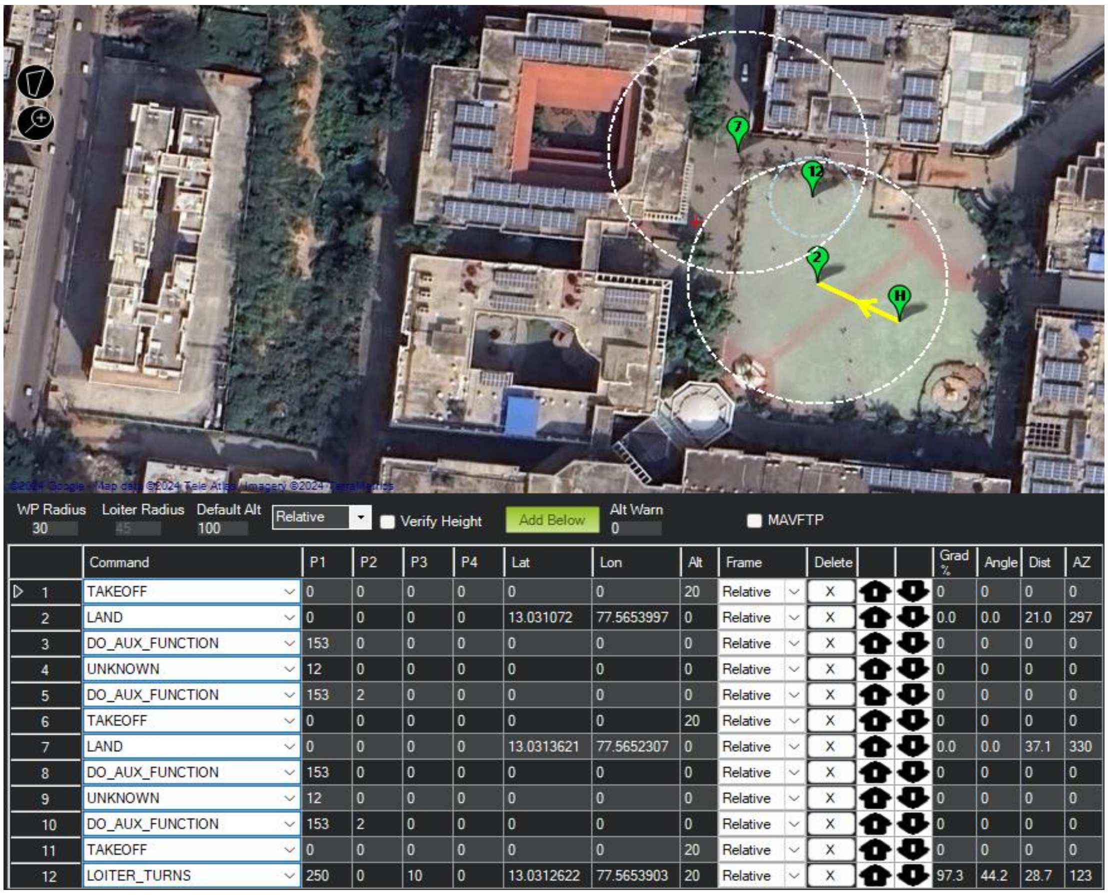
Monitoring Algorithm

ArduPilot has introduced support for Lua scripting. Scripting provides a safe, “sandboxed” environment for new behaviors to be added to the autopilot without modifying the core flight code. Scripts are stored on the SD card and run in parallel with the flight code.

The Monitoring algorithm performs

- Getting the co-ordinates of multiple charging pads in it’s proximity.
- SOC, distance, time of flight, velocity is monitored every 100 ms.
- Ensures that UAV remains in the safe proximity of the nearest charger pads so that it can travel the charging pad before it reaches the critical SOC. If it reaches the critical SOC then it finds the nearest charging pad in the proximity to dock for charging.
- During charging, the algorithm monitors the SOC so that once it’s fully charged the drone can resume to travel to the specified destination.

Configuration of destination and charging pad locations.



