PhotoFlightPlan V1.00
A free (GPL license) Python 3.5 photographic flight plan generation utility for manned aircrafts
By André Verville, Kildir Technologies

Purpose:

There are many photographic flight plan generation programs that are provided by UAV equipments providers or or also from the open source community. A good example of the second type is the excellent Mission Planner software from Michael Oborne that provides many options for UAV grid-based photographic surveys using automated waypoints navigation.

Recently involved producing the same kind of aerial photo coverages for manned aerial aircrafts, we have found that commercial software is expensive, complex and most often relies on Windows-based environments and proprietary equipments. Although there may exist some, we have found nothing that could be open, free and inexpensive to build and use. PhotoFlightPlan is provided under a GPL license, it is free to use and modify by the aerial photography industry.

Photogrammetry requirements are universal and pretty straightfoward but manned aircrafts have their own requirements that differ from fixed and rotating wings UAVs. Amongst them, the flying altitude constraint that can be imposed by flight traffic control. There is also the pilots' natural preference of flying upwind to minimize ground speed and limit yaw and direction corrections that induce roll movements to the aircraft. Those constraints make it such that often, there is a need to recompute the flight planning at the last minute.

After a few interactive flight plannings where I used Mission Planner to compute waypoints, then copy/paste them to text files and reformat/export the latitudes and longitudes to Google Earth and a GPS-based flight navigation assistant I have built, called Collimator, I decided that the time had come to automate the generation of those flight plans. I built photoflightplan.py for this purpose, bearing in mind that it needs to be simple, straightforward but also rigorous, photogrammetrically speaking. For now, it is a rather simple « terminal » based application. I will work out a GUI wrapper that shall make it even easier to use in this all-web era. In practice, a Web service could be developed quite easily around it. Anyhow, Python is simple to use and most of all, it is either already pre-installed or can be installed easily and freely on most computers. There is even a version for IOS, called Pythonista that makes photoflightplan.py something someone can use on an iPhone or iPad. Too bad folks, it costs a few dollars there :-(

The reader should bear in mind that the flight plan generated by photoflightplan.py is meant to produce a data file that can also be used as input by the Collimator flight navigation assist system (another parent project). Collimator requires special equipment to be used onboard an aircraft (Raspberry Pi computer, GPS, display and control unit, color LED strip) but the acquisition cost of the equipment is just a few hundreds of dollars, which is almost nothing in this world of aircraft-grade systems.

Characteristics:

- Python 3 based, can be executed on any platform, even on an IOS iPhone or iPad (Pythonista IOS app required).
- Takes all its operating parameters **from a user-created text** file named « photoflightplan.par ». An example parameters file is provided in the software package.
- The user provides the AOI (Area Of Interest) polygon points in latitudes and longitudes within the parameters file or else provides the name of kml files that contains these polygons (one kml per AOI polygon).

- PhotoFlightPlan takes camera parameters for any camera model, but the user needs to know and provide specific parameters values of his/her camera: focal length, sensor size and number of pixels on each axis.
- The user provides overlap and sidelap as percentages of the image size in each direction, where overlap is measured along the flying line and sidelap across it towards the other flight lines. These values are critical for photogrammetric image processing software packages like Pix4Dmapper or PhotoScan and are generally either imposed or suggested by the image processing software or past experience with a certain type of coverage (urban, agricultural, forest, etc).
- The flight lines are oriented towards (upwind) the provided wind direction, with entry point computed downwind. This is the simplest way of imposing flight lines orientation to PhotoFlightPlan.
- Multiple AOIs can be specified, PhotoFlightPlan generates a flight block for each of them. The same AOI can also be provided many times, each with a different wind direction. PhotoFlightPlan will then generate many flight blocks, each with a different flight line orientation.
- PhotoFlightPlan generates a kml file (photoflightplan.kml) that can be used to quickly visualize the results on Google Earth (all platforms).
- PhotoFlightPlan also generates a .pfp (photo flight plan) file that summarizes all project info and serves as input to the Collimator flight navigation assistant.

