Summary of the Auto-Takeoff and Auto-Land Procedure

Success! We were able to fly the aircraft through a variety of control points in a complex but organized manner. We should ask, in terms of the airplane control, what's next?

The difficulty of an aircraft mission can be defined on: The complexity of the mission accomplished (survey, scanning, telemetry), the different stages of flight (takeoff, loiter, cruise, landing), and/or the risks involved (crosswinds, visibility, amount of obstacles, and externalities such as people or other machines). This inherent difficulty interferes with the requirement of making the aircraft mission as simple as possible, which will, in terms of the benefitted user, reduce the time and cost involved on the performing of the desired mission. Therefore, the most ideal scenario is a place capable of doing any sort of mission with just the push of a huge button. This outcome, however, is not possible. Tradeoffs have to be made and personnel has to be trained in at least some way, to reduce the risks of failure to as close to zero as possible.

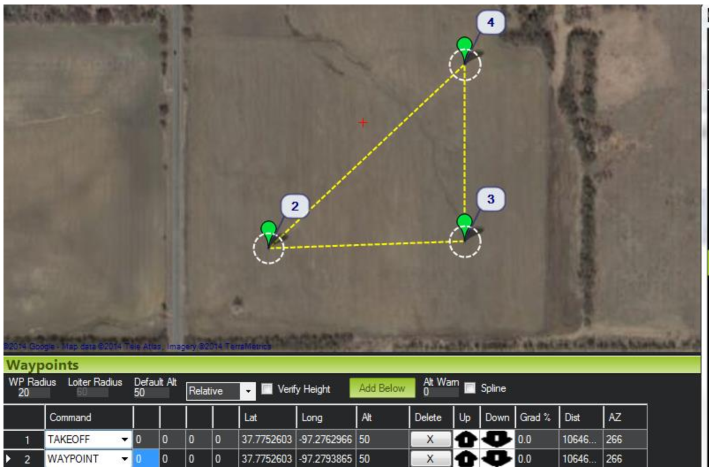
Ardupilot is a very promising platform. The amount of work done throughout the last 2 years has transformed this simple tool with high prospect into a reliable tool for aircraft mission automation, capable of tasks that before were just possible by employing costly equipment by very exclusive vendors. Thanks to its "Mission Planner" part, it can be set up to do several types of simple maneuvers and flight paths all by itself, using a set of control parameters (PID's) as the baseline for the response of the aircraft. Even with our improvised equipment, we were able to set up a system capable of performing a simple mission composed of a flight path and a return to launch command. This allowed us to gauge the capabilities of the modern mission planner, which, judging for its performance despite the lack of scanning capabilities (GPS, altimeter and accelerometers), has a very promising performance. The next step was a slightly more complex part of the program, a couple of instructions called auto-takeoff and auto-land.

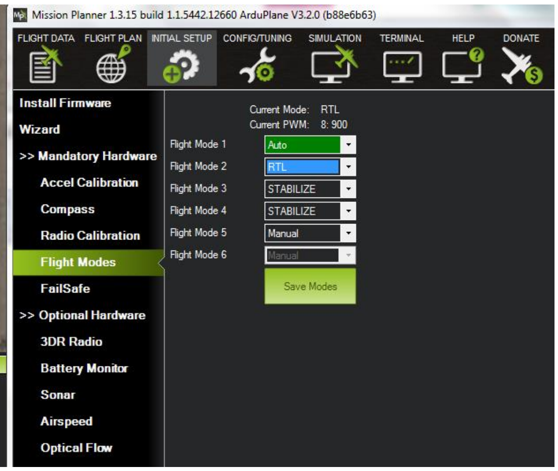
The need of these capabilities depends on the amount of simplification of the system. The more the airplane is able to do by itself, the better, as it requires less qualified personnel to control the airplane, and it also simplifies the chore of setting up the equipment, as it will only depend on how the information is methodically loaded onto the system. An airplane able to perform every stage of the mission automatically, for example, will not require an experienced pilot on the other side of the controls, meaning that the provider of the service doesn't need to be present in the field.

1. The definition of auto-takeoff

Auto takeoff is a setting present in the setup of the arduplane which throttles the aircraft to the maximum fixed throttle setting (75% in our case) and, using a certain pitch heading parameter, eventually reaches the desired mission altitude. The problem of the takeoff parameter is that it will depend on few or many variables depending on the type of takeoff desired, as the nature of the launch will depend on the type of takeoff used by the airplane.

