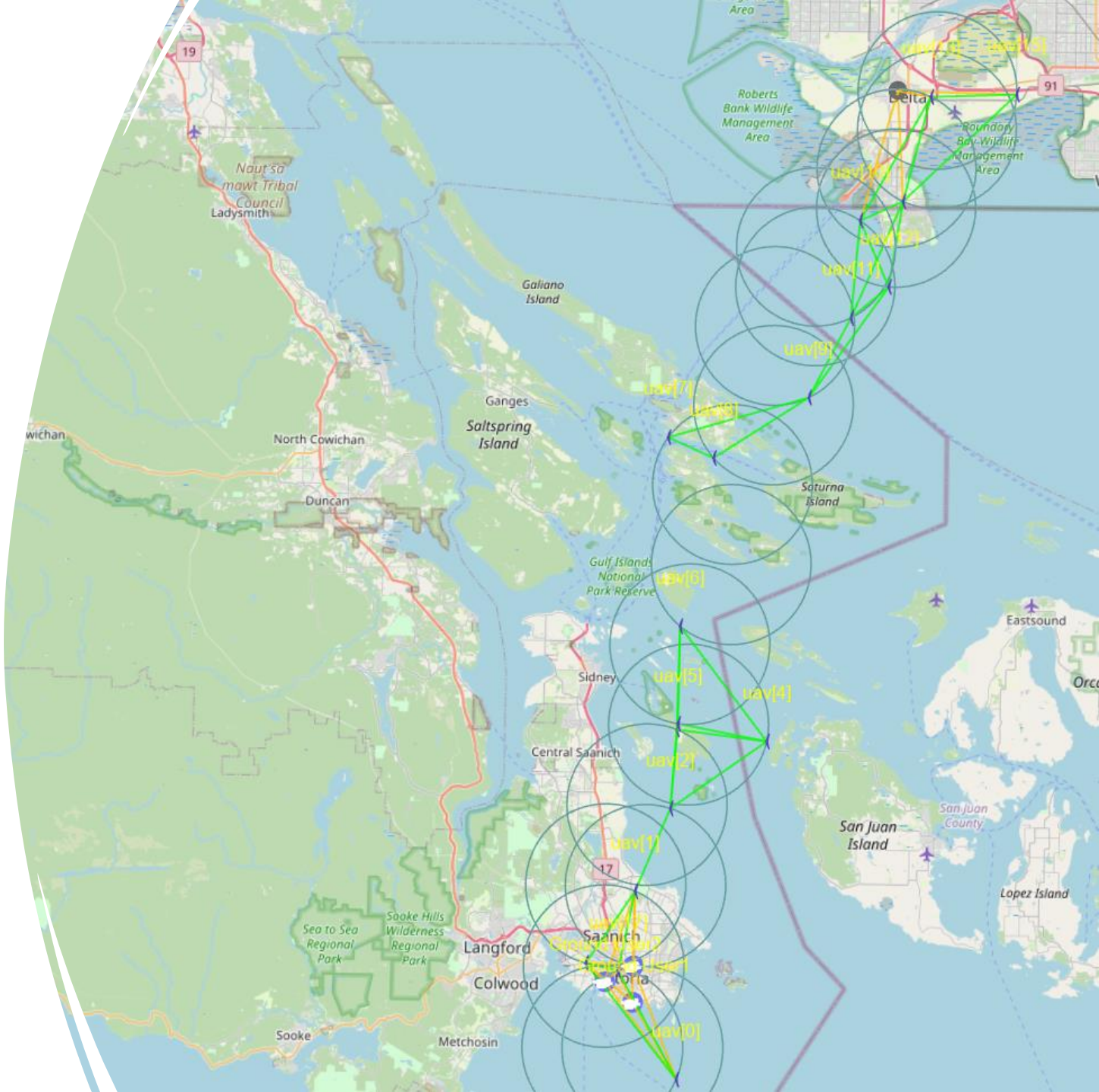
Optimal Routing Protocol and Deployment Strategy for Multi-UAV Relay Networks

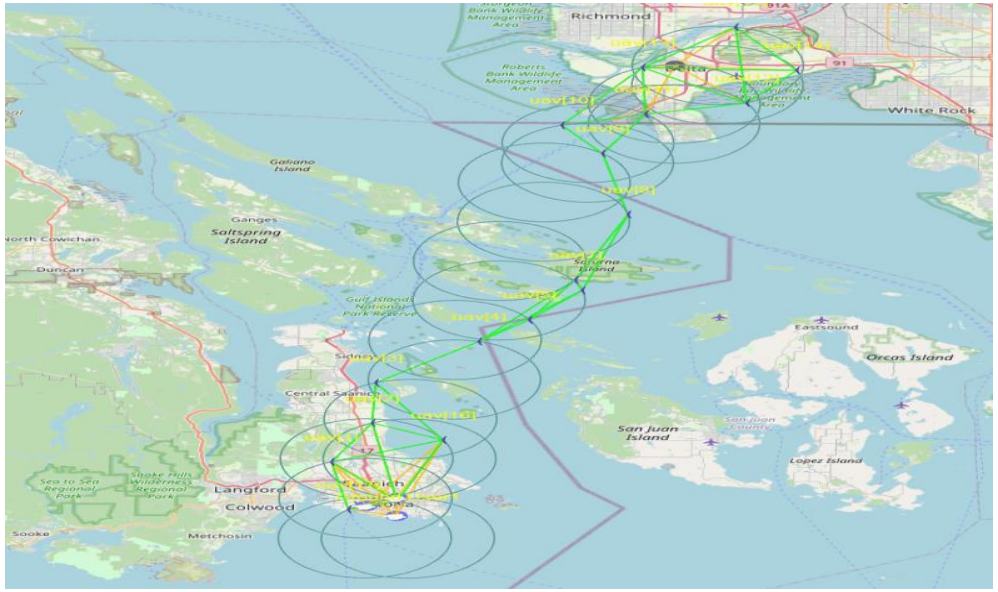
YoungHyo Kim, Jon Edwards

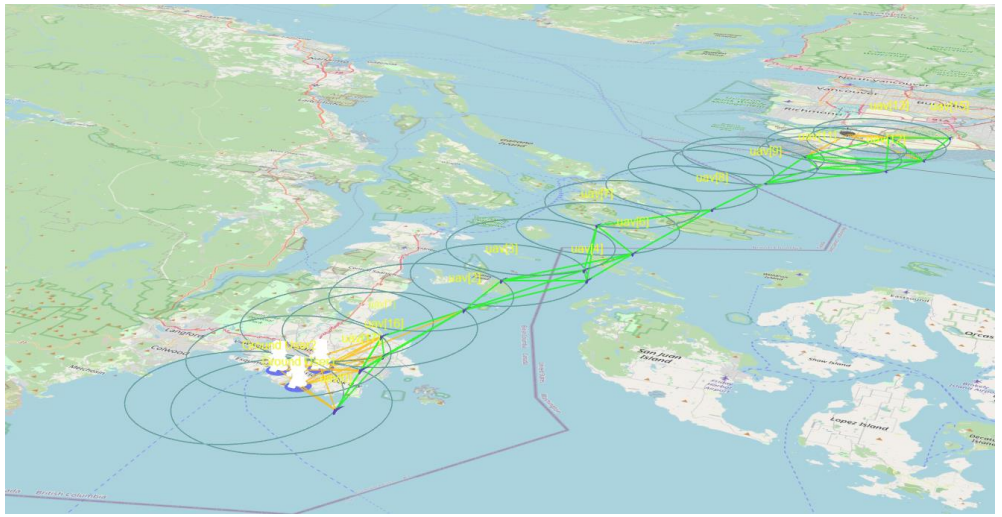
Overview of UAV Relay Network

- This project focuses on establishing an emergency communication network using a multi-UAV network in disaster and military scenarios where conventional infrastructure has been destroyed.
- When local ground base stations (GBS) are unavailable or inaccessible, a UAV-based Flying Ad Hoc Network (FANET) is deployed to maintain communication using a hybrid network structure.
- The key goal is to connect UAVs to an operational GBS and relay LTE/5G services to ground users via an airborne base station UAV.



