

Conditional Predictions Documentation — Sondehub

Summary

Using an existing site called CUSF Sondehub, standard or float predictions can be made with latitude, longitude, launch altitude, launch time, launch date, ascent rate, maximum altitude, and descent rate inputs. Conditional predictions are made by stopping the balloon's flight path and initiating descent when a certain condition is met. The only condition that can be manipulated in Sondehub is completing a certain duration of the float path at a constant float altitude. It will be referred to as "partial-float" in this documentation.

Links

SondeHub: <https://predict.sondehub.org/>

Download Google Earth: <https://www.google.com/earth/versions/>

GPS Visualizer (For Google Earth): https://www.gpsvisualizer.com/map_input?form=googleearth

Steps for a Partial-Float Prediction

- Identify latitude, longitude, launch altitude, launch time, launch date, ascent rate, float altitude, and descent rate of starting position
 - For UMN Ballooning, the ascent rate is 5 m/s, the launch altitude is ~300 m, and the descent rate is 7 m/s
- Start a model with float prediction selected
 - Click on "CSV" and open the file in Microsoft Excel
 - The float duration can be examined in the "datetime" column
 - ***For this example, a float duration of 20 min will be used - Because the Sondehub file increases by 20 min in the float period, this is the only float duration that can be used***
 - Splice the data when it reaches a float duration of 20 min
- Start a new model with standard prediction selected
 - Enter in the latitude, longitude, and launch time from the last data point in the spliced float prediction data set

- Enter the launch altitude using the prior float altitude
- Enter in a burst altitude in the prior float altitude slot
- ***KEEP EVERY OTHER PARAMETER CONSTANT***
- Click on “CSV” and open the file in Microsoft Excel
- Merge the 2 data sets together—float prediction first and standard prediction last—and save this file
- Open the GPS Visualizer site
 - Upload the saved Microsoft Excel file
 - In “track options”, choose clamped to the ground for the “altitude mode”
- Download the resulting KML file and open it on Google Earth

Conditional Predictions — Python Executable

Summary

Using the CUSF Sondehub site as a basis, a Python Executable was developed to run more types of conditional predictions. Now, in addition to float duration, these conditions can also be approaching the barrier of a gps fence (a rectangular boundary of certain latitude and longitude inputs) and meeting a certain real flight time. These conditions will be referred to as “gps fence” and “real time”, respectively.

Source Code Download Steps

- Go to UMN Ballooning → Float Predictions → Conditional Predictions → Executable
- Download the *conditional_predictions_source.py* and *reduce_query_time.py* files and save them in the same location
- Install the *simplekml* and *pyproj* libraries in Command Prompt

Source Code Explanation Videos

- [Part 1](#) (Initializing the Variables)
- [Part 2](#) (While Loop Pt.1)
- [Part 3](#) (While Loop Pt.2)

Basics of the IF-STATEMENT

```
# For if there is a gps fence condition:
if gps_fence == True:
    # 0.01 is the max difference the lat or long can have with the gps fence
    if abs(lat - gps_lat_start) <= 0.01 or abs(long - gps_long_start) <= 0.01 or abs(lat - gps_lat_end) <= 0.01 or abs(long - gps_long_end) <= 0.01:
        gps_fence = False

# For if there is a real time condition:
if real_time == True and time == real_time_val:
    real_time = False

# The CONSTANT Condition --> This toggles between ascent, float, and descent states based on set parameters
if altitude < float_altitude and gps_fence == True and real_time == True:
    altitude += (ascent_rate * 60)
    if altitude > float_altitude:
        altitude = float_altitude
elif float_duration >= 0 and end_condition == False and gps_fence == True and real_time == True:
    start_condition = True
else:
    float_altitude = -1 # Sets the float altitude to an impossible amount for the first condition to run
    altitude -= get_descent_rate(altitude)

if start_condition == True and gps_fence == True and real_time == True:
    float_duration = float_duration - 1
    altitude = altitude

if float_duration <= 0 or gps_fence == False or real_time == False:
    start_condition = False
    end_condition = True
    float_altitude = final_altitude
    float_duration = 1 # Sets the float duration to an impossible amount for this condition to run again
```