Auto takeoff can be set up so that it works entirely on the AUTO mode on the ardupilot, or as a point that can be set up on the mission planner. It depends on the amount of desired involvement by the user. As a group we might strive for either depending on the difficulty of the mission.





a. The power of the home position

The capabilities of the autopilot were discovered when the idea of the home position was barely known. The come position depends on where you first arm the ardupilot, which means that it is not a fixed parameter; it is a parameter with variable settings of position. However, home altitude, a parameter input in the mission planner software, will specifically target the altitude at which all the other non-dependent automatic parameters (loiter, return to launch, circle) will work. Therefore, it will NOT depend on the current pressure altitude position (whether we are in Denver or the Death Valley, for example). The rest of the altitude parameters specified on each of the mission points in INDEPENDENT on the home altitude, and will just depend on the altimeter present on the ardupilot. Besides, the altimeter on the ardupilot works with the arming position as the altitude baseline, and it seems to be decently accurate, to the point where there is no needs for an additional altitude sensor to parametrize further the altitude control.

b. Speed Sensing

One of the main issues for takeoff with variable mission payloads is the stall speed. Depending on the airplane, the stall speed will vary, as it will not only depend on the aerodynamics, but also on the weight distribution of the aircraft (static margin). It is IMPOSSIBLE for the ardupilot to sense stall speed before is too late, but for a known airplane such as the Bixler 2, the default parameter would work as long as the contained payload is not too far off from the normal

position. The good thing about the mission planner and the ardupilot itself is that the program deals with the issue in two different ways: a velocity-independent more automatic procedure, and a velocity-dependent more involved procedure.
For the velocity-independent procedure, a setting on the parameters of the apm such as:

Parameter	Function
STALL_PREVENTION	Prevents stall when airspeed sensor is off
ARSP_FBW_MIN	Minimum Airspeed for level flight

Will prevent the plane from stalling, provided that there is a fair knowledge of a speed ballpark to prevent it from stalling. Also, the airplane during takeoff will try to set the throttle at the maximum FIXED throttle setting possible, which ensures that the airplane will reach the desired altitude and airspeed in a considerably short amount of time.

For a more controlled approach, a speed dependent procedure is also possible. This procedure will depend on an additional piece of equipment, a pitot tube speed sensor, which will give a more accurate measurement of airspeed, accounting for any type of wing, allowing the aircraft to take into account incoming wild as one of the defining variables. This this type of equipment, it is possible for the airplane to correctly define and estimate the desired speed of flight that combines mission efficiency with accuracy on the completion of the objectives.

Takeoff can be artificially achieved via the mission planner, by the addition of control points in which the speed is carefully managed point by point, in a way that will ensure a proper conduction of the procedure, sensitive to both winds and weather conditions (thanks to the pitot tube). The main difference between the

GPS speed estimation and the pitot tube is the difference between ground speed and true airspeed. Ground speed in necessary for the completion of the mission in a timely manner, but it ignores the fact that the wind can increase or reduce the speed sensed by the airplane itself, which will directly affect performance.

c. Rolling takeoff vs Thrown takeoff vs Others

There are several types of takeoff, and each one of them utilizes different parameters on the arduplane interface. Airframes also vary drastically on the different types of takeoff, as some of them can be more convenient than others. Hand launch, for example, obviates the requirement of a runway, but it requires a more involved, more risky physical takeoff.

Auto takeoff depends of the following cases:

Hand Launch

Bixler 2 and many of the small planes with relatively low stall speeds, like foam gliders, use this method. It is also less dependent on the runway conditions and somewhat reliable, as long as the person throwing the airplane knows what he is doing. The biggest dangers on this type of takeoff are: the engine starting with respect to the launcher's arm and the climb rate controlled also by the launcher. It is preferred that if the motor only starts as soon as the engine is past your arm to prevent injury, as well as ensuring that the aircraft doesn't start climbing to steeply, which varies according to the takeoff weight, the direction of the launch, and the trust-to-weight ratio of the airplane.

The main parameters controlling the launch are:

Parameter	Function
TKOFF_THR_MINACC	Minimum forward acceleration for the throttle to engage in THR_MAX.
TKOFF_THR_DELAY	Delay in 0.1s units to hold off starting

	the motor after the minimum acceleration is achieved
TKOFF_THR_MINSPD	Minimum groundspeed (GPS) before the motor starts.
TECS_PITCH_MAX	Minimum pitch that the autopilot will demand during takeoff procedure

The delay and min speed parameters ensure that the airplane is out of reach when the engine is spinning, so that there is no gruesome accident.

Catapult

The main difference between hand launch and catapult is the method how gets to the air. With a catapult, the aircraft gets a greater level of forward acceleration, and the risk involved due to a possible stall is reduced. The main risk involved is that the propeller might crash against the catapult frame instead of the hand.

The parameters used for catapult launching are the same parameters as the hand launch, albeit different due to the higher accelerations.