Show cases


Richmond
White Rock
Galiano Island
Ganges
Saltspring Island
Duncan
Eastsound
Orcas Island
San Juan Island
Lopez Island
Langford
Colwood
Metchosin
Sooke



Project Goal

Optimizing UAV Deployment Strategy

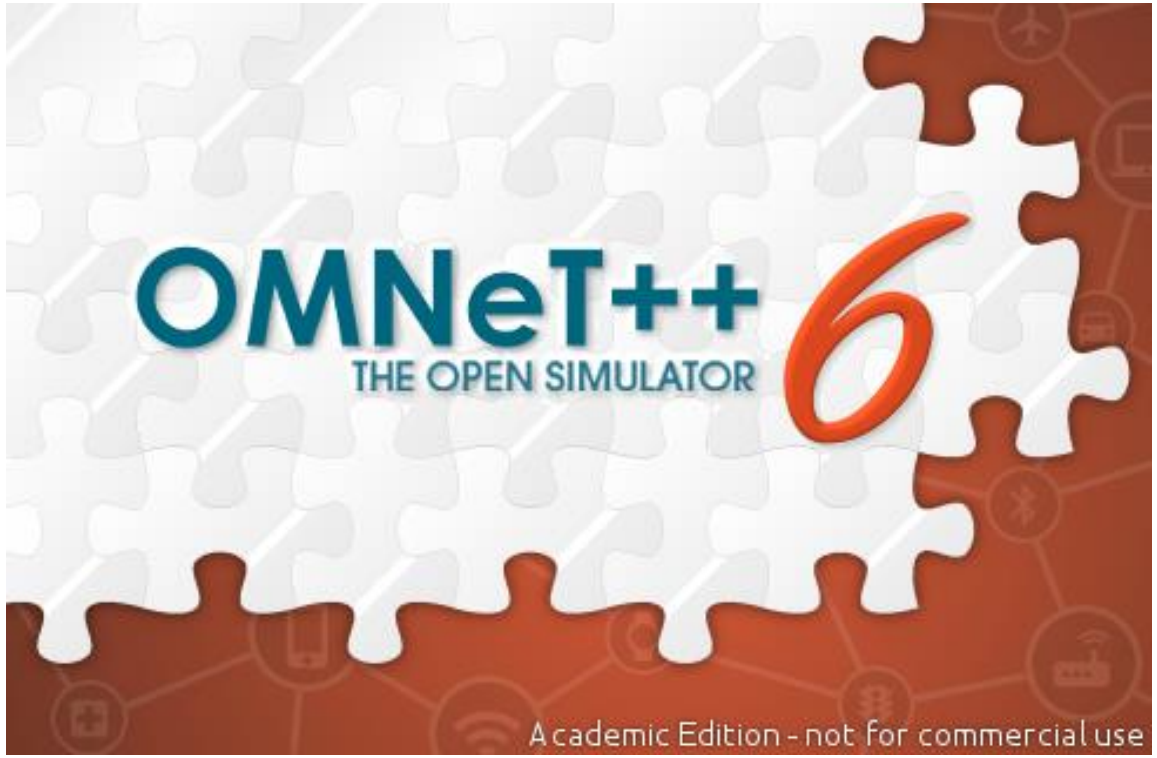
Finding an Optimal Routing Protocol

OMNet ++

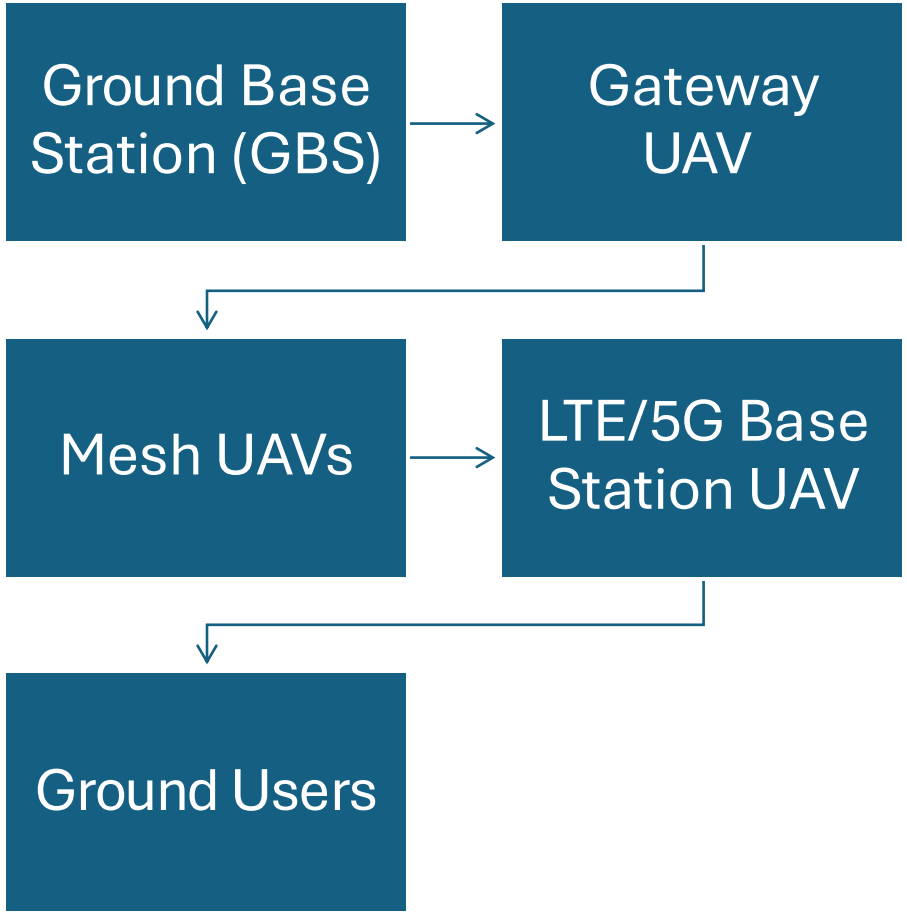
OMNeT++ is a modular C++-based open source simulation framework.

Commonly used for network simulations: wireless, IoT, UAV (FANET), and more.

Provides visual simulation tools with 2D and 3D support.