- These statements are about how the gps fence, real time, and partial-float conditions are implemented.
- The partial-float condition, with the float duration and float altitude, is called as the constant condition. Even if the gps fence and real time conditions are set to false, as long as there are values for the float duration and float altitude the code will execute. This is also why the float duration and float altitude must have values.
- The gps fence and real time conditions are shown above the constant condition, and they are essentially “added” on to the constant condition. If even one of the conditions happens, whether the gps fence, real time, and constant condition, the descent is initialized immediately.

```
# Booleans to start and end the condition for the prediction
start_condition = False
end_condition = False
```

- These booleans are initialized to start and end whatever condition(s) are being tested in the code

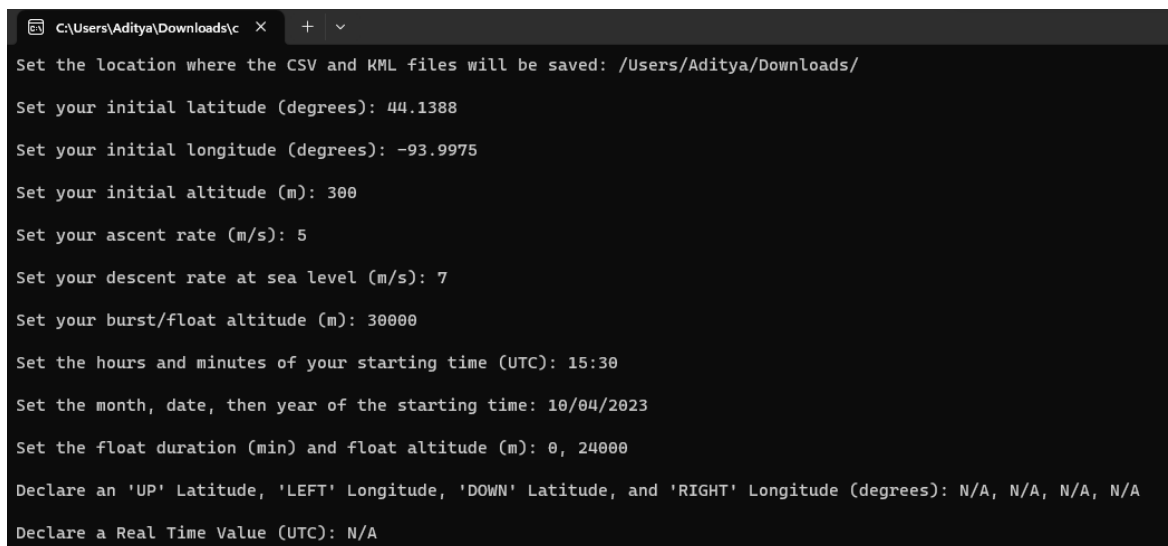
```
# Leave these booleans alone --They are for determining if the gps fence or real time conditions are being met
gps_fence = True
real_time = True
```

- These booleans are simply placeholders to determine if the gps fence or real time conditions are being met; ***DON'T CHANGE THEM***

Windows Executable Download Steps

- Go to UMN Ballooning → Float Predictions → Conditional Predictions → Executable
- Download and open the *conditional_predictions.exe* file
- (The *conditional_predictions_input* file is the *conditional_predictions_source* file formatted with user inputs so it can be made into an executable)

Windows Executable Explanation Image



```

C:\Users\Aditya\Downloads\c X
Set the location where the CSV and KML files will be saved: /Users/Aditya/Downloads/
Set your initial latitude (degrees): 44.1388
Set your initial longitude (degrees): -93.9975
Set your initial altitude (m): 300
Set your ascent rate (m/s): 5
Set your descent rate at sea level (m/s): 7
Set your burst/float altitude (m): 30000
Set the hours and minutes of your starting time (UTC): 15:30
Set the month, date, then year of the starting time: 10/04/2023
Set the float duration (min) and float altitude (m): 0, 24000
Declare an 'UP' Latitude, 'LEFT' Longitude, 'DOWN' Latitude, and 'RIGHT' Longitude (degrees): N/A, N/A, N/A, N/A
Declare a Real Time Value (UTC): N/A

```

- Set the parameters of the executable in ***EXACTLY THIS FORMAT***
- Problem ***TO BE INVESTIGATED***: The executable freezes for maybe 3-4 minutes every run, but this does not happen in the original code