7
12
2
H
©2024 Google · Map data ©2024 Tele Atlas, Imagery ©2024 TerraMetrics
Add Below

WP Radius 30 | Loiter Radius 45 | Default Alt 100 | Relative | Verify Height | Alt Warn 0 | MAVFTP

	Command	P1	P2	P3	P4	Lat	Lon	Alt	Frame	Delete			Grad %	Angle	Dist	AZ
1	TAKEOFF	0	0	0	0	0	0	20	Relative	X			0	0	0	0
2	LAND	0	0	0	0	13.031072	77.5653997	0	Relative	X			0.0	0.0	21.0	297
3	DO_AUX_FUNCTION	153	0	0	0	0	0	0	Relative	X			0	0	0	0
4	UNKNOWN	12	0	0	0	0	0	0	Relative	X			0	0	0	0
5	DO_AUX_FUNCTION	153	2	0	0	0	0	0	Relative	X			0	0	0	0
6	TAKEOFF	0	0	0	0	0	0	20	Relative	X			0	0	0	0
7	LAND	0	0	0	0	13.0313621	77.5652307	0	Relative	X			0.0	0.0	37.1	330
8	DO_AUX_FUNCTION	153	0	0	0	0	0	0	Relative	X			0	0	0	0
9	UNKNOWN	12	0	0	0	0	0	0	Relative	X			0	0	0	0
10	DO_AUX_FUNCTION	153	2	0	0	0	0	0	Relative	X			0	0	0	0
11	TAKEOFF	0	0	0	0	0	0	20	Relative	X			0	0	0	0
12	LOITER_TURNS	250	0	10	0	13.0312622	77.5653903	20	Relative	X			97.3	44.2	28.7	123

Figure 1: Autonomous flight plan

- We externally specify the destination co-ordinates that UAV should travel to.
- The charging pad co-ordinates to the UAV, so that UAV can get the accurate co-ordinates that can be used in the algorithm.

The Autonomous flight plan is executed on the UAV when it is set to Autonomous mode. In Autonomous mode UAV does the follows

- First the UAV takes off
- UAV heads towards the first charger pad and acquires the charging co-ordinates
- UAV heads towards the second charging pad and acquires the charging co-ordinates
- And so on..
- After UAV has acquired all the charging co-ordinates in it's proximity, then UAV starts to head towards the specified destination.

