only increases the total coverage of the network, but also reduces the possible number of cut vertex nodes in the network as well. Before the placement of RRNs in the partitioned network, Spider Web-1C first needs to identify the outer segments in AOI. To do this, it randomly picks $R_p$s from each partition and runs a convex hull algorithm. The convex hull algorithm returns a subset of $R_p$s that are situated on the corner of a convex hull polygon. After identifying the convex hull polygon, it determines the CoM of the polygon as calculated in CDCH. Then, RRNs are deployed along the line between a segment and CoM. Obviously, the RRNs around the CoM will be in the communication range of each other, and all disjoint segments become connected. Figure 12 shows the pictorial representation of connected network with Spider Web-1C heuristic approach. The main advantage of this approach is that it requires small number of RRNs’ placement to rehabilitate lost connectivity in the partitioned WSN network as compared with BD and CDCH approaches.

4. **Optimal Relay Node Placement Algorithm using Steiner Minimum Tree (SMT) on Convex Hull (ORC-SMT)**: ORC-SMT (Lee & Younis, 2012) pursues greedy heuristic that has two main phases: (1) To identify the Steiner Points ($SP$) at which RRNs would be placed with the objective of minimum number of deployed RRNs to connect the segments, and (2) add additional RRNs in order to form a fully connected inner $SP$ topology while taking the communication range ($r$) of a RRN into consideration. The first phase contains further two main substeps that are applied repeatedly till the required $SP$ are not calculated. In the first
substep, ORC calculates the convex hull using convex hull algorithm to identify the boundary disjoint segments. Then, $SP_s$ that connect every three non-collinear neighboring boundary disjoint segments are identified. These $SP_s$ are defined as first-tier $SP_s$. Furthermore, for the unengaged segments, the convex hull is again computed to identify boundary terminals (i.e. disjoint segments or first tier $SP_s$) that are used in the second round and then second tier $SP_s$ are found. Indeed, third-tier $SP_s$ will be identified based on the second-tier $SP_s$ and so on. In other words, two substeps are applied recursively for $m$ rounds until the number of points considered for computing a convex hull is less than three or they form a complete graph in terms of $R$ of a RRN. ORC-SMT then moves to the second phase in which the identified $SP_s$ and disjoint segments are bound together. Basically, every segment $Seg_i$ identifies the closest $SP$ and then RRNs get placed on the line from $Seg_i$ to such $SP$. The same procedure applies for the first-tier $SP$ to connect them to the second tier and so on. As mentioned above, in the first phase, ORC operates in rounds. In the first round ($m=0$), ORC identifies a set of disjoint segments in the damaged area, which forms the smallest polygon that contains the other segments. Considering the segments as terminals, the convex hull of all segments is used to identify the boundary segments. The authors assume that there exist at least three non-collinear segments such that the convex hull $ch_0$ found in the first round forms a closed polygon. To find a convex hull, the authors use the Graham Scan algorithm. The main advantage of this approach is that it deploys small number of RRNs to connect the large damaged area. Figure 13 illustrates the connected topology of the partitioned network after the execution of ORC-SMT.

7. Simulation results and discussion

This section evaluates the performance of our proposed approach RPFP through simulation experiments. The purpose of simulation experiments acts as a proof of concept for the designed protocol. Using simulation experiments, it can be determined that whether the designed protocol adheres to the design requirements. The goal of simulation is also used to observe that the proposed approach RPFP outperforms other proposed approaches like ORC-SMT, CDCH, and Spider Web-1C. Our proposed approach is implemented and validated in C++ environment on (NS2.34, 2013). In the simulation, we also consider a linear energy model where energy consumption due to the movement of the RRNs is equal to 27.96 J/m (Ranga et al., 2015). Table 1 shows the simulation parameters used in the simulation.

The following parameters are considered in our experiments for simulation and comparison:

(1) Number of segments ($N_s$).
(2) Communication range ($R$) of an RRN.
(3) Number of placed RRNs.
(4) Recovery time ($T$)
(5) Average path length (APL)
In the simulation experiments, we have taken different topologies with the number of outer segments varies from 4 to 8 and inner segments varies from 2 to 5 i.e. total disjoint segments varies from 6 to 13 that are randomly located in an area of interest (i.e. 3000 m × 3000 m). While study the impact of R on the performance of the proposed approaches, it varies between 50 and 200 m in the simulation experiments. The results of the individual experiments are averaged over 40 trials of different topologies. We have also observed that with 95% confidence level, the simulation results stay within 6–10% of the sample mean. We also consider the number of placed RRNs, recovery time (sec), and APL for evaluating the performance of our proposed approach and compare with the existing approaches.

