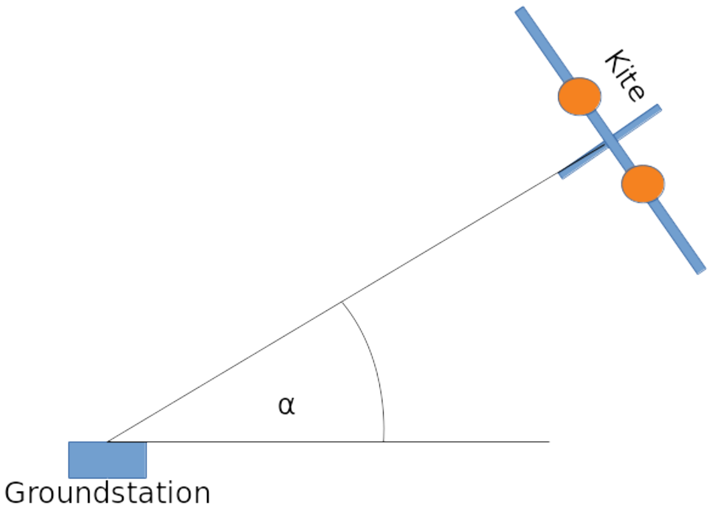
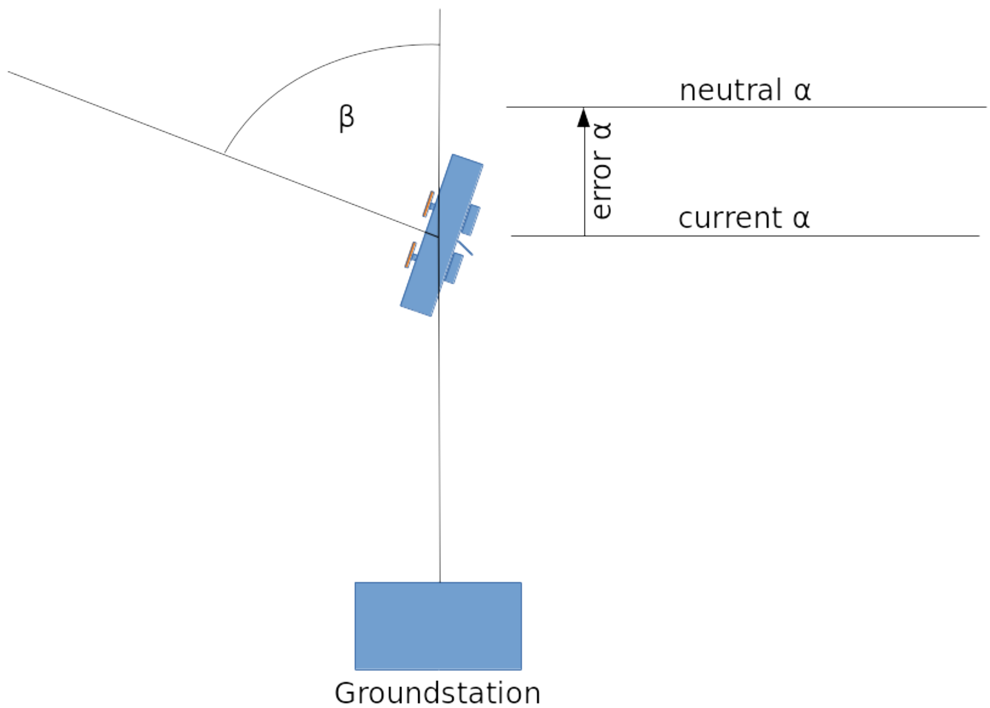
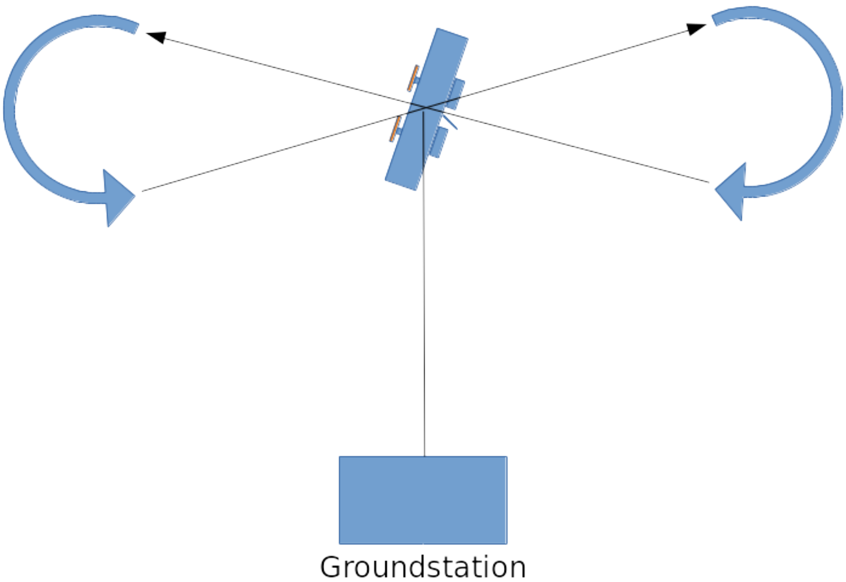
Flying figure eights autonomously



Drawing 1: view from the side



Drawing 2: view from groundstation



Drawing 3: figure eight

Sensors:
- acellerometer and gyroscope (for example MPU6050)
- atmospheric pressure sensor (for example BMP280)

Actuators:
- rudder

Coordinate systems and angles involved:

The angle β is defined in the tangent space to the sphere that is defined by the current length of the kite line with the groundstation as its center. In this tangent plane there is a direction of steepest ascent relative to the horizon. β is the angle relative to the this direction of steepest ascent. The angle β can be calculated using dot and cross product Math covered in high school.

Method:

TODO: define β properly, +- pi/2 etc.

The angle α (Drawing 1) can be measured directly using the acellerometer if the kite/aircraft is designed in a roll-stable fashion (for example using dihedral). If the aircraft is roll-stable, the tension of the kite line replaces the role of the gravitational force, thus forcing the wings to be perpendicular to the kite line.

The "figure eight" path flown by this device (much like the figures of a stunt kite) can be thought of as iterations of an almost horizontal section followed by a 180 degree turn (Drawing 3).

During the horizontal path the angle α is controlled by a P controller (PID controller without I and D terms). The output of this controller (error α, Drawing 2) serves as input to a D-controller controlling the direction β of flight). The direction β is (which should be obvious) controlled using a P-controller on the rudder.

For the turns at the end of each almost horizontal straight line flight path section the same P-controller on the rudder can be used to "hold" the intended direction β which is now slowly being changed at a predefined rate (angular velocity) until the kite is back on a horizontal track.

Simplifying everything, we do not have to distinguish between straight line section and turn of the figure eight. Both can be implemented using the exact same controller for the angle β. We then introduce a variable B, which follows β but does not change faster than the predefined maximum turning speed.

The high level algorithm would be:

```
B := 0
direction := 1;

EVERY n seconds:
        direction := -direction

β := D-gain * error-α + direction * π/2
IF(B<β)
        B += max-turning-speed
ELSE
        B -= max-turning-speed
```