The following state machine algorithm has 5 stages of operation

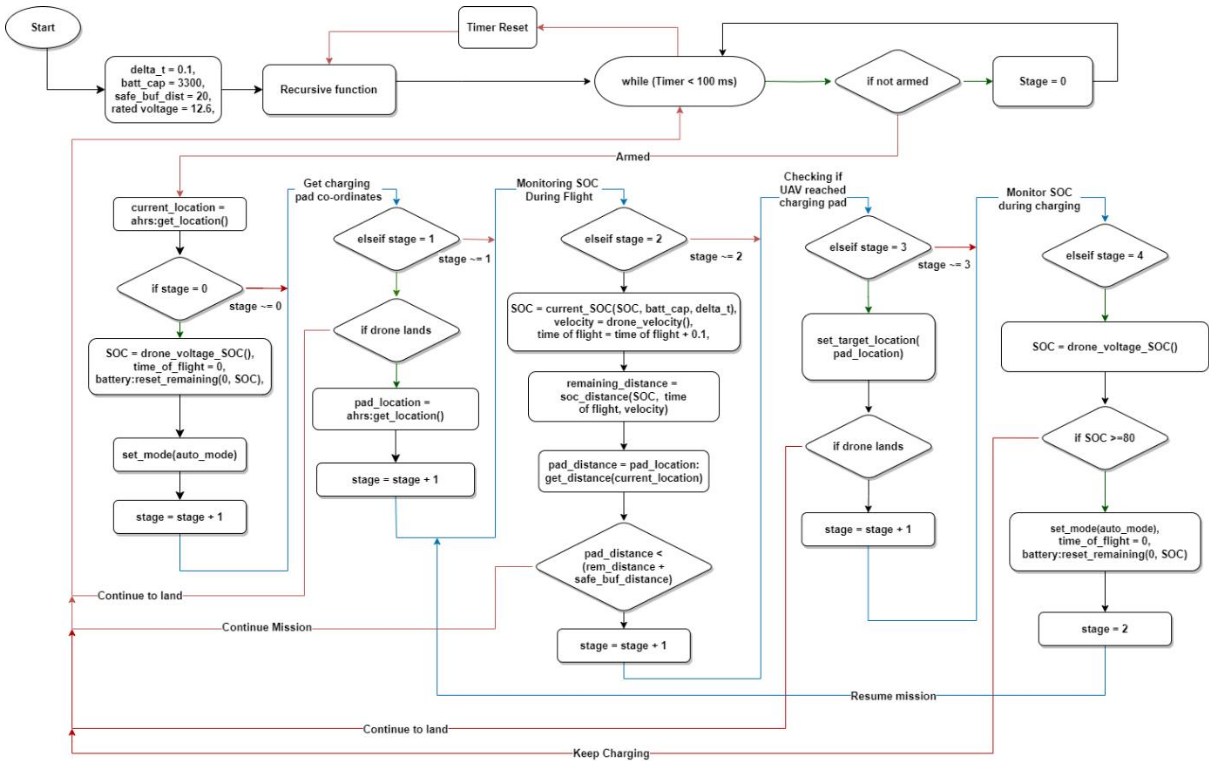

Start
delta_t = 0.1, batt_cap = 3300, safe_buf_dist = 20, rated voltage = 12.6,
Recursive function
Timer Reset
while (Timer < 100 ms)
if not armed
Stage = 0
Armed
current_location = ahrs:get_location()
if stage = 0
stage ~= 0
SOC = drone_voltage_SOC(), time_of_flight = 0, battery:reset_remaining(0, SOC),
set_mode(auto_mode)
stage = stage + 1
Get charging pad co-ordinates
elseif stage = 1
stage ~= 1
if drone lands
pad_location = ahrs:get_location()
stage = stage + 1
Monitoring SOC During Flight
elseif stage = 2
stage ~= 2
SOC = current_SOC(SOC, batt_cap, delta_t), velocity = drone_velocity(), time of flight = time of flight + 0.1,
remaining_distance = soc_distance(SOC, time of flight, velocity)
pad_distance = pad_location: get_distance(current_location)
pad_distance < (rem_distance + safe_buf_distance)
stage = stage + 1
Checking if UAV reached charging pad
elseif stage = 3
stage ~= 3
set_target_location(pad_location)
if drone lands
stage = stage + 1
Monitor SOC during charging
elseif stage = 4
SOC = drone_voltage_SOC()
if SOC >=80
set_mode(auto_mode), time_of_flight = 0, battery:reset_remaining(0, SOC)
stage = 2
Continue to land
Continue Mission
Resume mission
Continue to land
Keep Charging

Figure 2: Algorithm Flow Chart

Functions specified in the flow chart

- drone_velocity() : Calculation of velocity of the UAV using x, y, z components of the velocity vector.
- drone_voltage_SOC() : Gives the SOC based on the battery voltage using the SOC table.
- current_SOC(SOC, battery capacity, delta_t) : Gives the current SOC using coulomb counting method
- battery:reset_remaining(0, SOC) : The SOC parameter of the UAV is set to our specified SOC.
- set_mode(mode) : Used to set the UAV to autonomous mode, land mode, etc.
- ahrs:get_location() : Used to get the current location of the drone.
- soc_distance() : Calculation of the distance UAV can travel based on it's SOC.
- set_target_location(location) : Sets the target location UAV should head to.

Recursive function iterates every 100 ms, hence all the monitoring values are updated every 100 ms for accurate results. The stages are defined in the recursive function for state machine like operation.

Pre-stage operations

- All the important parameters such as battery capacity, Δt, safe buffer distance, rated voltage of battery are initialized as global variables.
- Ensured that UAV is armed and is ready for flight during the entire operation.
- Monitor the current location of the UAV.

Stage 0:

- Getting the initial SOC of the battery using the voltage-SOC function.
- Time of flight, SOC is initialized.
- The UAV is set to autonomous flight mode, and head to stage 1.