7.1. Number of placed RRNs
This metric reports the total number of required RRNs to rehabilitate the lost connectivity in the partitioned network. As aforementioned, RRNs are usually more expensive than sensor nodes. Thus, this metric reflects the total cost of repairing of partitioned network. Figure 14(a and b) shows the required number of RRNs placement while varying node radio range in the configuration. It is clear from the simulation graphs that the proposed ORC-SMT performs better than our proposed approach RPFP only when the number of partitions in the network is less than 5. However, our proposed approach performs well for any number of disjoint segments. Moreover, BD approach shows the large number of RRNs placement for the repairing of partitioned network due to deployment of nodes along the border of the convex hull in the circular fashion as explained earlier. Furthermore, Spider Web-1C shows similar result like our proposed approach RPFP as the node radio range increases due to observe the large size web-like structure made by Spider Web-1C approach for the placement of RRNs’ towards CoM.

In a nutshell, it is observed that ORC-SMT fails when the number of outer disjoint segments is more than 5 as shown in Figures 15 and 16. The reason is that in the randomly deployed topologies, when number of outer disjoint segments increases, more than one of the angles of the Steiner Triangle of SMT comes out to be greater than 120° (obviously some angles are less than equal to 120°), therefore, the calculated SP, comes out to be on the segment itself (on the segment whose edge angle is greater than 120°) as it is the property of the FP (i.e. SMT with three points behave like a Fermat Point). This raises the question of convergence ability of ORC-SMT approach towards the center for which the authors have claimed in their approach (Ranga et al., 2015). The situation becomes more intensive and critical when the number of outer segments grows critically which is a more common scenario in a harsh environment and thus, ORC-SMT fails in the simulations as observed in our experiments. In a nutshell, we can say that ORC-SMT behaves well when it serves with the small number of disjoint segments (i.e. less than 5), as we have verified in the simulation as shown in Figure 14(a and b). Furthermore, the Spider Web-1C heuristic runs almost parallel to the CDCH when the node radio range is larger. This is because the web formed by Spider Web-1C would be much earlier from the CoM, therefore, the small number of nodes is required for the repairing of the lost connectivity. The RPFP shows excellent results as compared with ORC-SMT, Spider Web-1C, CDCH, and BD as the number of outer disjoint segments increases or as a node radio range is larger. The reason is the
deployment of a small number of RRNs from all the segments towards zero gradient point $FP$ as explained earlier. Moreover, Figures 15 and 16 confirm the effectiveness of our proposed approach.

7.2. Recovery time of the network (RT)

It defines the total time taken by any approach to rehabilitate lost connectivity due to failure of large-scale nodes in the network. It is always expressed in CPU time (i.e. in sec). Figure 17(a) shows the effectiveness of our proposed RPFP over compared approaches. The recovery time of BD and CDCH is more due to their centralized nature. Although, CDCH performs better than BD due to node deployment with this approach towards CoD rather than on the edges of convex hull polygon. It means CDCH takes less time to rehabilitate the network partitioning as we observed in Figure 16(a).

The ORC-SMT behaves well because of its distributed nature when number of segments are up to 5. Moreover, our proposed approach RPFP also shows excellent results as the node radio range is larger in the network. However, Spider Web-1C shows good results as the node radio range is larger, because the network connects quickly due to large size of web made by the proposed solution before reaching to CoD. Figure 17(b) shows the recovery time varying with the number of outer disjoint segments. Proposed solution shows excellent results due to deployment of RRNs towards zero gradient point inside the triangle and can work for any number of disjoint segments. Moreover, RPFP shows small variation in its plot due to the least change in zero gradient point position in the triangle for a large number of segments.