Key Components of our simulation



1) Local Ground Base Station (GBS)

- Function: Acts as the primary backhaul for the UAV network.
- Connection: Directly communicates with a gateway UAV.
- Purpose: Provides access to the internet and external communication networks.

2) Gateway UAV

- Function: Serves as the primary link between the GBS and the mesh UAV group.
- Operation: The UAV closest to the GBS dynamically assumes the gateway role.
- Purpose: Relays network traffic between GBS and other UAVs in the FANET.

3) Mesh UAV

- Function: Maintains network connectivity between UAVs and enables multi-hop relaying.
- Operation: Uses mesh networking to find the optimal path for packet transmission.
- Purpose: Expands network coverage and prevents single points of failure.

Simulation setup

Sender: user[0]

Receiver: gs[0]

UAV:

- Num: 4
- Altitude: 500m
- Radius: 500m
- StartingPhase: uniform(0deg, 360deg)
- RotationPeriod: 60s

PingApp

- **Interval:** Every 3 seconds
- Start Time: 20s
- Stop Time: 3600s

Protocols Tested:

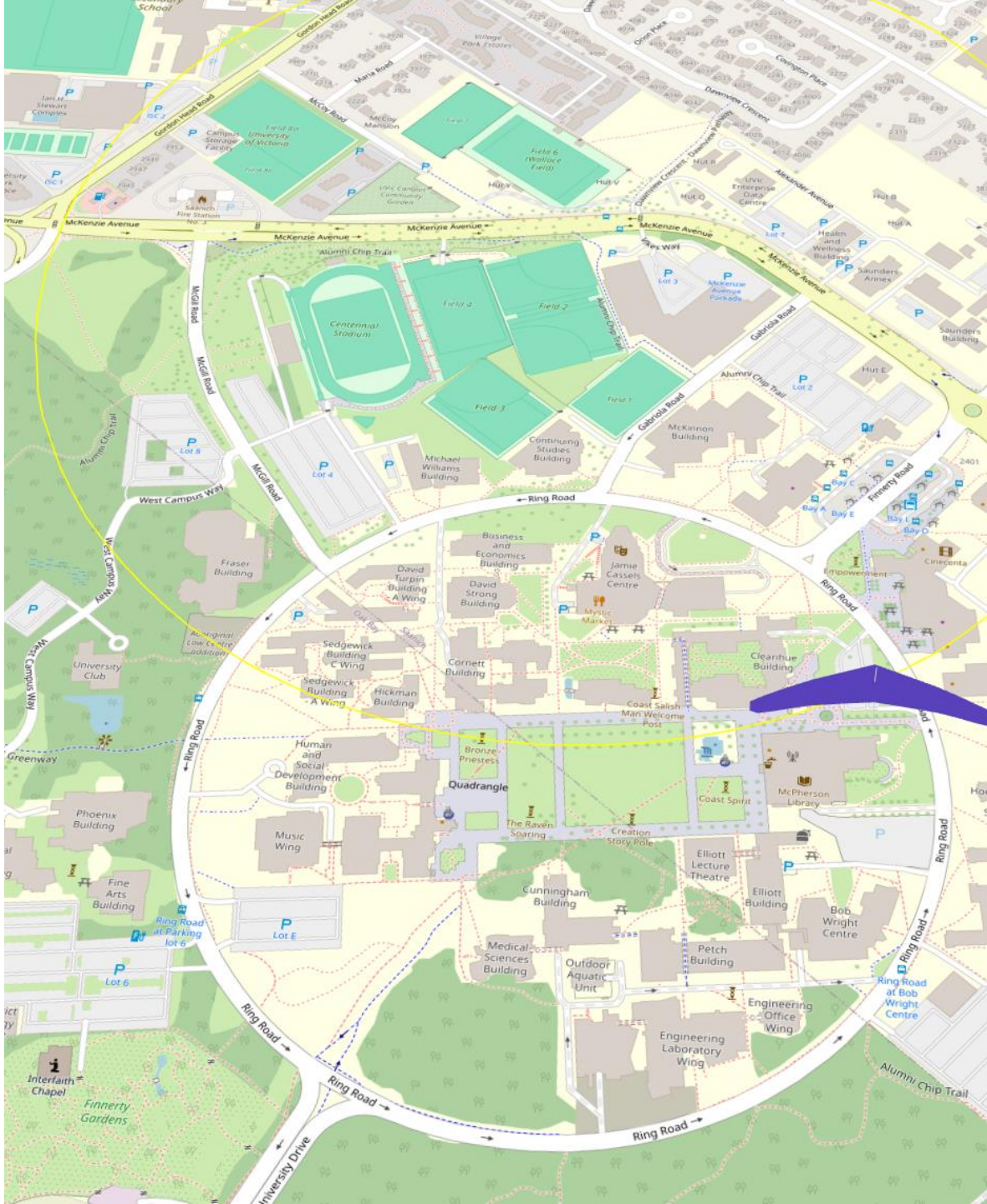
- GPSR
- DYMO
- AODV
- RGR
- DSDV

Communication Ranges:

- User = 3000m
- GS = 50000m
- UAV= 30000m

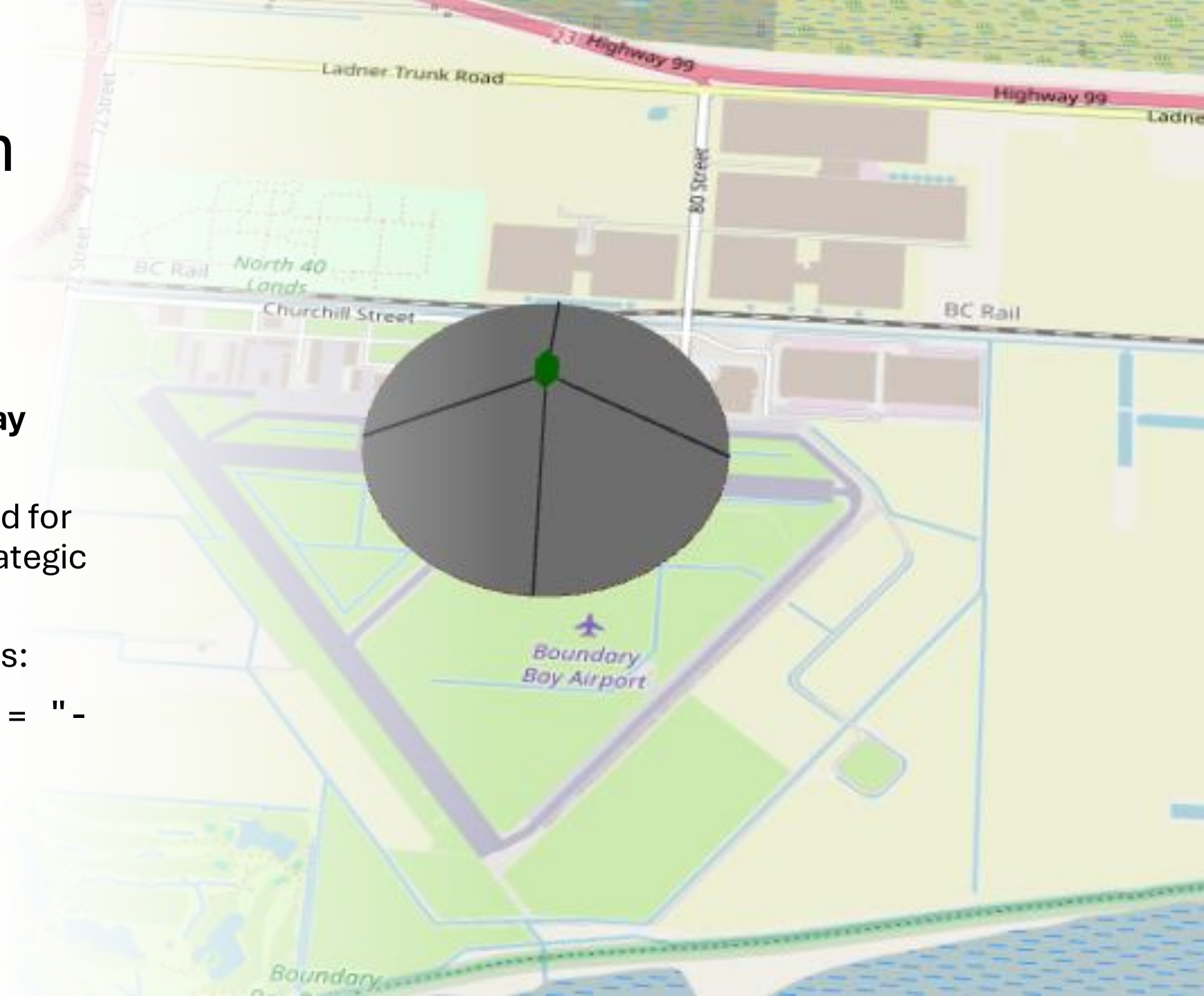
Ground User Deployment

- We deployed **two users** on the **University of Victoria** campus.
- **User[0]** is positioned **in front of the MacLaurin Building**.
- **User[1]** is positioned **in front of the Library**.
- The coordinates are set as:
- `*.user[0].orbitCenter = "-13727180 6184200"`
- `*.user[1].orbitCenter = "-13726850 6184300"`



Ground Station Deployment

- The **Ground Station** is deployed at **Boundary Bay Airport** in **Vancouver**.
- This location was selected for its open airspace and strategic coverage.
- The coordinates are set as:
- `*.gs[0].orbitCenter = "-13693000 6288000"`



UAV Displacement Strategy

- **Positions:**
  - BS at $B = (x_B, y_B)$
  - User at $U = (x_U, y_U)$
  - UAV at (x, y) with fixed altitude h
- **Free-Space Path Loss:**

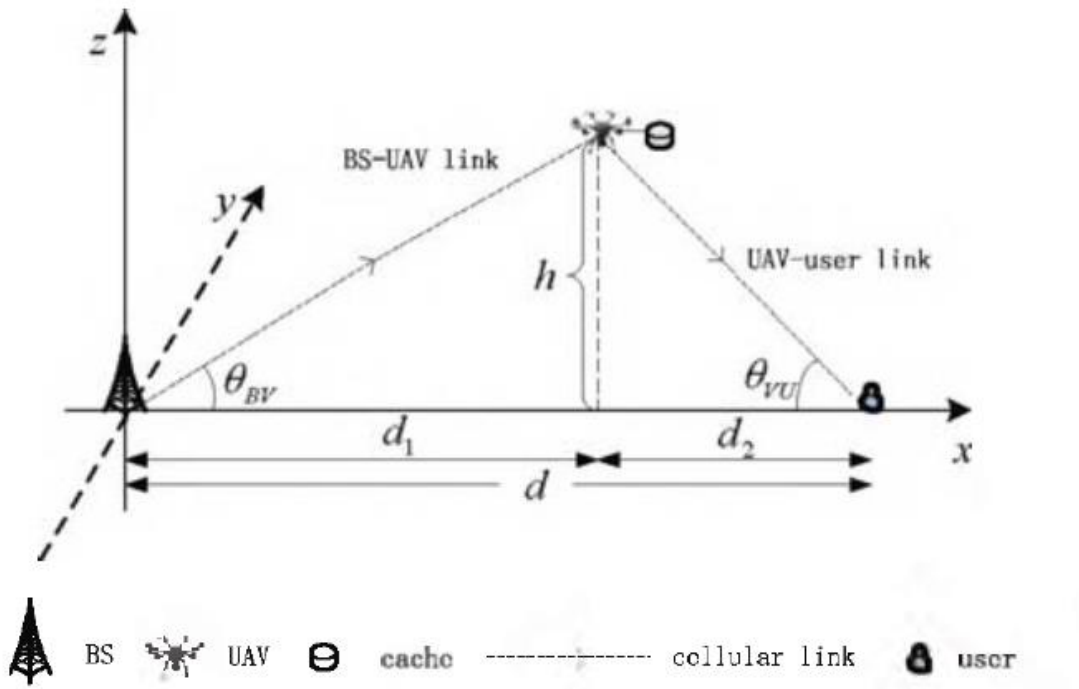
$$g(d) = \frac{\beta_0}{d^2}$$

- **SNR Expression:**

$$\text{SNR} = \frac{P\beta_0}{\sigma^2 d^2}$$

- **Throughput per Link (half-duplex):**

$$R = \frac{1}{2}\log_2(1 + \text{SNR})$$



UAV Displacement Strategy (Single Relay)

- Define distances:

$$d_1 = \sqrt{(x - x_B)^2 + (y - y_B)^2 + h^2}, \quad d_2 = \sqrt{(x - x_U)^2 + (y - y_U)^2 + h^2}.$$

- **Optimal Condition:**
  - To balance the SNRs, set $d_1 = d_2$.
- With fixed altitude h, equate:

$$(x - x_B)^2 + (y - y_B)^2 = (x - x_U)^2 + (y - y_U)^2.$$

- This condition places the UAV on the perpendicular bisector of B and U (ideally at the midpoint).

UAV Displacement Strategy (Multi Hop)

- **Given Coordinates:**
  - BS: $B = (-13\,693\,000,\ 6\,288\,000)$
  - User: $U = (-13\,727\,000,\ 6\,184\,200)$
- **Difference Vector:**

$$\Delta = U - B = (-34\,000,\ -103\,800)$$

- **Total Distance:** ≈ 109,300 m
- **For 4 hops (3 relays):**

$$\Delta_{\text{seg}} = \frac{\Delta}{4} = (-8\,500,\ -25\,950)$$

- **Calculated Positions:**
  - UAV1: $B + \Delta_{\text{seg}} = (-13\,701\,500,\ 6\,262\,050)$
  - UAV2: $B + 2\Delta_{\text{seg}} = (-13\,710\,000,\ 6\,236\,100)$
  - UAV3: $B + 3\Delta_{\text{seg}} = (-13\,718\,500,\ 6\,210\,150)$
  - User remains at $U = (-13\,727\,000,\ 6\,184\,200)$
- Each segment ≈ 27.3 km, which is within the 30 km maximum range.

