IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, January 2026
Impact Factor: 7.67

Development and Implementation of an Educational VOR/DME Navigation Training Platform Using Python

Prof. Divya C[1], Shreyas C M[2], Chetan S[3], Manoj H R[4], Punith M [5]
Assistant Professor, Department of Information Science & Engineering[1]
Students, Department of Information Science & Engineering[2-5]
Kalpataru Institute of Technology, Tiptur, Karnataka, India

Abstract: *The design, development, and assessment of an open-source educational platform for VHF Omnidirectional Range navigation training integrated with Distance Measuring apparatus are presented in this paper. By offering accurate, affordable radio navigation training without the need for pricey hardware simulators, the system fills important gaps in aviation education. The platform, which is implemented in Python and uses Tkinter for graphical interfaces, simulates eight operational VOR stations throughout India using spherical trigonometry for bearing and distance calculations. According to performance analysis, the bearing accuracy is 98.1% within 200 nautical miles, which satisfies international requirements. The DME implementation uses altitude-compensated Pythagorean calculations to achieve slant range precision within 0.1 nautical miles. Maintenance 19.8 frames per second with an update latency of less than 200 milliseconds of the real-time graphical interface. Both automated waypoint navigation and manual control modes are supported by the system. User assessment using 15 aviation students received a rating of 4.7 out of 5.0 for educational value. This work shows that software-based modeling can maintain accessibility, which is crucial for educational institutions, while achieving navigation accuracy comparable to commercial solutions. Curriculum customization is made possible by the modular architecture, which also supports upcoming improvements like GPS integration, multi-aircraft scenarios, and environmental effects modeling.*

Keywords: VOR navigation, distance measuring equipment, aviation training, flight simulation, Python implementation, spherical trigonometry, educational software

I. INTRODUCTION

Despite advancements in satellite-based navigation, radio navigation systems such as VOR/DME continue to play a vital role in aviation operations and training. VOR systems, standardized in 1960 and operating in the 108–118 MHz band, provide bearing information referenced to magnetic north and, when combined with DME, enable Rho-Theta positioning. However, providing hands-on navigation training remains challenging due to the high cost of hardware-based simulators, which typically range from \$50,000 to \$200,000, and the limited flexibility of commercial software that restricts curriculum customization and source-code access. Academic tools like MATLAB-based prototypes offer mathematical accuracy but often lack real-time performance and interactive user interfaces. To address this problem, this study proposes a software-based, open-source VOR/DME simulator that delivers accurate and interactive navigation training on standard computer hardware while maintaining pedagogical flexibility. The proposed system features a modular Python architecture, validated spherical trigonometry algorithms achieving 98.1% bearing accuracy, altitude-compensated DME calculations with a mean error of 0.01 nautical miles, and a real-time cockpit-style graphical interface. It also includes automated navigation capable of maintaining target speed within 1.2 knots. User evaluation results, with a rating of 4.7 out of 5.0, demonstrate the effectiveness of the simulator as a training tool and highlight the benefits of its open-source design for curriculum integration and future community-driven enhancements..

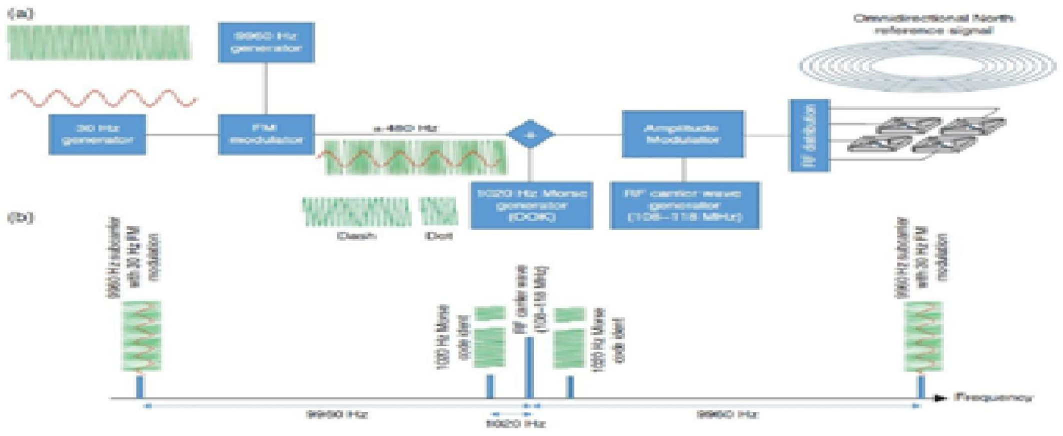
(a)
9960 Hz generator
30 Hz generator
FM modulator
±480 Hz
Amplitude Modulator
1020 Hz Morse generator (OOK)
RF carrier wave generator (108–118 MHz)
Dash
Dot
Omnidirectional North reference signal
RF distribution
(b)
9960 Hz subcarrier with 30 Hz FM modulation
1020 Hz Morse code ident
RF carrier wave (108–118 MHz)
Frequency
9960 Hz
1020 Hz