Stage 1:

- The charging pad co-ordinates is recorded by the UAV. UAV lands on each charging pad to get the accurate co-ordinates of the charging pads. After all the charging pad co-ordinates within proximity is stored by our algorithm, head to stage 2.
- Here we don't specify the co-ordinates manually as it's not accurate enough. UAV lands and gets the co-ordinates during the autonomous operation instead, as these co-ordinates much more accurate.

Stage 2:

- Monitoring of SOC is done every 0.1 s.
- Ensures UAV is in safe proximity of the nearest charger pad so that UAV can travel to the charging pad before it reaches the critical SOC. Hence, remaining distance UAV could travel should be less than some safe buffer distance + distance between UAV and nearest charging pad in it's proximity.
- If SOC of the battery reaches the critical SOC during it's mission, then it abandons the mission temporarily, and heads to the nearest charging pad in it's proximity taken care of by stage 3.

Stage 3:

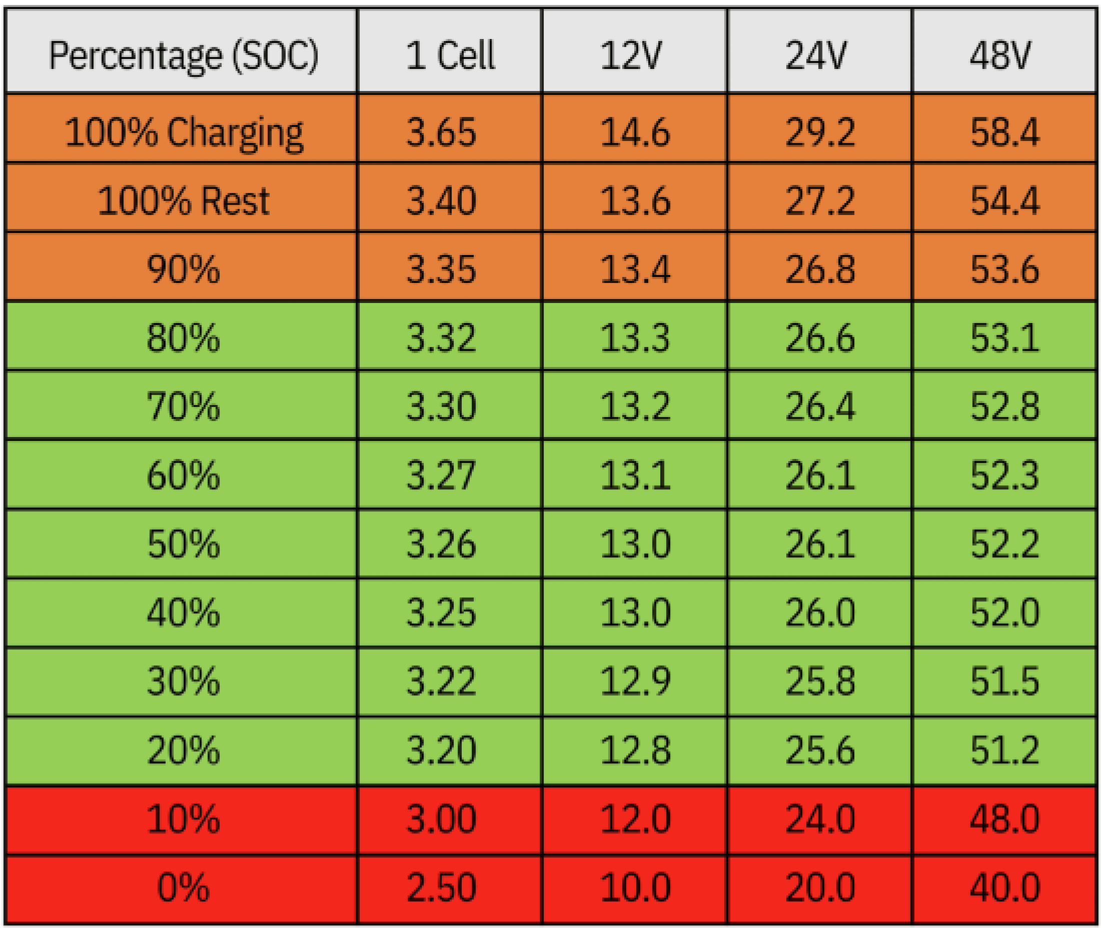
- Algorithm calculates the nearest charging pad from it's current location, hence the UAV heads towards the nearest charging pad and landing operation is performed on the charging pad.
- If UAV has landed, head on to stage 4.