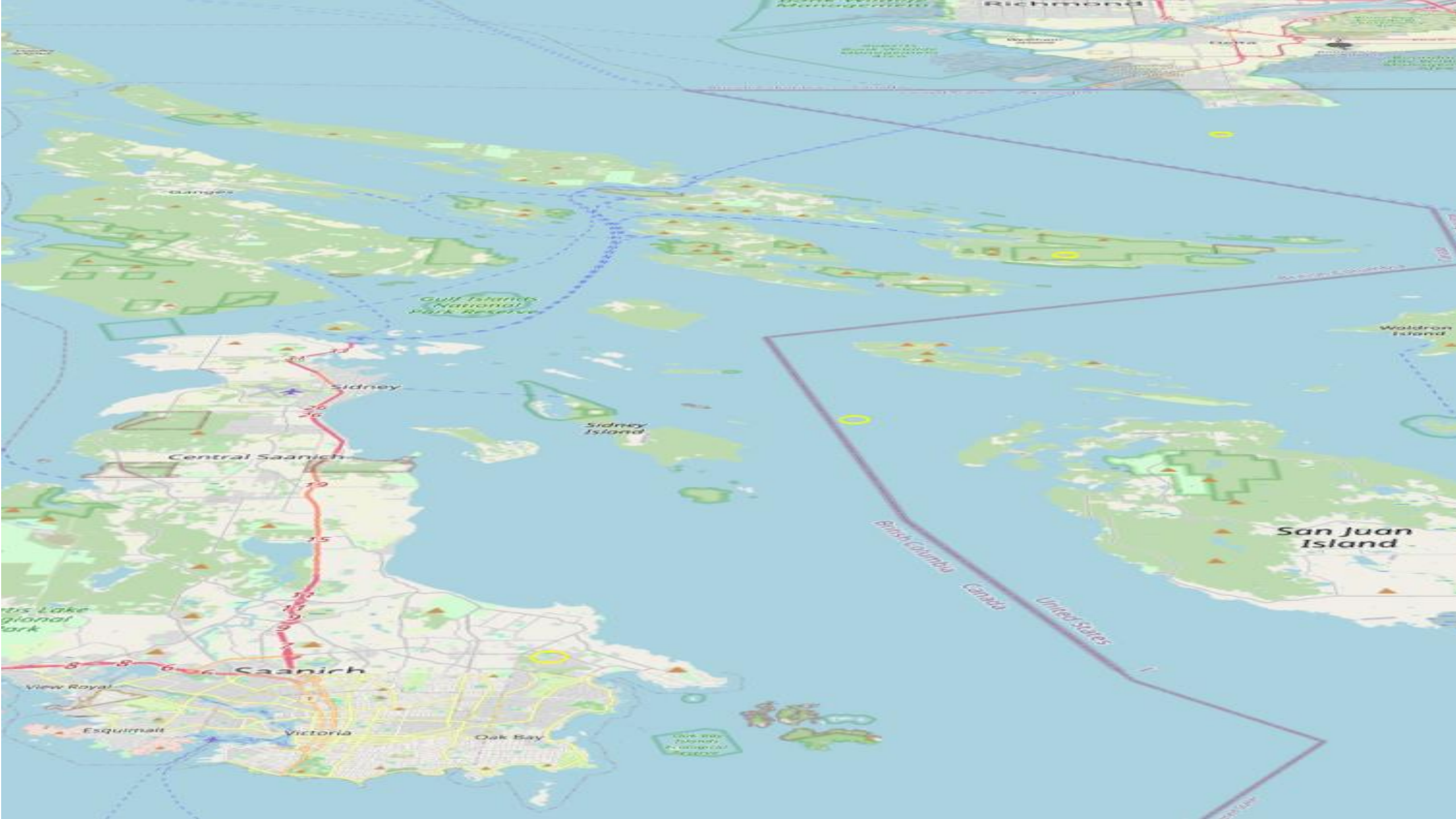
- When D (the ground distance between BS and User) is too large, use multiple UAV relays.
- Divide D into N equal segments:

$$d = \frac{D}{N}.$$

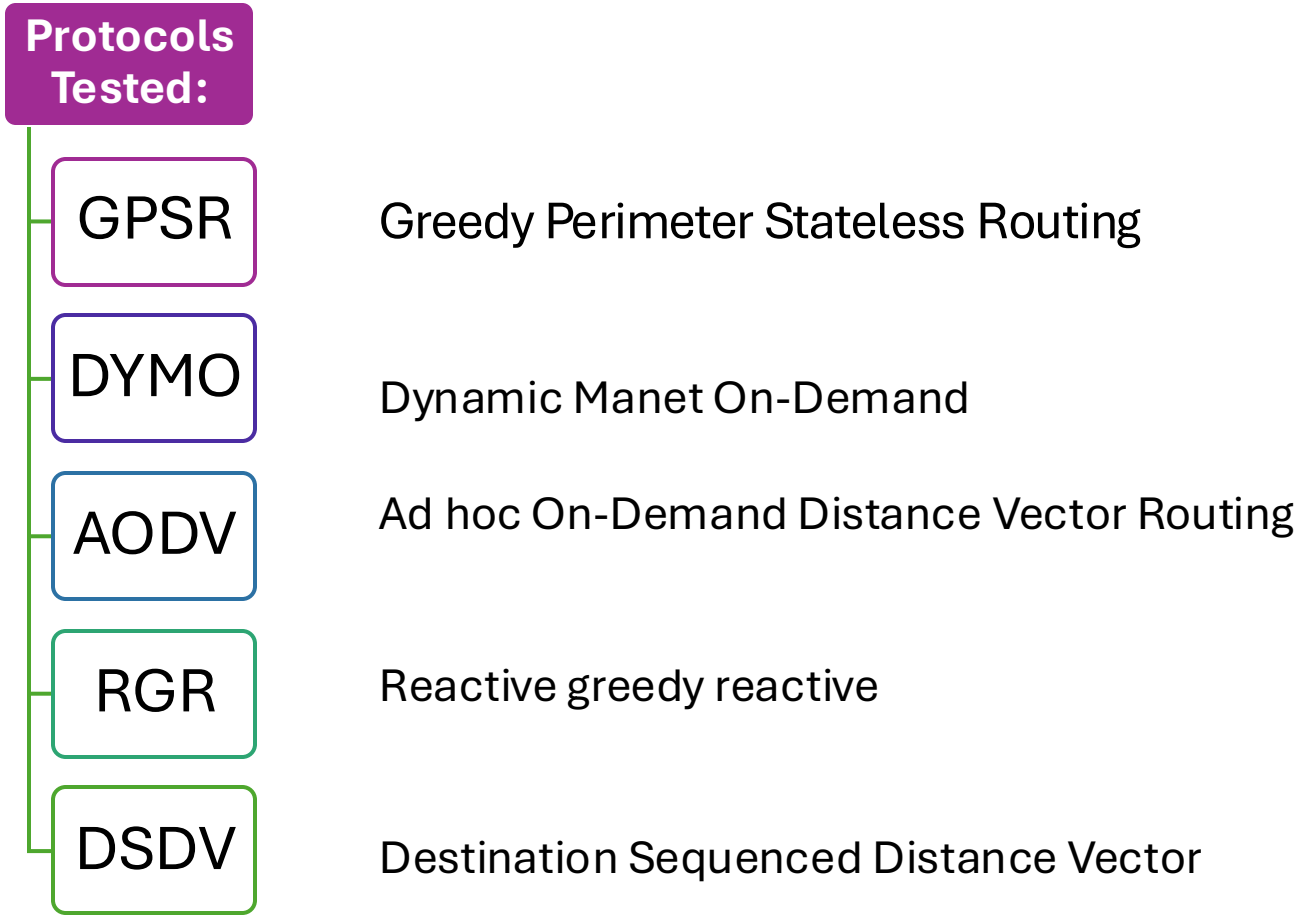
- Place each relay at positions corresponding to fractions of the displacement vector from B to U:

$$\text{Position of relay } i: \quad B + \frac{i}{N}(U - B) \quad (i = 1, 2, \ldots, N-1).$$

- This ensures each hop is within the communication range and the SNRs are balanced.