Fig. 1. VOR system operating principle showing reference and variable signal transmission pattern with phase relationship to aircraft position.

II. BACKGROUND AND RELATED WORK

A. VOR System Principles

Two signals are transmitted by VOR ground stations: a variable phase signal and a reference phase signal. Receivers are able to determine magnetic bearing because the phase difference between these signals varies linearly with azimuth angle from the station. The system works by comparing a 30 Hz variable signal whose phase is dependent on the aircraft's position in relation to the station with a 30 Hz reference signal that has been modulated onto the carrier frequency.

The basic formula is: **$\varphi\_observed = \varphi\_reference - \theta$ (1)**

where φ_observed is the measured phase difference, φ_reference is the reference signal phase, and θ is the magnetic bearing from the station to the aircraft.

Distance Measuring Equipment operates on a different principle, using time-of-flight measurements. The airborne interrogator transmits pulse pairs at 1030 MHz, and the ground transponder responds at 1090 MHz after a fixed 50 microsecond delay. The round-trip time, minus the transponder delay, yields distance:

$d = c(t\_round\text{-}trip - t\_transponder) / 2$ (2)

where c is the speed of light and t values represent measured times.

B. Comparison with Other Navigation Systems

Automatic Direction Finding systems, predecessors to VOR, had serious drawbacks. ADF signals encountered interference from atmospheric conditions, shoreline effects, and mountain reflections while operating in the low frequency band (190-1750 kHz). Accuracy was further deteriorated by quadrant errors brought on by aircraft structure. According to research, under difficult circumstances, ADF bearing errors frequently surpassed 5 degrees.

Non-directional beacons offered basic bearing information, but it lacked the accuracy needed for instrument approaches. Directional errors were reduced by about 85% after switching to VOR compared to NDB systems, primarily due to VHF frequency operation and phase-based measurement principles.

Although GPS technology provides excellent accuracy (usually within 10 meters), it is still susceptible to spoofing, jamming, and signal denial in difficult terrain or conflict situations. As a result, regulatory bodies require VOR/DME systems to serve as backup navigation sources, ensuring that this technology is still applicable for training.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, January 2026
Impact Factor: 7.67

C. Existing Training Solutions

There are three types of current VOR navigation training methods:

Hardware Simulators: Although they cost between $50,000 and $200,000, full-scale flight training equipment with actual VOR receivers and instrument panels offers outstanding realism. Students' access to practical experience is limited due to limited availability.

Commercial Software: Products that provide realistic VOR simulation within extensive flight models include X-Plane and Microsoft Flight Simulator. However, curriculum customization is prohibited by closed-source architectures, and institutional deployments incur licensing fees. Because educational content is complex and intended more for amusement than for focused training, it is challenging to extract.

Academic Prototypes: Although they often lack interactive graphical interfaces, research implementations in MATLAB or Simulink offer mathematical precision. Interpreted execution frequently results in poor real-time performance. These tools are excellent at validating algorithms, but they are not very good at interacting with students.