The parameters file (flightplan.par):

- It contains the input parameters on a line by line basis. The rules to write down this file are the following:
- Use a basic text editor (not a word processing) to create it or modify the example that is provided.
- Like Python scripts and many others, comments are usually preceded by the "#" character.
- The AOI polygon can be provided within the parameters file, the points are numbered from 01 to XX but the point 01 must be repeated at the end to close the polygon. When such provided, the "aoikml" parameter must be given "none" for its name to instruct PhotoFlightPlan to use those polygon points..
- Parameter names are the keywords, PhotoFlightPlan searches for the keywords in the following format: *keyword =* , at the beginning of a line.
- The value(s) for each parameter are provided within quotes as in "contents".
- Comments can be added after the quoted parameter, they can start with a "#" but anything after the quotes is just ignored.
- The "flightblock =" parameter needs to be the last of an AOI dataset. When PhotoFlightPlan reads this parameter, is interrupts reading and processes the block. It then resumes reading the parameters file, seeking for more data if there is another AOI. If no more AOI data is provided, PhotoFlightPlan ends processing when it encounters the end of the file.

Parameters description:

aoiname: the name of the Area Of Interest (AOI).

aoiextent: the value of the extent. This value is not used by PhotoFlightPlan itself but will be read by Collimator to trigger the camera before entering into the polygon, ahead of the flight line entry point and also after exiting the AOI.

aoikml: the name of a .kml file that contains the AOI polygon data. When no such file is provided and the AOI polygon data is provided within the parameters file, the keyword "none" must be provided for this parameter.

polygon: the AOI polygon points, one per line and finishing by a repeat of the first, such as the following: "01 46.450000 -71.500000". Point number must be a number between 01 and 99, latitude and longitudes are signed decimal degrees, all values separated by spaces.

altitudeaglft: this is where the user can impose an AGL (Above Ground Level) altitude in feet. When imposing an altitude, PhotoFlightplan computes a camera focal length that will give the required ground resolution, this is useful when we use a zoom lens on the camera, that can be adjusted at various focal lengths. For a fixed lens, "altitudeaglft" must be entered as "compute". PhotoFlightPlan will then compute it, using camera parameters and desired ground resolution.

lensfocal: this parameter contains the imposed lens focal length. When imposing a focal length, PhotoFlightPlan computes an AGL (Above Ground Level) flying altitude. When the user enters "compute", the focal length is computed from the imposed AGL altitude in feet.

resolution: provides the desired ground resolution for the photo coverage in centimeters. Decimals can be provided if needed. Resolution is sometimes called "Ground Sample Distance" or GSD by image processing software providers but GSD is computed from the results of the imagery itself and can be slightly different. We can then say the resolution is the "intended GSD".

sensor_xpix: provides the camera sensor resolution in pixels on its "x" (landscape) axis, usually the largest size, perpendicular to the flight direction when the camera top faces ahead.

sensor_xmm: provides the size of the camera sensor in millimeters along its "x" (landscape) axis, usually the largest size, perpendicular to the flight direction when the camera faces ahead. It is not always easy to find, because low quality camera manufacturers are not always very proud of the very small size of their sensors. Some specifications searching on the Internet may be required for certain camera types but for higher quality cameras like SLRs, they are normally well documented.

sensor_ypix: provides the camera sensor resolution in pixels on its "y" (landscape) axis, usually the smallest size, along the flight direction when the camera top faces ahead.

sensor_ymm: provides the size of the camera sensor in millimeters along its "y" (landscape) axis, usually the smallest size, along the flight direction when the camera top faces ahead. It is not always easy to find, because low quality camera manufacturers are not always very proud of the very small size of their sensors. Some specifications searching on the Internet may be required for certain camera types but for higher quality cameras like SLRs, they are normally well documented.