Richmond
Ganges
Gulf Islands National Park Reserve
Sidney
Sidney Island
Central Saanich
Saanich
View Royal
Esquimalt
Victoria
Oak Bay
Waldron Island
San Juan Island
British Columbia
Canada
United States

The Protocols We Tested



GPSR: Greedy Perimeter Stateless Routing

What is GPSR?

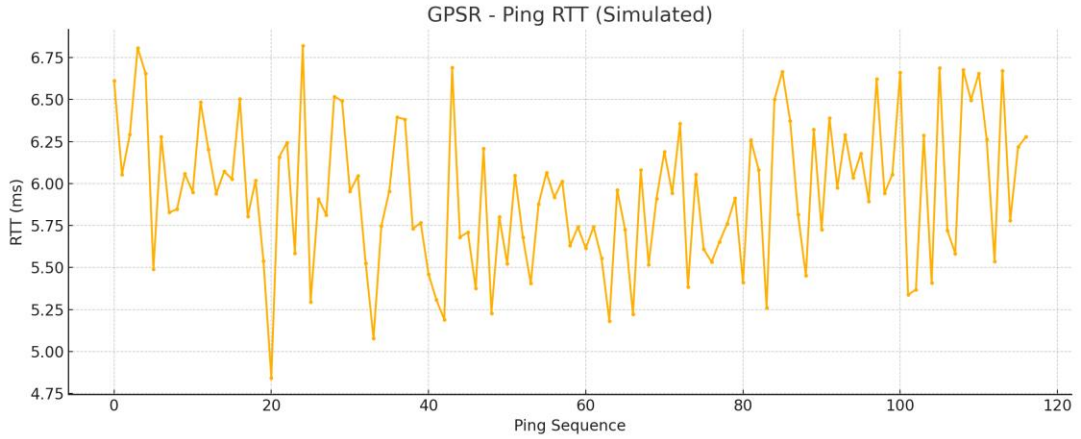
- **Position-based routing protocol** for wireless ad hoc networks.
- Uses **node location (GPS coordinates)** instead of traditional routing tables.

How It Works

- **Greedy Forwarding**
  - Forward packet to the **closest neighbor** to the destination.
  - Fast and efficient in dense networks.
- **Perimeter Forwarding (Recovery Mode)**
  - Used when greedy forwarding fails (e.g., no closer neighbor).
  - Follows the **right-hand rule** along the network graph until a better path is found.

Result

- ----------
- OsgIndoorNet.user[0].app[0]
- ----------
- sent: 114 received: 114 loss rate (%): 0
- round-trip min/avg/max (ms): 5.81661/5.88933/8.56215
- stddev (ms): 0.411984 variance:1.69731e-07
- ----------



DSDV: Destination Sequenced Distance Vector

What is DSDV?

- A **proactive (table-driven)** routing protocol for mobile ad hoc networks.
- Based on the classic **Bellman-Ford algorithm** with added sequence numbers.

How It Works

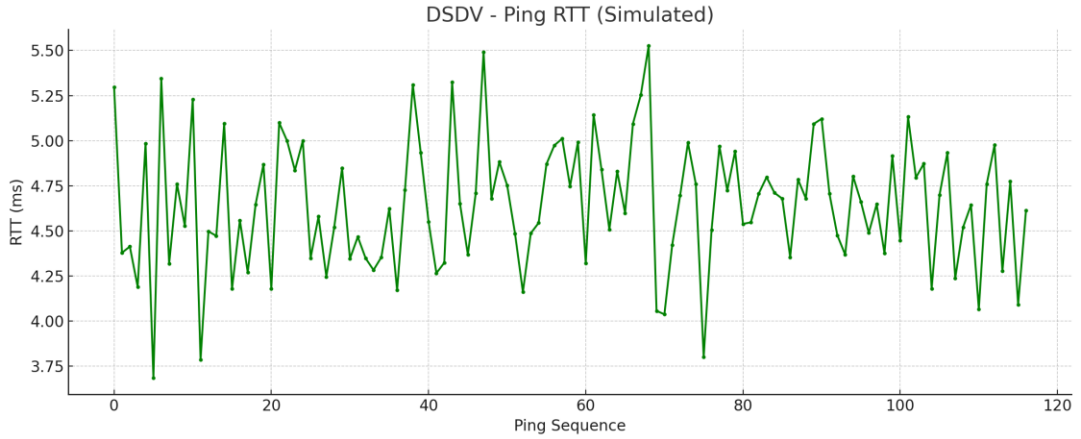
- Each node maintains a **routing** table with:
  - Next hop
  - Distance (hop count)
  - **Destination sequence number** (to avoid loops)
- Routing tables are **periodically broadcast** to neighbors.
- Sequence numbers help ensure **loop-free** and **up-to-date** routes.

Result

- ----------

- OsgIndoorNet.user[0].app[0]
- ----------

- sent: 114 received: 101 loss rate (%): 11.4035
- round-trip min/avg/max (ms): 4.53671/4.63518/7.28218
- stddev (ms): 0.410985 variance:1.68909e-07
- ----------



DYMO: Dynamic MANET On-demand Routing

What is DYMO?

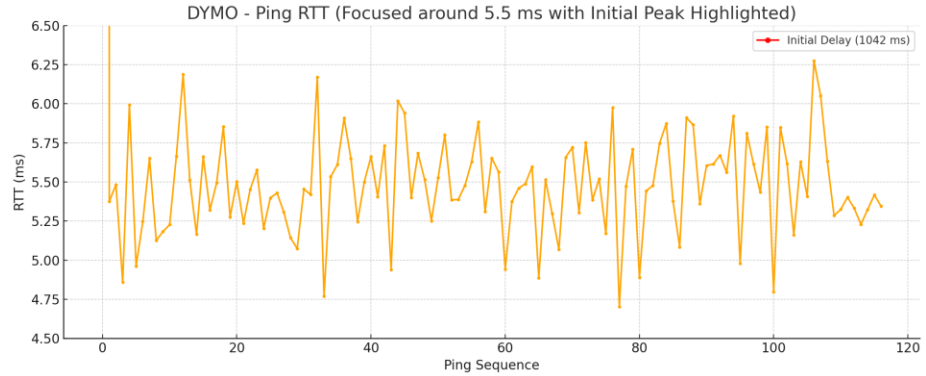
- A **reactive (on-demand)** routing protocol for mobile ad hoc networks (MANETs).
- Successor to **AODV**, with a simpler and more efficient design.