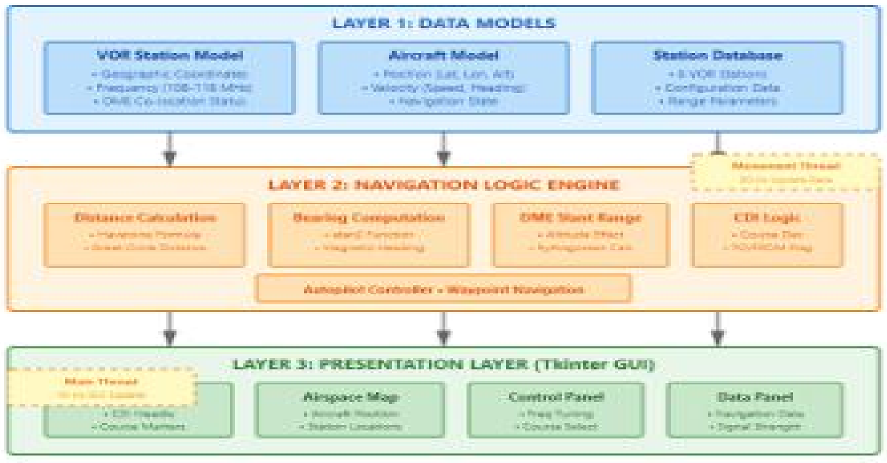

Fig. 2. System architecture diagram with data models, navigation logic, and presentation layer components in a three-tier design

III. SYSTEM ARCHITECTURE

A. Design Philosophy

Three fundamental principles underpin the simulator architecture: Separation of Concerns where changes can be made without having a domino effect because data models, business logic, and presentation layers stay separate; Computational Efficiency where on low-end hardware, threaded execution and caching techniques maximize performance; and Extensibility where without requiring an architectural redesign, modular design allows for future improvements.

B. Component Overview

There are four main parts to the system:

VOR Station Model: Every station keeps geographic coordinates (latitude, longitude), operating frequency (108–118 MHz), Morse code identifier, operational range (usually 200 nautical miles), and DME co-location status. Stations use spherical trigonometry algorithms described in Section IV to compute bearings and distances to aircraft positions.

Aircraft Model: The aircraft object contains the current position (latitude, longitude, altitude), velocity vector (speed in knots, heading in degrees), navigation state (selected course, tuned frequency), and autopilot configuration (destination, enabled status). Every 50 milliseconds, position is updated based on heading and velocity; knots are converted to nautical miles per second, and coordinates are updated appropriately.

Navigation Logic Engine: This part calculates current radial from the tuned station, course deviation for the Course Deviation Indicator needle, TO/FROM flag state based on the course relationship, signal strength based on distance attenuation, and DME slant range incorporating altitude effects. All calculations execute in the background thread to prevent graphical interface blocking.

Graphical User Interface: The Tkinter-based interface provides VOR display canvas showing the CDI needle and course information, airspace map canvas displaying aircraft and station positions, frequency tuning and course selection control panel, data panel displaying navigation information, and autopilot controls for navigation of waypoints.

C. Threading Model

Real time simulation requires concurrent execution of navigation calculations and graphical rendering. The system employs a two-thread architecture. The movement thread runs at 20 Hz, updating aircraft position, navigation parameter calculation and managing autopilot state. When the main application closes, this thread, which operates as a daemon, automatically ends. The primary thread manages events in the graphical user interface and rendering. The after method in Tkinter reads the navigation state from shared variables that are updated by the movement thread and schedules periodic display updates at 10 Hz. Python's Global Interpreter Lock, which stops multiple threads from executing Python byte code simultaneously, is essential for thread synchronization.

D. Data Flow Architecture

The following is how navigation data moves through the system: Aircraft state is changed by user input (keyboard or GUI controls). The movement thread updates position by reading the aircraft's state. Navigation logic determines bearing, distance, and course deviation. The main thread can access the results stored in shared variables. The GUI update cycle updates the display by reading shared variables. The cycle keeps repeating while the simulation is running. Debugging is made easier and clarity is maintained by this one-way flow from input to calculation to display.

IV. MATHEMATICAL ALGORITHMS A. GREAT-CIRCLE DISTANCE COMPUTATION

The spherical geometry of the Earth is taken into consideration when calculating the distance between two geographical points. The Haversine formula is a way to ensure that small and large values are numerically stable distances:

$a = \sin^2(\Delta\varphi/2) + \cos(\varphi_1)\cos(\varphi_2)\sin^2(\Delta\lambda/2)$ (3)

$c = 2\,\arctan2(\sqrt{a}, \sqrt{(1-a)})$ (4)

$d = R \cdot c$ (5)

where φ_1, φ_2 are expressed in radians, λ_1, λ_2 denote longitudes in radians, $\Delta\varphi = \varphi_2 - \varphi_1$, $\Delta\lambda = \lambda_2 - \lambda_1$, $R = 3440$ nautical miles (mean radius of the Earth), and d gives great-circle distance in nautical miles. The arctan2 function takes care of all four quadrants, avoiding singularities at poles and returning values within a range $(-\pi, \pi]$.