Stage 4:

- Algorithm calculates the SOC while charging, so that once fully charged the mission is resumed, the UAV resumes it's journey towards the destination it temporarily abandoned.
- We resume stage 2 execution.

SOC Estimation

Initial SOC i.e., SOC(t_0) is calculated using the SOC table



Percentage (SOC)	1 Cell	12V	24V	48V
100% Charging	3.65	14.6	29.2	58.4
100% Rest	3.40	13.6	27.2	54.4
90%	3.35	13.4	26.8	53.6
80%	3.32	13.3	26.6	53.1
70%	3.30	13.2	26.4	52.8
60%	3.27	13.1	26.1	52.3
50%	3.26	13.0	26.1	52.2
40%	3.25	13.0	26.0	52.0
30%	3.22	12.9	25.8	51.5
20%	3.20	12.8	25.6	51.2
10%	3.00	12.0	24.0	48.0
0%	2.50	10.0	20.0	40.0

Figure 3: SOC Table

Then the present SOC i.e., SOC(t) is calculated using coulomb counting method

$$SOC(t) = SOC(t_0) + \frac{I(t)}{Battery\ Capacity * 1000} * \frac{\Delta t}{3600}$$

Here,

- 'Δt' (secs) = Duration of battery consumption
- 'SOC(t_0)' = Initial Charge of the battery at time t_0
- 'SOC(t)' = Present charge of the battery after time '$t = t_0 + \Delta t$'
- 'I(t)' (Amps)= Current consumption during the time 'Δt'. Current consumption is considered as -ve.
- 'Battery Capacity' (mAh)

In the algorithm,

- The SOC is monitored every 0.1 secs. Hence, ‘Δt’ is 0.1 secs
- Initially ‘SOC(t_0)’ is configured by the SOC table given above. After the first iteration of algorithm, ‘SOC(t_0)’ is the configured to be ‘SOC(t - 1)’ i.e., SOC(t) calculated in the previous iteration of the algorithm.
- SOC(t) is calculated every 0.1 secs in the algorithm, hence gives very accurate SOC estimation of the battery.

Calculation of distance, drone could travel from proximity of the nearest charger pad

The safe distance the drone could travel to nearest charging pad before reaching critical SOC is,

$$Remaining\ Distance = \frac{Remaining\ Battery\ Capacity}{Energy\ consumption\ Rate}$$

Here,

- Remaining Battery Capacity (Ah) is Calculated using SOC estimation which is given by,
  $$Remaining\ Battery\ Capacity = Battery\ Capacity * 1000 * \frac{SOC(t)}{100}$$
- Energy consumption Rate (watt-hrs/meter) is given by,
  $$Energy\ consumption\ Rate = \frac{Energy\ consumption}{Distance\ travelled}$$

Calculation of Energy consumption Rate

To calculate Energy consumption Rate, we have to calculate Distance travelled (meters) and Energy consumption (watt-hrs)

$$Distance\ Travelled = Velocity * Time\ of\ flight$$

Here,

- Velocity (m/s) is the net velocity of the UAV. It is updated every 0.1 seconds in the algorithm. The velocity is calculated using the x, y, z velocity vector components
  $$velocity = \sqrt{x\_velocity^2 + y\_velocity^2 + z\_velocity^2}$$
- Time of flight is updated every 0.1 secs i.e., every iteration of the algorithm.
  $time\ of\ flight = n * 0.1$; here n is no. of iterations of the algorithm during flight

$$Energy\ consumption = Power\ consumption * \frac{Time\ of\ flight}{3600}$$

Here,

- Power consumption (watt) is given by
  $power\ consumption = voltage * current$

- Time of flight is updated every 0.1 secs i.e., every iteration of the algorithm.
  $time\ of\ flight\ = n * 0.1$; here n is no. of iterations of the algorithm during flight