| Table 1. Simulation parameters (Ranga et al., 2015) |
| Parameter | Value |
|-----------|-------|
| Simulation area | 2,000 m x 2,000 m |
| Sensor nodes (n) | 250 |
| Radio model | Path loss model |
| MAC layer | IEEE 802.15.4 |
| Communication range (R) | 50-200 m |
| Node initial energy ($E_i$) (AA battery) | 16,200 J |
| Total number of partitions observed | 6-13 |
| Channel frequency | 2.4 GHz |
| Packet size | 1 byte |
| Antenna model | Omni-directional |
| Failure model | Random |
| Data transmission rate | 1 packet/s |
| Simulation time | 100 s |
7.3. APL
This metric depicts the expected path length between the partitions after the recovery has been done. A small APL is desirable as explained in various state-of-the-art approaches, because it will reduce the data latency between the various partitions when network partitioning is rehabilitated between various disjoint segments. Figure 18(a), (b) show APL as the function of number of outer segments and node radio range, respectively. It has been clearly seen that ORC-SMT shows excellent results only when the number of disjoint segments is smaller, but our proposed approach RPFP shows excellent results for any number of disjoint segments. Moreover, the accuracy of the calculated zero gradient point and the diameter of the convex hull polygon also affects the APL. Thus, Spider Web-1C shows excellent results compared with BD and CDCH approaches because of
big size web-like structure maintained by this approach as node radio range increases. Furthermore, ORC-SMT and our proposed approaches show similar results as the node radio range is higher and when number of outer disjoint segments are 5, because of accuracy of zero gradient point (i.e. like a center point of a circle) inside the convex hull polygon as observed in Figure 18(b).

8. Conclusion and future scope
The performance of unattended WSNs depends upon the connectivity among the backbone connected dominating set (CDS) of sensor nodes. There are many reasons which can disrupt network connectivity, permanently such as low battery level of CDS nodes, physical damage of large-scale sensor nodes due to hostile natural incidents like natural disasters such as a sandstorm, earthquake, fire storm, explosive attack of an enemy, etc. In this paper, we have proposed a new approach based on the Fermat Point called RPF for RRNs’ placement in the large-scale partitioned WSN. The main strength of our proposed approach is that it requires a small number of RRN placements and also works for any number of disjoint segments as compared with existing benchmark approaches. The simulation results also confirm the goodness of our proposed approach. In the future, our study can focus on testing of our proposed approach with routing protocol in the presence of obstacles to evaluate the actual network performance parameters like throughput, end-to-end delay, packet loss, delivery ratio, etc.
Appendix A

(Proof of $\parallel g_d(x, y) \parallel^2 = 0$):

Proof We know that $w_1 u_1 + w_2 u_2 + w_3 u_3 = \sum_{i=1}^{3} w_i u_i = 0$ such that $g_d(P) = 0$. Therefore, we have

$$g_d(P) = w_1 u_1 + w_2 u_2 + w_3 u_3$$

(36)

Take square of Equation (36) i.e.

$$\parallel g_d(P) \parallel^2 = (w_1 u_1 + w_2 u_2 + w_3 u_3)(w_1 u_1 + w_2 u_2 + w_3 u_3)$$

(37)

$$= \left(w_1^2 + w_2^2 + w_3^2 + 2w_1w_2u_1u_2 + 2w_1w_3u_1u_3 + 2w_2w_3u_2u_3\right)$$

We know that according to cosine law

$$w_1 w_2 = \cos(\Pi - \Theta_1), \quad w_2 w_3 = \cos(\Pi - \Theta_2), \quad w_3 w_1 = \cos(\Pi - \Theta_3)$$

Replace these values in Equation (38), and then we get following Equations

$$\parallel g_d(P) \parallel^2 = \left(w_1^2 + w_2^2 + w_3^2 + 2w_1w_2[\cos(\Pi - \Theta_1)] + 2w_1w_3[\cos(\Pi - \Theta_2)] + 2w_2w_3[\cos(\Pi - \Theta_3)]\right)$$

(38)

$$\parallel g_d(P) \parallel^2 = \left(w_1^2 + w_2^2 + w_3^2 - 2w_1w_2\cos\Theta_3 - 2w_1w_3\cos\Theta_2 - 2w_2w_3\cos\Theta_1\right)$$

(39)

Again, according to cosine law

$$w_1^2 + w_2^2 - 2w_1w_2\cos\Theta_3 = w_3^2$$
Putting this value to Equation (39) for further simplification

\[
\| g_d(P) \|= (w_2^2 + w_3^2 - 2w_1w_3 \cos \Theta_2 - 2w_1w_3 \cos \Theta_1)
\]  

(40)

Add and subtract \( w_1^2 \) to Equation (40) for further simplification

\[
\| g_d(P) \|= (w_2^2 + w_3^2 - 2w_1w_3 \cos \Theta_2 + w_2^2 - 2w_1w_3 \cos \Theta_1 - w_1^2)
\]  

(41)

\[
\| g_d(P) \|= (w_2^2 + w_3^2 - 2w_2w_3 \cos \Theta_1 - w_1^2)
\]  

(42)

\[
\| g_d(P) \|= (w_2^2 - w_1^2) = 0
\]  

(43)

Hence proved.