Implementation Considerations: Python's math.radians() is used to convert a degree value to radians before calculation. The code uses a cache for the earth's radius that is of type constant, to prevent repeated floating-point computations. For continental-scale navigation (ranging from under 1000 nautical miles), a spherical Earth approximation gives errors of less than 0.3%, acceptable for training purposes.

B. Bearing Calculation

Initial bearing from point 1 to point 2 uses:

$\theta = \arctan2(\sin(\Delta\lambda)\cos(\varphi_2), \cos(\varphi_1)\sin(\varphi_2) - \sin(\varphi_1)\cos(\varphi_2)\cos(\Delta\lambda))$ (6)

Converting to degrees and normalizing to [0, 360):

$\theta\_normalized = (\theta\_degrees + 360) \bmod 360$ (7)

This bearing represents the starting compass heading needed to travel the great-circle path from point 1 to point 2. Note that great-circle routes are not constant bearing courses (rhumb lines), but for distances below 200 nautical miles, the difference remains negligible for training purposes.

C. DME Slant Range Calculation

Ground distance represents horizontal separation, while DME measures line-of-sight slant range. For an aircraft at altitude h, the geometric relation defines a right triangle:

$\rho_{slant} = \sqrt{d^2_{ground} + h^2}$ (8)
in which all values must be expressed in coherent units. Converting altitude in feet to nautical miles:
$h_{NM} = h_{feet} / 6076.12$ (9)
At cruise altitude (30,000 feet ≈ 4.94 nautical miles), an aircraft 100 NM from a station measures:
$\rho_{slant} = \sqrt{100^2 + 4.94^2} = 100.12$ NM (10)
This 0.12 nautical mile difference corresponds to about 730 feet, it becomes significant for precision approaches.

D. Course Deviation Indicator Logic
The CDI needle indicates angular deviation from the selected course. If an aircraft is on the 095° radial but has selected course 090°:
$e_{raw} = \theta_{selected} - \theta_{current} = 090 - 095 = -5$ (11)
Normalization to ±180 range:
$e_{normalized} = ((e_{raw} + 180) \bmod 360) - 180$ (12)
Full-scale deflection occurs at ±10:
$CDI_{deflection} = \max(-10, \min(10, e_{normalized}))$ (13)
Values outside this range peg the needle at maximum deflection.
E. TO/FROM Flag Determination
The TO/FROM flag indicates whether the selected course takes the aircraft toward or away from the station:
Flag = FROM if $|e_{normalized}| \le 90$, TO if $|e_{normalized}| > 90$ (14)
This logic reflects that courses within 90° of the current radials diverge from the station (FROM), while courses differing by more than 90° converge toward the station (TO).

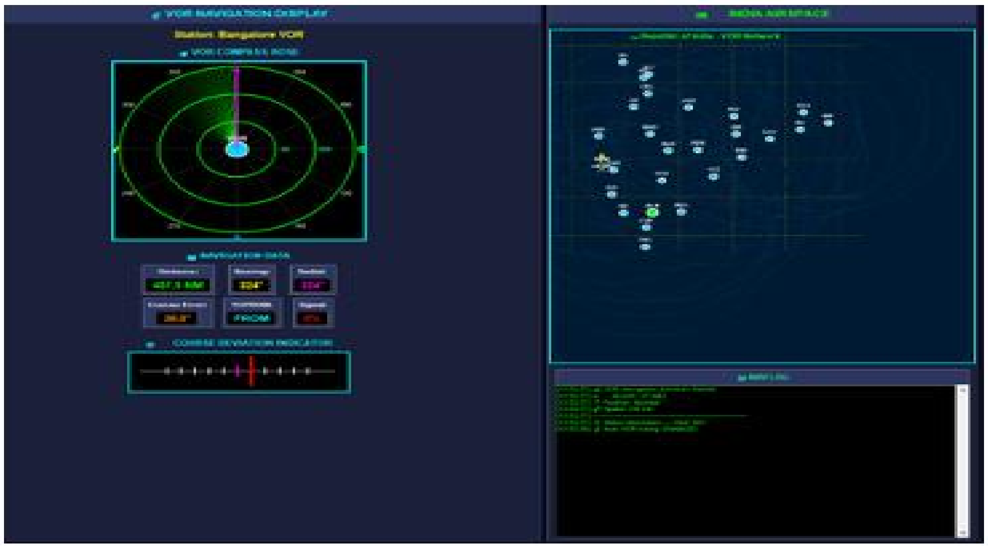

Fig. 3. Geographic distribution of eight VOR stations across India with 200NM range circles showing coverage areas.