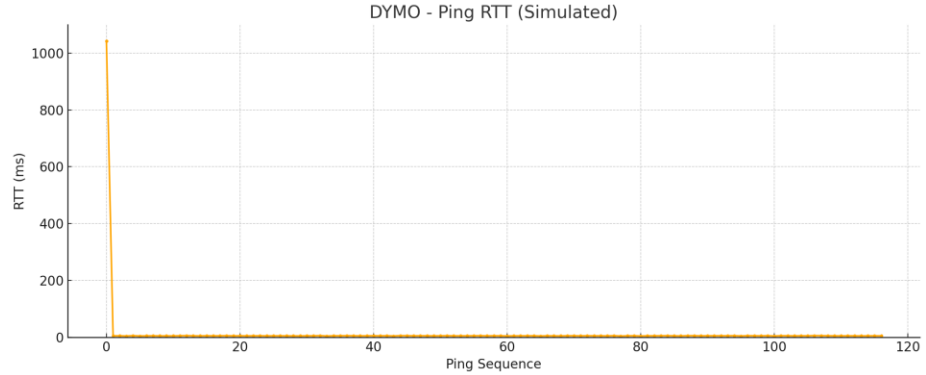
How It Works

- **Route Discovery**
  - When needed, the source node **broadcasts a Route Request (RREQ)**.
  - The destination replies with a **Route Reply (RREP)**.
- **Route Maintenance**
  - If a link breaks, a **Route Error (RERR)** message is sent to affected nodes.
- **Routing Table Updates**
  - Nodes **learn routes** as they forward messages, reducing overhead.

Result

- ----------
- OsgIndoorNet.user[0].app[0]
- ----------
- sent: 117 received: 117 loss rate (%): 0
- round-trip min/avg/max (ms): 5.48913/14.4093/1042.3
- stddev (ms): 95.8482 variance:0.00918688
- ----------





AODV: Ad hoc On-Demand Distance Vector Routing

What is AODV?

- A **reactive** routing protocol for mobile ad hoc networks (MANETs/FANETs).
- **Routes are created only when needed** (on-demand).
- Uses **sequence numbers** to ensure freshness and avoid loops.

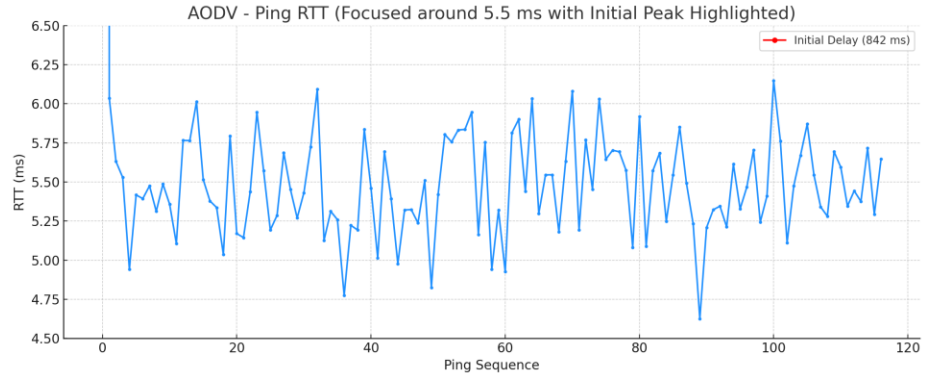
How It Works

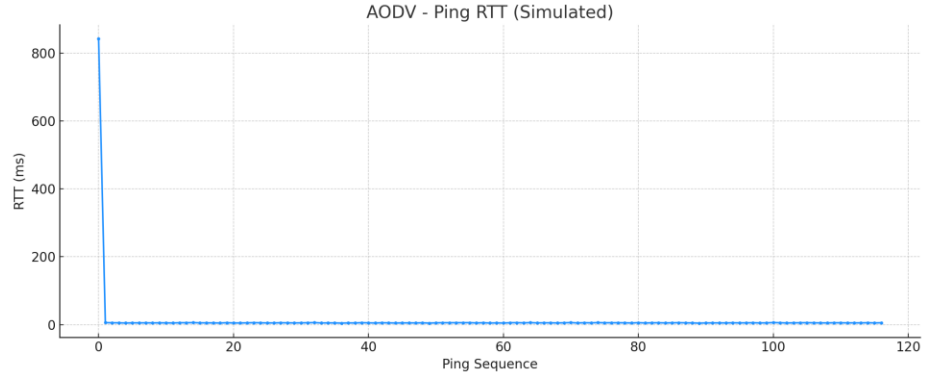
- **Route Discovery**
  - Source node broadcasts a **Route Request (RREQ)**.
  - Destination or intermediate node replies with a **Route Reply (RREP)**.
- **Route Maintenance**
  - If a link breaks, a **Route Error (RERR)** is sent to notify affected nodes.
- **Routing Table**
  - Each node maintains a **routing table** with active routes.

Result

- ----------
- OsgIndoorNet.user[0].app[0]
- ----------

- sent: 117 received: 117 loss rate (%): 0
- round-trip min/avg/max (ms): 5.48931/12.6974/842.001
- stddev (ms): 77.331 variance:0.00598008
- ----------





RGR: Reactive-Greedy-Reactive Routing Protocol

What is RGR?

- A **hybrid routing protocol** designed for **highly dynamic networks** like FANETs.
- Combines the strengths of **reactive** and **greedy geographic routing**.
- Designed to handle **frequent topology changes** in UAV-based networks.

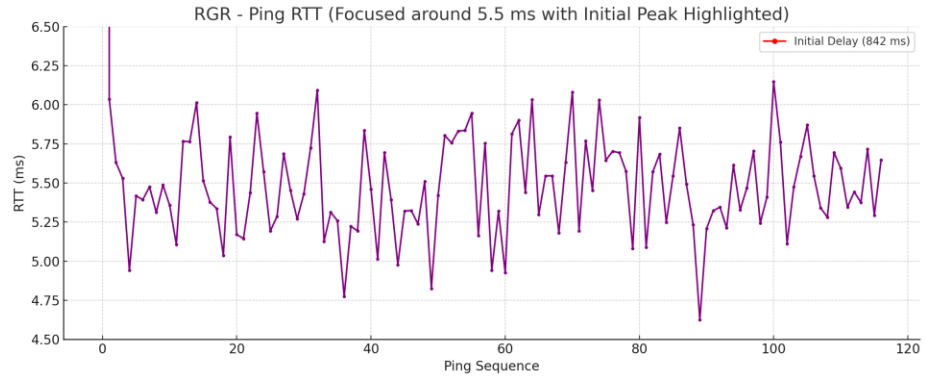
How It Works

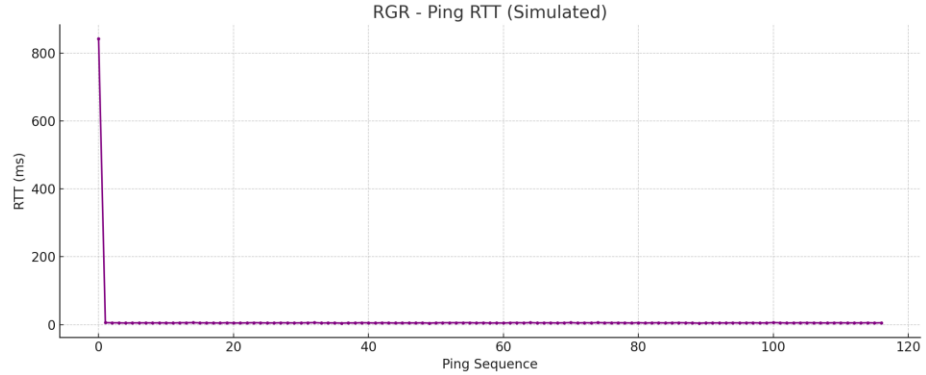
- **Reactive Mode (Start & End)**
  - **AODV-style** route discovery is used when no known path exists.
- **Greedy Mode (Middle)**
  - Once the route is partially discovered, data is forwarded using **geographic greedy forwarding** toward the destination.
- **Back to Reactive**
  - If greedy forwarding fails (e.g., local minimum), protocol falls back to reactive discovery.