Bungee launching

This is just elastic that works as a cheap catapult. It gives good results for small and medium sized models. The risks and parameters compared to both the catapult and the hand launch are the same. We just need to keep in mind that the parameters themselves are going to be generally in between those form the catapult and those from the hand, as the acceleration is in between.

Runway Takeoff

For much heavier models full of equipment, as well as a much broader range of payload customization, is imperative that runway takeoff can be achieved using ardupilot. The problem is based on the fact that more parameters and more tuning is required, as the takeoff speed will vary on the roughness of the track, the weight of the aircraft, and the size of the wing. Most of these parameters have to be set up by hand. Therefore, I suggest sticking to hand launch as much as we can, provided we have engines with high thrust to weight ratio. For more endurance dependent missions, due to the sheer size of the wing and the low T/W, this takeoff is necessary. This type of takeoff is set up using the Mission planner as one of the segments. It will depend on whether the configuration is either tricycle or tail dragger, as there are specific parameters for each of these configurations. It is desired, however, to use the tricycle configuration for perfect takeoff handling.

The following parameters are important:

Parameter	Function
TKOFF_TDRAG_ELEV	Used to hold the tail of a tail dragger hard on the runway during the initial stages of takeoff, to give it enough grip on the runway to steer. For a tail dragger this is normally set to 100, meaning that 100% up elevator is applied during the initial stages of takeoff.
TKOFF_TDRAG_SPD1	When the takeoff starts, the autopilot will apply TKOFF_TDRAG_ELEV elevator (as a percentage) until the aircraft reaches a speed of TKOFF_TDRAG_SPD1 meters per second. You need to set TKOFF_TDRAG_SPD1 to a speed below the takeoff speed, but above the speed where the aircraft is able to steer using its rudder.
TKOFF_THR_SLEW	Controls the throttle slew rate (as a percentage per second) during takeoff. This is used to allow the throttle to ramp up at a rate appropriate for your aircraft.
TKOFF_ROTATE_SPD	Controls when the autopilot will try to raise the nose (pitch up) to leave the ground.
TECS_PITCH_MAX	Controls the maximum pitch used when climbing on takeoff.
GROUND_STEER_ALT	Controls the altitude switching from the ground (nose wheel) to the air (rudder) by specifying the cutoff altitude.

Once all these parameters are configured, takeoff can be calibrated successfully. The extent of the ground takeoff will depend at the end, however, on the setup of the mission points on Mission Planner.

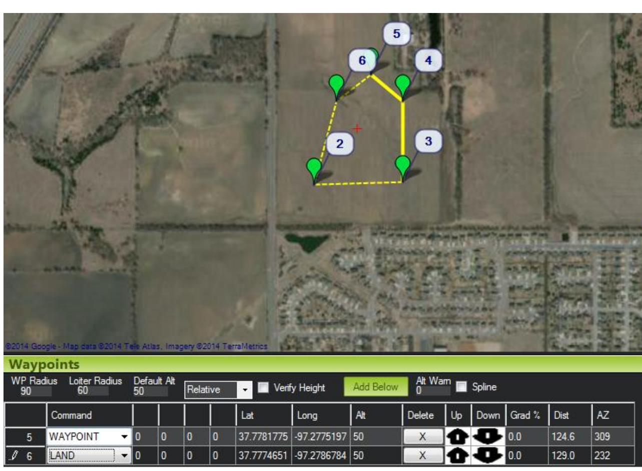
d. Limitations of Auto takeoff

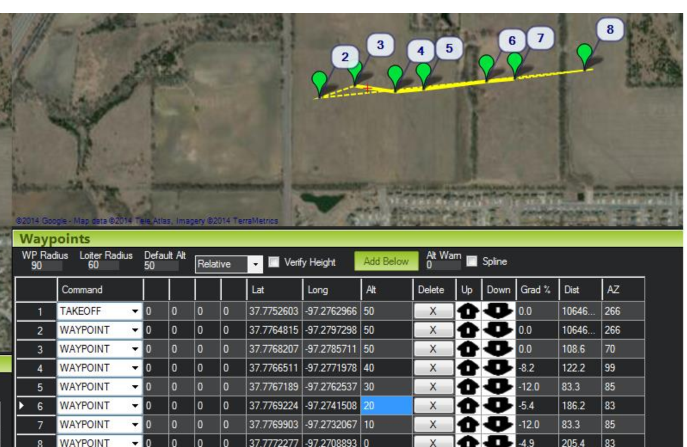
Auto takeoff, despite all the complexity and all the completeness imputed by Mission Planner, is not perfect. Crosswind takeoffs are only a problem if the runway direction is not correctly set up, as the aircraft is needed to takeoff exclusively against the wind, which will depend on the user, even more so for hand launch. For exclusively crosswind takeoffs (in case of a runway with winds perpendicular to the runway), the stability of takeoff will depend on the configuration of the control system, as well as the amount of wind (conditions up to 2σ in windy cities such as Wichita).

