PathGeometry - Quick Start Guide

Installation

No dependencies required - pure Python 3.8+

bash

```
# Just copy the file
cp path_geometry.py your_project/
```

Basic Usage

1. Calculate Path Distance and Azimuth

python

```
from path_geometry import PathGeometry, GeoPoint

# Create locations (latitude, longitude in degrees)
tx = GeoPoint.from_degrees(44.65, -63.57)  # Halifax, NS
rx = GeoPoint.from_degrees(51.51, -0.13)   # London, UK

# Calculate path
path = PathGeometry()
path.set_tx_rx(tx, rx)

# Get results
print(f"Distance: {path.get_distance_km():.1f} km")
print(f"Azimuth: {path.get_azimuth_tr_degrees():.1f}°")
```

Output:

```
Distance: 4622.6 km
Azimuth: 57.0°
```

2. Calculate Hop Parameters

python

```python
from path_geometry import hop_distance, RinD, EarthR

# Parameters
elevation = 10 * RinD  # 10 degrees in radians
virt_height = 300      # km

# Calculate hop distance
hop_dist_rad = hop_distance(elevation, virt_height)
hop_dist_km = hop_dist_rad * EarthR

print(f"Hop distance: {hop_dist_km:.1f} km")
print(f"Hops needed: {path.hop_count(elevation, virt_height)}")
```

Output:

```
Hop distance: 2192.9 km
Hops needed: 3
```

3. Get Points Along Path

python

```
# Get midpoint
midpoint = path.get_point_at_dist(path.dist / 2)
lat, lon = midpoint.to_degrees()

print(f"Midpoint: {lat:.2f}°, {lon:.2f}°")
```

Output:

```
Midpoint: 52.61°, -34.21°
```

Complete Example

python

```python
from path_geometry import PathGeometry, GeoPoint, hop_distance, RinD, EarthR

def analyze_path(tx_lat, tx_lon, rx_lat, rx_lon):
    """Analyze HF propagation path"""

    # Create path
    tx = GeoPoint.from_degrees(tx_lat, tx_lon)
    rx = GeoPoint.from_degrees(rx_lat, rx_lon)

    path = PathGeometry()
    path.set_tx_rx(tx, rx)

    # Calculate parameters
    elev = 10 * RinD
    height = 300
    hop_dist = hop_distance(elev, height) * EarthR
    hops = path.hop_count(elev, height)

    # Print results
    print(f"Path Analysis:")
    print(f"  Distance:     {path.get_distance_km():.0f} km")
    print(f"  Azimuth:      {path.get_azimuth_tr_degrees():.1f}°")
    print(f"  Hop distance: {hop_dist:.0f} km")
    print(f"  Hop count:    {hops}")

    return path

# Example usage
analyze_path(44.65, -63.57, 51.51, -0.13)
```

API Reference

GeoPoint Class


python

```
GeoPoint(lat, lon)                    # lat/lon in radians
GeoPoint.from_degrees(lat_deg, lon_deg)  # Create from degrees
point.to_degrees()                    # Convert to degrees
```

PathGeometry Class


python

```
path = PathGeometry()
path.set_tx_rx(tx_point, rx_point)      # Set locations
path.get_distance_km()                  # Distance in km
path.get_azimuth_tr_degrees()           # Azimuth Tx->Rx
path.get_azimuth_rt_degrees()           # Azimuth Rx->Tx
path.get_point_at_dist(distance_rad)    # Point along path
path.hop_count(elev_rad, height_km)     # Number of hops
```

Functions


python

```
hop_distance(elev_rad, height_km)        # Hop distance (radians)
hop_length_3d(elev, hop_dist, virt_height) # 3D slant distance
calc_elevation_angle(hop_dist, height)   # Elevation from hop
sin_of_incidence(elev, height)           # Sin of incidence
cos_of_incidence(elev, height)           # Cos of incidence
```

Constants


python

```
RinD = π/180    # Convert degrees to radians
DinR = 180/π    # Convert radians to degrees
EarthR = 6370   # Earth radius in km
```

Testing

bash

```bash
# Run full test suite
python3 test_path_geometry.py

# Run examples
python3 path_geometry.py
python3 integration_example.py
```

Common Patterns

Multiple Targets

python

```python
targets = [
    ('London', 51.51, -0.13),
    ('Tokyo', 35.68, 139.65),
    ('Sydney', -33.87, 151.21)
]

tx = GeoPoint.from_degrees(44.65, -63.57)

for name, lat, lon in targets:
    rx = GeoPoint.from_degrees(lat, lon)
    path = PathGeometry()
    path.set_tx_rx(tx, rx)
    print(f"{name}: {path.get_distance_km():.0f} km at {path.get_azimuth_tr_degrees():.0f}°")
```

Reflection Points

python

```python
# Get reflection points for each hop
elev = 10 * RinD
height = 300
hop_dist = hop_distance(elev, height)

path = PathGeometry()
path.set_tx_rx(tx, rx)

hops = path.hop_count(elev, height)
for i in range(1, hops):
    point = path.get_point_at_dist(i * hop_dist)
    lat, lon = point.to_degrees()
    print(f"Hop {i}: {lat:.2f}°, {lon:.2f}°")
```

Files in This Package

File	Purpose
`path_geometry.py`	Main module
`test_path_geometry.py`	Test suite
`integration_example.py`	Integration examples
`PATHGEOMETRY_PORT_SUMMARY.md`	Full technical docs
`PROJECT_STATUS.md`	Project roadmap
`QUICK_START.md`	This file

Getting Help

- Check test file for examples: `test_path_geometry.py`
- See integration examples: `integration_example.py`
- Read full documentation: `PATHGEOMETRY_PORT_SUMMARY.md`

Next Steps

1. ✅ Basic calculations (you are here)
2. → Integrate with existing system
3. → Port additional VOACAP modules
4. → Enhance dashboard with path visualization

Ready to use! Copy `path_geometry.py` to your project and start calculating! 🚀