Result

- ----------
- OsgIndoorNet.user[0].app[0]
- ----------

- sent: 117 received: 117 loss rate (%): 0
- round-trip min/avg/max (ms): 5.48931/12.6974/842.001
- stddev (ms): 77.331 variance:0.00598008
- ----------





Results

Protocol	Packet Loss Rate (%)	Average RTT (ms)	Maximum RTT (ms)	Standard Deviation (ms)
GPSR	0	5.89	8.56	0.41
DYMO	0	14.41	1042.3	95.85
DSDV	11.4	4.63	7.28	0.41
AODV	0	12.7	842	77.33
RGR	0	12.7	842	77.33

1. GPSR (Greedy Perimeter Stateless Routing)

- **Characteristics:** Uses geographic information for greedy forwarding. No route discovery is needed.
- **Results:**
  - Consistently **low RTT from the beginning** (average: 5.89 ms)
  - **Very low standard deviation** (0.41 ms) — indicating highly stable response times
  - **No packet loss**

2. DYMO (Dynamic MANET On-demand)

- **Characteristics:** A successor to AODV. **Reactive** protocol that discovers routes on-demand.
- **Results:**
  - The **first ping** suffered from a **very high RTT (1042 ms)** due to route discovery
  - Afterwards, RTT stabilized at around **5 ms**
  - **No packet loss**

3. AODV (Ad hoc On-demand Distance Vector)

- **Characteristics:** Similar to DYMO, a **reactive on-demand** routing protocol
- **Results:**
  - The **first packet** showed a **delay of 842 ms** — due to route discovery
  - RTT after that was **very stable**, around **5 ms**
  - **No packet loss**

4. DSDV (Destination-Sequenced Distance Vector)

- **Characteristics:** A **proactive** routing protocol that maintains routing tables continuously
- **Results:**
- Achieved the **lowest average RTT (4.63 ms)**
- ⚠️ However, had a **packet loss rate of 11.4%**

Conclusion

GPSR had the best overall performance — fast, stable, and no packet loss.

DYMO, AODV, and RGR were a little slower at first, due to route discovery, but then very reliable.

DSDV had fast responses, but the packet loss makes it less suitable for changing environments like UAV networks.

RGR Application Problem

RGR protocol is supposed to be better than AODV

The results are still identical to AODV or sometime worse than AODV.

For example, when AODV had 1% of loss rate, RGR had 3% of loss rate in the same configuration.

We’re actively modifying the implementation and analyzing related works to improve it.

Reference

- [1] F. Liang, J. Zhang, B. Li, Z. Yang and Y. Wu, "The Optimal Placement for Caching UAV-assisted Mobile Relay Communication," 2019 IEEE 19th International Conference on Communication Technology (ICCT), Xi'an, China, 2019, pp. 540-544, doi: 10.1109/ICCT46805.2019.8947051.
- [2] M. Mozaffari, W. Saad, M. Bennis and M. Debbah, "Efficient Deployment of Multiple Unmanned Aerial Vehicles for Optimal Wireless Coverage," in IEEE Communications Letters, vol. 20, no. 8, pp. 1647-1650, Aug. 2016, doi: 10.1109/LCOMM.2016.2578312.
- [3] I. Bekmezci, O. K. Sahingoz and Ş. Temel, "Flying Ad-Hoc Networks (FANETs): A survey," Ad Hoc Networks, vol. 11, no. 3, pp. 1254–1270, May 2013, doi: 10.1016/j.adhoc.2012.12.004.

What are Fly ad-hoc networks(FANETS)

Wireless decentralized networks
Made up of drones

Primary Uses:
- -Emergencies
- -Military

Fast moving and quick to deploy.

Types of Routing Protocols

- 1. Network Architecture-Based Routing Protocols:
- **Topology-Based:**
  - **Proactive Routing:** Protocols like OLSR (Optimized Link State Routing) establish routes proactively, meaning they maintain route information in advance.
  - **Reactive Routing:** Protocols like AODV (Ad hoc On-Demand Distance Vector Routing) and DSR (Dynamic Source Routing) establish routes only when needed, reacting to requests.
- **Position-Based:**
- **GRP (Geographic Routing Protocol):** Uses the geographic location of nodes to determine the optimal route.
- **Hierarchical:**
- **Cluster-Based:** Divides the network into clusters with cluster heads, enabling efficient data forwarding.
  - 2. Data Forwarding-Based Routing Protocols:
- **Static Routing:**
  - Routes are pre-determined and do not change during operation, offering simplicity but limited adaptability.
- **Data-Centric Routing (DCR):**
- Focuses on the data requested by nodes, suitable for one-to-many transmissions in cluster-based topologies.
- **Load Carry and Deliver Routing (LCAD):**
- UAVs transport data directly from a source to a destination ground station, offering security and high throughput but potentially high latency over long distances.
- **Multi-Path Routing:**
- Establishes multiple paths between source and destination, enhancing fault tolerance, bandwidth, and security, but can be complex to manage.
- **Swarm Intelligence Routing:**
- Utilizes algorithms inspired by swarm intelligence, where multiple agents interact to find optimal routes.
- **AI-Enabled Routing:**
- Employs machine learning to model and predict network topology, channel status, traffic mobility, and environmental factors for enhanced routing.

Step By Step

- **Route Discovery (Reactive Start):**
- When a **source node** wants to send data but doesn't know the route, it initiates a **reactive route discovery**.
- It may use position info (e.g., via GPS or a location service) to get the **destination's coordinates**.
- **Greedy Forwarding:**
- Once the destination's position is known, the source starts **greedy forwarding**.
- Each node forwards the packet to its **neighbor closest to the destination** (based on geographic distance).
- This minimizes the number of hops and speeds up delivery.
- **Local Minimum Problem:**
- Sometimes, a node may find **no neighbor** closer to the destination than itself. This is called a **local minimum**.
- At this point, greedy forwarding **fails**.
- **Recovery Phase (Back to Reactive):**
- When greedy forwarding fails, the protocol switches back to **reactive mode**.
- It may trigger a local route discovery or use **face/perimeter routing** to bypass the obstacle.
- Once it finds a new path around the problem, it returns to greedy forwarding.
- **Delivery:**
- The packet continues using this hybrid logic until it **reaches the destination**.