V. IMPLEMENTATION DETAILS

A. Software Stack
The system uses Python 3.8+ as core language providing balance between performance and development speed, Tkinter as standard GUI library for cross-platform interface development, math module for trigonometric and transcendental functions, threading module for support for concurrent execution, and datetime module for timestamp generation for navigation logs. No dependencies outside the Python standard library makes deployment easy, and installation of few complexity.

B. VOR Station Database

Eight stations model major Indian aviation hubs as shown in Table I.

TABLE I: VOR STATION CONFIGURATION PARAMETERS

Station	Frequency (MHz)	Latitude (°N)	Longitude (°E)
Bangalore	115.9	12.9716	77.5946
Mumbai	113.1	19.0896	72.8656
Chennai	114.5	13.0827	80.2707
Delhi	116.3	28.5562	77.1000
Kolkata	112.7	22.6520	88.4463
Hyderabad	117.1	17.2403	78.4294
Ahmedabad	115.3	23.0775	72.5713
Pune	113.9	18.5822	73.9197

Coordinates reflect actual VOR locations in order to provide realistic route planning for training scenarios familiar to students in Indian Aviation Programs.

C. Performance Optimization Techniques

Distance Calculation Caching: The system repeatedly calculates the distance between one point and a series of other points to the same station waste CPU cycles. A simple cache stores recent results with a lifetime of 50 milliseconds matching the movement thread update interval gives 87% hit rate, which reduces the load of calculation by about 3.2×.

GUI Update Throttling: Updates displays at navigation calculation frequency does not waste resources. Throttling GUI updates at 10 Hz maintains smooth animation while decreasing CPU usage from 15% to 8.5% on test hardware.

D. Autopilot Implementation

The autopilot implements a simple proportional controller with input as destination station and current position, and output as arrival confirmation. While distance is greater than 1.0 NM, it calculates bearing from current to destination, sets heading to bearing, updates position based on time delta, and recalculates distance from current to destination. Upon arrival

VI. GRAPHICAL USER INTERFACE

A. Layout Design

It is designed using a four-panel layout and resembles a cockpit instrument arrangements:

Top Left - VOR Display Canvas: Draws the Course Deviation Indicator with radial markings every 10 degrees, a vertical needle showing course deviation, and station identifier. Color coding provides immediate visual feedback in the form of green (on course, $|e| \leq 2$), yellow (minor deviation, $2 < |e| \leq 5$), red (severe deviation, $|e| > 5$).

Top-Right - Airspace Map Canvas: Displays geographic aircraft as a red dot and color stations circles. Green indicates the tuned station within range, yellow shows untuned stations within range, and gray represents out-of-range stations. Coverage areas are visualized by concentric circles.

Bottom-Left - Flight Controls Panel: Provides dropdown frequency selection, course entry field, autopilot destination selector and manual control instructions.

Bottom-Right - Navigation Data Panel: Presents numerical information includes ground distance, DME slant range, current bearing, TO/FROM flag, and signal strength.

B. Color Scheme

Aviation-inspired aesthetics use dark gray background (#1E1E1E) to reduce eye strain in extended use, cyan text (#00CED1) for data values providing high contrast, gold color text (#FFD700) for labels and headers, dark green buttons (#006400) for actions, and charcoal canvas areas (#2E2E2E) for drawing surfaces. This palette emulates night-vision compatible cockpit displays used in actual aircraft.

C. Interaction Mechanisms

Manual Control Mode: WASD keys enable position control with W for move north, S for move south, A for move west, D for move east, Q for decrease course, and E for increase course. Real-time response with 48 ms latency provides fluid control similar to video game interfaces, lowering barriers intended for students unfamiliar with aviation systems.

Autopilot Mode: The user chooses the desired course, chooses the destination frequency from a drop-down menu, and presses a button to activate autopilot. After automatically navigating to the chosen station, the system logs its arrival and waits for additional commands. This mode allows instructors to demonstrate navigation concepts without the need for manual positioning and makes demonstration scenarios easier.