Other unknown conditions present during takeoff depend on the coarseness of the parameters, which were defined with a balance of simplicity (being able to cover several models) and usability (as many parameters as possible to ensure correct takeoff behavior).
These limitations, however, do not downplay the usefulness of the automatic takeoff capability of the ardupilot, which fulfils all the requirements needed by us regarding the types of missions we use it for.

2. The definition of auto-land

Auto-land is a function that allows ardupilot to perform landings with little or no aid from the user of the airplane. This capability has evolved steadily since the beginning of the platform a couple of years ago, and it is becoming more reliable as time passes. Auto land can be done in two different ways: using the landing option present on the mission planner along with the different parameters associated with it (see below), or manually through the proper set up of mission waypoints at different altitudes.





a. Inherent Complexities and Key Parameters

Both automatic landing methods that exist in the platform require a proper setup of different aircraft parameters, as well as awareness of the conditions and proper calibration of the sensing equipment. Just like full-size airplane autoland, the altimeter plays a critical role during the landing. Crosswinds or any other type of wind also affects the way the aircraft performs during a particular landing, independently whether the aircraft lands in wheels or not.

The key parameters that control automatic landing are:

Parameter	Function
NAV_LAND	Command set up at the end of the mission planner waypoint list. You have to indicate the location (lat, long) and altitude (often 0) of the desired touchdown point.
LAND_FLARE_ALT	Desired altitude at which the airplane is desired to flare, independent of its descent rate.
LAND_FLARE_SEC	Time in seconds before the aircraft would hit the ground if it continued with its current descent rate.
LAND_PICHT_CD	Minimum pitch of the aircraft once the flare has happened. This parameter controls the flare segment.
TECS_LAND_ARSPD	Parameter, in m/s, which controls the landing airspeed at which the aircraft will land. It should stay above stall.
TECS_LAND_SPDWGT	0-2. It controls whether priority is set at maintaining either airspeed (close to 0) or control (close to 2).

The parameters above are very useful in the added auto land capability. If more accuracy is required, however, waypoints can be set up so that the airplane follows a controlled landing waypoint after waypoint, in which the user could set up every single detail of the landing (landing distance, speed, attitude, and time).

b. The need of a more accurate speed sensor

Just like during takeoff, landing needs to take into account headwind and crosswinds as well, because precise management of the airspeed is needed at speeds close to stall. The speed sensor used for automatic takeoff is enough for the landing as well. The system will just make the speed readings more accurate, as the GPS will give the groundspeed, while the Pitot tube will give the true airspeed.

In terms of altitude sensing, the barometer provided with the APM platform is good enough for most part, as long as the ardupilot is set up at the baseline altitude at which will be used to land the plane. If the base altitude is such that is much higher than the landing ground, the airplane might think that it landed, when that might not be the case. An accident also would happen if the ground used as home position is much lower than the landing grounds, as the airplane might crash thinking that it hasn't reached the required altitude to flare.

c. Rough Landing or Controlled Crash?

When Julio and David did the test with the Bixler 2, the fact that hand launched airplanes don't have landing gear makes them land more like a controlled crash than a landing. This hand launched airplanes, however, tend to have the advantage of the weight saving resulting from the lack of landing gear, meaning that more weight can be used for the payload, which varies in size according to the mission.
The good thing about the arduplane is that it can record telemetry data when crashes happen. They are called dataflash logs, and they are automatically included on the 16mb on-board memory on the ardupilot. The can be configured specifically for the essential parameters, and the function starts automatically when the system is turned on. This telemetry system can record some of the accelerometer data and deflections, but it is severely limited on its discretization due to the limited space on the equipment, as well as the increased CPU requirement of a higher data rate. This data rate can be set up using the mission planner 'telemetry' option under the 'software' tab.

d. Limitations of Auto-Land

Auto-land, just like the automatic takeoff, suffers from similar limitations, many due to parameter limitation (there is a limited space and therefore a limited number of parameters than can be specialized), resource limitation (limited amount of memory and CPU power to perform much more complex landing maneuvers (high risk crosswind landings, landing in a very thin strip of land, etc.), and sensing limitation (the accuracy and number of sensors employed in the landing maneuver).

These limitations however don't render the automatic landing unfit for the tasks that we need it for. Since we are going to work with mostly desirable conditions and relatively open spaces for most of the time, there is room for the autoland to perform its function. The exact point for landing doesn't need to be achieved as long as it is within the reach of the operator.