wind_direction: provides the planned wind direction bearing (real). The flight lines entry points will be computed to guide the aircraft into entering the flight lines in the other direction (upwind). This parameter is also used to direct PhotoFlightPlan to align the flight lines along a given direction, if per example the wind is planned to be calm enough and the user wants to minimize the number of flight lines to be along the longest axis of the AOI.

overlap: provides the desired overlap in percentage. This value will not have any effect on the flight lines themselves but will be used by PhotoFlightPlan to compute the camera trigger distance. This trigger distance is then used by Collimator to trigger the camera if such option is used. If using a time-lapse shutter triggering mechanism, this trigger distance in the report can be used to decide which time lapse to apply, given the planned aircraft speed.

sidelap: provides the desired sidelap in percentage. Sidelap percentage is very important because it is used to compute the distance between flight lines.

flightblock: provides the number of the flight block. The value is "01" if there is only one AOI but if there are many, each subsequent flightblock must be named "02", 03", etc in sequence. PhotoFlightPlan does not check those characters and treats them as as character strings but Collimator needs real sequential values with leading zeros.

<u>Processing details:</u>

In its current terminal-based configuration, PhotoFlightPlan is a Python script with the following file name: photoflightplanX.XX.py, X.XX being the version name. One can rename the file without its version number if preferred. All other files must have the name « photoflightplan » with the following extensions:

.par for the photoflightplan.par parameters file
.pfp for the photoflightplan.pfp report and flight plan file
.kml for the photoflightplan.kml AOI polygon(s) and planned flight line(s)
.log for the execution log where PhotoFlightPlan puts all execution time info and error reporting

Execution is initiated by running the script in a working directory where all the files are present, including the execution script. The following command is used:

python3 photoflightplan.py [paramsfile].par [flightplanfile].pfp [googleearthfile].kml

Be careful with past processed files: PhotoFlightPlan overwrites any existing .pfp and .kml files if they already exist. The ".par", ".pfp" and ".kml" provided filenames will be taken into account but are optional and may be provided in any order. For any of these, if no filename is provided on the command line, "photoflightplan.xxx" will be assumed.

The following is a transcript of the execution that happens with the provided example files on a Macbook Pro where PhotoFlightPlan is the directory name and vervillea the username, the command itself starting after the $ sign :

```
MBPro3-AV:Flightplan vervillea$ python3 photoflightplan1.01.py
------------------------------------------------------------
2017.01.02 13:11:53  PhotoFlightPlan V1.01 by André Verville
------------------------------------------------------------
Scanning across AOI "Quebec" for block 01 (100%) done
Found and recorded 12 flight lines for block 01
Scanning across AOI "Levis" for block 02 (100%) done
Found and recorded 7 flight lines for block 02
Flight plan(s) generated in "photoflightplan.pfp" and "photoflightplan.kml"

MBPro3-AV:Flightplan vervillea$
```

The "flightplan.log" log file

Flightplan puts run time processing information into a log file that can be consulted to recall past processes but also to obtain additional information in case of errors and abnormal termination. Here is an example of such a log file:

```
2017.01.02 13:11:53 PhotoFlightPlan V1.01: Startup
2017.01.02 13:11:53 PhotoFlightPlan V1.01: Processing block01
2017.01.02 13:11:53 PhotoFlightPlan V1.01: Using AOI polygon from Quebec.kml
2017.01.02 13:11:53 PhotoFlightPlan V1.01: Given lensfocal, flight altitude computed
2017.01.02 13:11:58 PhotoFlightPlan V1.01: 12 flight lines computed for block01
2017.01.02 13:11:58 PhotoFlightPlan V1.01: Processing block02
2017.01.02 13:11:58 PhotoFlightPlan V1.01: Using AOI polygon provided within parameters file
2017.01.02 13:11:58 PhotoFlightPlan V1.01: Given lensfocal, flight altitude computed
2017.01.02 13:12:04 PhotoFlightPlan V1.01: 7 flight lines computed for block02
2017.01.02 13:12:04 PhotoFlightPlan V1.01: Flight plan(s) generated in "photoflightplan.pfp" and "photoflightplan.kml"
2017.01.02 13:12:04 PhotoFlightPlan V1.01: Normal  End
```

photoflightplan.par example:

The following example shows a parameters file that contains two AOI (Areas Of Interest) one after the other. Parameters that are not changed in the second can be skipped but it is a best practice to repeat them. In this example, the AOI polygon points are provided within the parameters file itself but only the ones for Levis will be read from the parameters, those provided for Quebec will be ignored because a Quebec.kml file has been provided. These two AOIs are Québec City and Lévis (Kildir Technologies is located within the Lévis AOI !).

```
# ===================================
# flightplan.py V1.0x parameters file
# Kildir Technologies (kildir.com)
# ===================================

# =====================================================
# Area of interest polygon (number, latitude, longitude)
# first point repeated at the end for last polygon point
# -----------------------------------------------------
aoiname = "Quebec"  # appears in .pfp and .kml outputs
aoiextent = "250"  # extension of the AOI for photo coverage by Collimator (in meters)
aoikml = "Quebec.kml"  # if AOI provided in a kml file, "none" if points are listed below

polygon = "01 46.791865 -71.224538"
polygon = "02 46.803665 -71.203081"
polygon = "03 46.825593 -71.195158"
polygon = "04 46.822689 -71.210880"
polygon = "05 46.806997 -71.242248"
polygon = "01 46.791865 -71.224538"

# Project parameters (id, value and comments)
# -------------------------------------------
altitudeaglft = "compute"  # altitude constraint ("compute" if left open)
lensfocal = "35.0"  # lens focal length in millimiters ("compute" if left open)
resolution = "6.0"  # desired pixel ground resolution in centimeters
sensor_xpix = "8688"  # pixel width for Canon EOS 5DS (swath direction)
sensor_xmm = "36.0"  # sensor width in millimeters for Canon EOS 5DS (swath direction)
sensor_ypix = "5792"  # pixel height for Canon EOS 5DS (flight direction)
sensor_ymm = "24.0"  # sensor height in millimeters for Canon EOS 5DS (flight direction)
wind_direction = "230"  # wind direction in degrees (real)
overlap = "80"  # in percentage
sidelap = "65"  # in percentage
flightblock = "01"  # 01, 02, 03, etc (must be the last statement of a block to process)

# =====================================================
# Area of interest polygon (number, latitude, longitude)
# first point repeated at the end for last polygon point
# -----------------------------------------------------
aoiname = "Levis"  # appears in .pfp and .kml outputs
aoiextent = "250"  # extension of the AOI for photo coverage by Collimator (in meters)
aoikml = "none"  # if AOI provided in a kml file, "none" if points are listed below

polygon = "01 46.775490 -71.167708"
polygon = "02 46.786012 -71.152328"
polygon = "03 46.800394 -71.172122"
polygon = "04 46.789090 -71.187274"
polygon = "01 46.775490 -71.167708"

# Project parameters
# ------------------
altitudeaglft = "compute"  # altitude constraint ("compute" if left open)
lensfocal = "35.0"  # lens focal length in millimiters ("compute" if left open)
resolution = "8.0"  # desired pixel ground resolution in centimeters
sensor_xpix = "8688"  # pixel width for Canon EOS 5DS (swath direction)
sensor_xmm = "36.0"  # sensor width in millimeters for Canon EOS 5DS (swath direction)
sensor_ypix = "5792"  # pixel height for Canon EOS 5DS (flight direction)
sensor_ymm = "24.0"  # sensor height in millimeters for Canon EOS 5DS (flight direction)
wind_direction = "135"  # wind direction in degrees (real)
overlap = "80"  # in percentage
sidelap = "65"  # in percentage
flightblock = "02"  # 01, 02, 03, etc (must be the last statement of a block to process)
```

photoflightplan.pfp example:

The following example shows a the processing results obtained when processing the example photoflightplan.par file. The report repeats the input parameters, adding the computation results as well. The file is presented in two columns but the text in the right column follows the left one in the "photoflightplan.pfp" file.

```
================================================
2017.01.02 13:11:53 PhotoFlightPlan V1.01 output
================================================
parameters file:   photoflightplan.par
flight plan file:  photoflightplan.pfp (this file)
Google Earth file: photoflightplan.kml
flightblock:       01
aoiname:           Quebec
aoiextent:         250 m
lensfocal:         35.0 mm (imposed)
resolution:        6.0 cm
sensor_xpix:       8688 pixels
sensor_xmm:        36.0 mm
sensor_ypix:       5792 pixels
sensor_ymm:        24.0 mm
wind_direction:    230 degrees
overlap:           80%
sidelap:           65%
imagewidth:        521 m
imageheight:       348 m
altitudeaglm:      507 m (computed)
altitudeaglft:     1663 ft (computed)
triggerdist:       70 m
corridorwidth:     182 m

--------------------------------
AOI Polygon points (5):
From file: Quebec.kml
--------------------------------
polygon: 01 46.791865 -71.224538
polygon: 02 46.803665 -71.203081
polygon: 03 46.825593 -71.195158
polygon: 04 46.822689 -71.210880
polygon: 05 46.806997 -71.242248
polygon: 01 46.791865 -71.224538
--------------------------------

Flight lines (12):
-----------------------------------------------------------
flightline: 01 01 46.805228 -71.202523 46.792264 -71.225020
flightline: 01 02 46.807928 -71.201546 46.793541 -71.226512
flightline: 01 03 46.810627 -71.200569 46.794819 -71.228005
flightline: 01 04 46.813327 -71.199591 46.796096 -71.229497
flightline: 01 05 46.816015 -71.198634 46.797373 -71.230990
flightline: 01 06 46.818714 -71.197656 46.798651 -71.232483
flightline: 01 07 46.821413 -71.196678 46.799928 -71.233976
flightline: 01 08 46.824112 -71.195700 46.801194 -71.235489
flightline: 01 09 46.825135 -71.197634 46.802471 -71.236982
flightline: 01 10 46.824124 -71.203103 46.803748 -71.238475
flightline: 01 11 46.823112 -71.208571 46.805025 -71.239968
flightline: 01 12 46.814549 -71.227150 46.806302 -71.241462
-----------------------------------------------------------
```

```
================================================
2017.01.02 13:11:58 PhotoFlightPlan V1.01 output
================================================
parameters file:   photoflightplan.par
flight plan file:  photoflightplan.pfp (this file)
Google Earth file: photoflightplan.kml
flightblock:       02
aoiname:           Levis
aoiextent:         250 m
lensfocal:         35.0 mm (imposed)
resolution:        8.0 cm
sensor_xpix:       8688 pixels
sensor_xmm:        36.0 mm
sensor_ypix:       5792 pixels
sensor_ymm:        24.0 mm
wind_direction:    135 degrees
overlap:           80%
sidelap:           65%
imagewidth:        695 m
imageheight:       463 m
altitudeaglm:      676 m (computed)
altitudeaglft:     2217 ft (computed)
triggerdist:       93 m
corridorwidth:     243 m

--------------------------------
AOI Polygon points (4):
From parameters file
--------------------------------
polygon: 01 46.775490 -71.167708
polygon: 02 46.786012 -71.152328
polygon: 03 46.800394 -71.172122
polygon: 04 46.789090 -71.187274
polygon: 01 46.775490 -71.167708
--------------------------------

Flight lines (7):
-----------------------------------------------------------
flightline: 02 01 46.789901 -71.186155 46.776352 -71.166432
flightline: 02 02 46.791510 -71.183998 46.777898 -71.164181
flightline: 02 03 46.793119 -71.181841 46.779431 -71.161912
flightline: 02 04 46.794729 -71.179683 46.780976 -71.159662
flightline: 02 05 46.796338 -71.177526 46.782522 -71.157411
flightline: 02 06 46.797947 -71.175368 46.784068 -71.155160
flightline: 02 07 46.799556 -71.173211 46.785613 -71.152910
-----------------------------------------------------------
```

photoflightplan.kml (text format) example:

The following example shows the xml structure and contents of the kml file.


```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2"
xmlns:gx="http://www.google.com/kml/ext/2.2"
xmlns:kml="http://www.opengis.net/kml/2.2"
xmlns:atom="http://www.w3.org/2005/Atom">
<Document>
<name>flightplan.kml</name>
<StyleMap id="kildirtech">
</StyleMap>
<Style id="flightplan1">
<LineStyle>
<color>ffff00ff</color>
<width>3</width>
</LineStyle>
</Style>
<Style id="flightplan2">
<LineStyle>
<color>ffff0000</color>
<width>5</width>
</LineStyle>
<PolyStyle>
<color>33ff0000</color>
</PolyStyle>
</Style>
<Folder>
<name>Collimator flight plan</name>
<open>1</open>
<Placemark>
<name>AOI Quebec</name>
<styleUrl>#flightplan2</styleUrl>
<Polygon>
<tessellate>1</tessellate>
<outerBoundaryIs>
<LinearRing>
<coordinates>
-71.224538,46.791865,0 -71.203081,46.803665,0
-71.195158,46.825593,0 -71.210880,46.822689,0
-71.242248,46.806997,0 -71.224538,46.791865,0
</coordinates>
</LinearRing>
</outerBoundaryIs>
</Polygon>
</Placemark>
<Folder>
<name>Block 01</name><open>1</
open><Placemark>
<name>Line 01</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.202523,46.805228,0 -71.225020,46.792264,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 02</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.201546,46.807928,0 -71.226512,46.793541,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 03</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.200569,46.810627,0 -71.228005,46.794819,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 04</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.199591,46.813327,0 -71.229497,46.796096,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 05</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.198634,46.816015,0 -71.230990,46.797373,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 06</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.197656,46.818714,0 -71.232483,46.798651,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 07</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.196678,46.821413,0 -71.233976,46.799928,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 08</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.195700,46.824112,0 -71.235489,46.801194,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 09</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.197634,46.825135,0 -71.236982,46.802471,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 10</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.203103,46.824124,0 -71.238475,46.803748,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 11</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.208571,46.823112,0 -71.239968,46.805025,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 12</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.227150,46.814549,0 -71.241462,46.806302,0
</coordinates>
</LineString>
</Placemark>
</Folder>
<Placemark>
<name>AOI Levis</name>
<styleUrl>#flightplan2</styleUrl>
<Polygon>
<tessellate>1</tessellate>
<outerBoundaryIs>
<LinearRing>
<coordinates>
-71.167708,46.775490,0 -71.152328,46.786012,0
-71.172122,46.800394,0 -71.187274,46.789090,0
-71.167708,46.775490,0 </coordinates>
</LinearRing>
</outerBoundaryIs>
</Polygon>
</Placemark>
<Folder>
<name>Block 02</name><open>1</open><Placemark>
<name>Line 01</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.186155,46.789901,0 -71.166432,46.776352,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 02</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.183998,46.791510,0 -71.164181,46.777898,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 03</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.181841,46.793119,0 -71.161912,46.779431,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 04</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.179683,46.794729,0 -71.159662,46.780976,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 05</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.177526,46.796338,0 -71.157411,46.782522,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 06</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.175368,46.797947,0 -71.155160,46.784068,0
</coordinates>
</LineString>
</Placemark>
<Placemark>
<name>Line 07</name>
<styleUrl>#flightplan1</styleUrl>
<LineString>
<tessellate>1</tessellate>
<coordinates>
-71.173211,46.799556,0 -71.152910,46.785613,0
</coordinates>
</LineString>
</Placemark>
</Folder></Folder>
</Document>
</kml>
```

photoflightplan.kml (Google Earth graphic format) example:

The following example shows the xml contents of the kml file once opened in Google Earth.