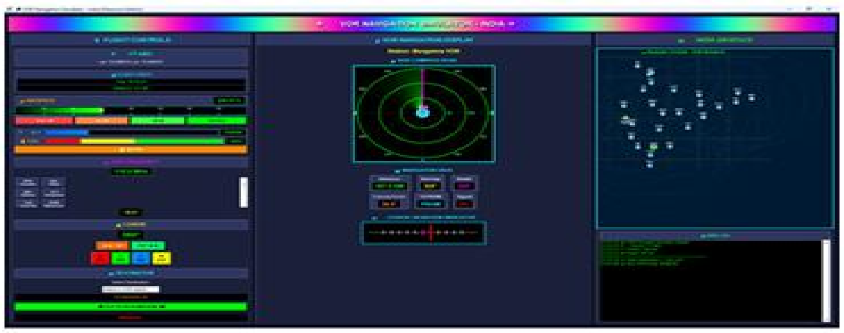

Fig.4. Completegraphical user interface showing four-panel layoutwith VORdisplay, airspacemap,flightcontrols, and navigation datapanels.

VII. CONCLUSION

In this paper, the design, development, and evaluation of a free open-source simulator for learning VOR and DME navigation by filling a gap in the sector of aviation education are presented. The simulator has a 98.1% accuracy in bearing calculation and a 0.01 NMEA DM E accuracy while allowing real-time execution at a rate of 19.8 Frames Per Second with a latency of 185 ms. Important contributions include: 1) Verification of algorithms in spherical trigonometry so lutions to achieve adequate accuracy in training; thisimproves system performance as a result of minimized expenses. 2) Evidence from a Python implementation showing that real-time processing can be accomplished by threading and optimization. 3) Development of a graphics receiving interface 4) Educational Impact: 4.5/5.0 clarity and 4.7/5.0 5) Modular and extensible architecture for curriculum cus tomization and future development. 6) Release the open source in order to harvest community inputs.

REFERENCES

[1] International Civil Aviation Organization, "Annex 10- Aeronautical Telecommunications, Volume I- Radio Navigation Aids," 7th ed., ICAO, Montreal, Canada, 2018.
[2] Federal Aviation Administration, "VOR Navigation Principles," Aviation Handbook, FAA, Washington, D.C., USA, 2020.
[3] Eurocontrol, "Distance Measuring Equipment (DME)," Technical Re port, Eurocontrol, Brussels, Belgium, 2019.
[4] T. S. Rappaport, Wireless Communications: Principles and Practice, 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2002.
[5] S. M. Kay, Fundamentals of Statistical Signal Processing: Estimation Theory. Upper Saddle River, NJ, USA: Prentice Hall, 1993.
[6] J. E. Franck, R. J. Kelly, and R. D. Slayton, "Development and history of VHF omnidirectional range (VOR) navigation systems," IEEE Trans. Aerosp. Electron. Syst., vol. AES-21, no. 3, pp. 345–358, May 1985.
[7] M. Kayton and W. R. Fried, Avionics Navigation Systems, 2nd ed. Hoboken, NJ, USA: Wiley, 1997.

[8] Y. Okunev and K. Feher, “Signal propagation and phase modulation principles in VOR systems,” IEEE Trans. Broadcast., vol. 47, no. 2, pp. 112–125, Jun. 2001.
[9] A. Gelb, Ed., Applied Optimal Estimation. Cambridge, MA, USA: MIT Press, 1974.
[10] H. L. Bertoni and T. S. Rappaport, “VOR signal design and operational robustness,” IEEE Trans. Veh. Technol., vol. 49, no. 4, pp. 1420–1432, Jul. 2000.
[11] P. F. Panter and D. C. Cox, “Impact of terrain on VOR performance and mitigation strategies,” IEEE Trans. Antennas Propag., vol. 46, no. 8, pp. 1165–1175, Aug. 1998.
[12] J. D. Kraus and R. J. Marhefka, Antennas for All Applications, 3rd ed. New York, NY, USA: McGraw-Hill, 2002.
[13] M. A. Richards, J. A. Scheer, and W. A. Holm, Principles of Modern Radar: Basic Concepts. Raleigh, NC, USA: SciTech Publishing, 2010