Wildfire Monitoring Drone Algorithm

Wildfire Monitoring Drone Algorithm

Wildfire Monitoring Drone Algorithm

Step 1: Initialization

1. Power On:
   - Initialize all sensors (GPS, LiDAR, IMU, camera).
   - Establish communication with the ground control station.
2. Calibrate:
   - Perform IMU and compass calibration.
   - Test motors and sensors for functionality.
3. Set Mission Parameters:
   - Define the wildfire monitoring area using GPS coordinates.
   - Upload waypoints or grid coordinates for the area to cover.

Step 2: Takeoff

1. Stabilize:
   - Use IMU data to stabilize the drone during liftoff.
   - Increase throttle gradually until the drone hovers at a safe altitude.
2. Safety Check:
   - Continuously monitor battery level and sensor status.

Step 3: Navigate to the Target Area

1. Path Planning:
   - Use GPS to navigate between waypoints.

2. Obstacle Avoidance:

   - Continuously scan the environment using LiDAR or ultrasonic sensors.

Step 4: Area Scanning

1. Grid-Based Scanning:

   - Divide the wildfire monitoring area into a grid of predefined size.

2. Camera Feed Analysis:

   - Capture and process real-time video frames using computer vision algorithms.

Step 5: Dynamic Navigation

1. Recalculate Path:

   - Mark GPS locations of detected fires.

2. Avoid Hazards:

   - Detect strong winds, smoke clouds, or extreme heat sources.

Step 6: Return to Home (RTH)

1. Battery Monitoring:

   - Initiate RTH when the battery reaches a predefined threshold.

2. Navigate Home:

   - Use GPS to return to the launch point.

Step 7: Landing

1. Pre-Landing Checks:

   - Confirm a safe landing area using the camera.

2. Land Safely:

    - Reduce altitude gradually while maintaining stability.

Key Components of the Algorithm

1. Wildfire Detection (Code Example):

```
import cv2

import numpy as np


def detect_fire(frame):

    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    lower_bound = np.array([0, 50, 50])  # Adjust for fire's HSV range

    upper_bound = np.array([10, 255, 255])

    mask = cv2.inRange(hsv, lower_bound, upper_bound)

    return cv2.countNonZero(mask) > 1000  # Threshold for fire detection
```

2. Obstacle Avoidance (Code Example):

```
def avoid_obstacle(sensor_data):

    if sensor_data['distance'] < 2.0:  # Example threshold (meters)

        alter_path()  # Function to change drone path
```

3. GPS-Based Path Planning (Code Example):

```
from geopy.distance import geodesic


def calculate_next_waypoint(current_location, target_location, step_size):

    # Calculate direction vector and move step_size toward target
```

Wildfire Monitoring Drone Algorithm

```
# Return next GPS coordinate
```