Software Version Control Standard Operating Procedures for Miniature Unmanned Air Systems to ensure Safety of Flight

Advanced Robotics and Systems Engineering Laboratory
Naval Postgraduate School

<table>
<thead>
<tr>
<th>Version</th>
<th>Rel. date</th>
<th>Changes/Additions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>7/25/2013</td>
<td>Revisions for 2013-2014 IFC.</td>
</tr>
<tr>
<td>2.0</td>
<td>6/13/2014</td>
<td>Revisions for 2014-2015 IFC.</td>
</tr>
</tbody>
</table>

Target Systems

This document applies to the following software systems:

- ArduPlane and ArduCopter (also known as APM:Plane and APM:Copter) autopilot software
- ARSENAL lab uses the following hardware systems with those autopilot software systems:
  - APM, [http://store.3drobotics.com/products/apm-2-6-kit-1](http://store.3drobotics.com/products/apm-2-6-kit-1)
  - Pixhawk [http://store.3drobotics.com/products/3dr-pixhawk](http://store.3drobotics.com/products/3dr-pixhawk)
  - PX4, [http://store.3drobotics.com/products/px4fmu-1](http://store.3drobotics.com/products/px4fmu-1) and [https://pixhawk.ethz.ch/px4/modules/px4fmu](https://pixhawk.ethz.ch/px4/modules/px4fmu)
- Ground Control Station software that works with those systems:
  - Mission Planner
  - MAVProxy
  - QGroundControl
  - NPS Aerial Combat Swarms (still under development)
    - While under development, this software can only be run with either MAVProxy, QGroundControl, or Mission Planner running as a backup station for any plane under control an Aerial Combat Swarms GCS.

Purpose

The NPS ARSENAL lab is requesting an IFC that allows for:

- New releases to autopilot software coming in regularly from the Open Source community, and updates to software made by NPS as well. Being tied to a specific autopilot or ground control station version for an entire year will hamper research.
• Patches made by NPS researchers to allow for:
  o Improved safety constraints on software (the hobby market tends to be less concerned with safety and containment than NPS).
  o Modifications to autopilot control algorithms (for example, the PID controller, L1 controller, and navigation system) to allow individual planes to act as members of a swarm.
  o Modifications to ground control software to ensure operators can control or abort individual planes, swarms of planes, or all planes.

• Other parts of the autopilot will likely not be modified (e.g., we do not plan to modify the inertial navigation sensors, software associated with sensing, the process scheduler, etc.). However, the nature of research is that we may encounter an unforeseen situation requiring other software or hardware modifications sometime before the IFC expires. We need an expedited method to amend the IFC should such a case arise.

Software Configuration Management

In an environment where software development is ongoing it is essential that experimental software branches be separated from vetted software in order to ensure that no plane ever flies with untested software. NPS utilizes the git revision control system to maintain software that is under development separate from stable release versions. NPS follows the recommended procedures outlined by git developers by:

• Maintaining the main development branch under the section of the repository labeled “development.” The git revision system allows for changes to be tracked per developer, annotated, and rolled back.

• Maintaining experimental branches separate from main development branches. Each branch is labeled based on what is being experimented on separate from the main branch, e.g., “Boids_Flocking_Branch” or “Barrel_Roll_Attach_Branch.” Beyond this additional labeling, we use git checksums to keep track of individual commits. As in the development branch, changes can tracked and rolled back. For these branches, git allows us to merge experimental code into the main repository when deemed appropriate.

• Maintaining software that has been tested and considered stable as special branches known as “tags.” Each release is labeled with a date as follows: May-14-2013. When a new version is released it is not considered stable until full lab and field tests (described in the subsequent section) have been completed. New version will therefor maintain a release candidate label (e.g., rc_May-2013) until fully vetted.

• Since we are small shop, the only release that is under active maintenance is the current release intended for experimental use. When bugs are identified in an existing release they are tracked via the Redmine bug tracker. Before the lead developer will accept a bug fix as closed the fix must pass all unit and regression tests and then be merged into (1) the current development branch, (2) the current release candidate release (as necessary). Since software is currently
under active development, in practice this requires that lab and field testing be re-done prior to EVERY experimental event (occurs about once every 2 months).

Lab and Field Test Procedures for a New Software Release Candidate

NOTE: This testing is IN ADDITION to unit testing performed on software during the development process.

NPS recognizes that even with software that is in development, the autopilot must provide containment and positive aircraft control. Before a software release candidate can be used in an operational context, the following fail safes have been verified to be in place through the current version of our autopilot. The following tests are performed in the lab in a simulator that encompasses the autopilot, a flight simulator, and a ground control station connected to a payload computer running NPS software that communicates with the ground control station. Currently we are using the autopilots and GCS tools indicated at the beginning of this document.

Furthermore, each new release candidate must also perform an operational test flight in the field where these failsafes are again tested. As we are small lab, the software review board consists of: (1) the lead developer running tests in simulation, and (2) a pilot verifying the tests during a test flight. Only after both have agreed that the software meets these minimum baselines is a release candidate considered stable.

With the exception of the manual abort, all systems reside on board the autopilot and are not dependent upon communications with the ground:

- Geofence which maintains plane within a three dimensional box that is a subset of the restricted airspace. Breaching the geofence results in the plane performing an RTB.
- Command communications monitor. Any plane losing contact with the ground control station will RTB.
- Battery health monitor. Any plane whose battery life reaches a lower bound threshold will RTB.
- Mission Abort. Ground control station allows for any plane aloft to be commanded to perform either an RTB (controlled autonomous landing) an immediate landing (kill throttle and set surfaces for circling), or an emergency throttle kill.

The following fail safes have been added by the NPS ARSENAL lab to augment the autopilot software:

- GPS fail safe improvement. Any plane loses GPS will not attempt to fly home using the IMU (vendor’s behavior on some autopilots), rather it will perform an immediate landing by killing the throttle and setting control surfaces to cause the plane to circle.
- Fail safes recognize that when the plane is in Manual mode, that pilot control should not be overridden. For example, a plane under manual control that has lost GPS should allow an operator to manually pilot it home, rather than attempting to kill the motor. If it is possible, a manual pilot with eyes on the plane is usually the safest way to bring in a malfunctioning plane.
Software releases **must not** modify the above fail safes; every plane in the air must have these baseline safeties in place. Verification procedures taken for previous and current versions of autopilot (these procedures must also be taken for future versions):

- Verification that failsafes are in place via software checks and bench tests. Failsafes that can be checked in this manner are:
  - Communications loss is bench tested in the lab by disconnecting the GCS antenna. Autopilot should then enter fail safe mode.
  - Battery health is bench tested by modifying the low battery threshold voltage value and ensuring that autopilot enters fails safe mode.
  - Loss of GPS is bench tested by disconnecting the GPS antenna. Autopilot should then enter fail safe mode.
  - Permanent Manual override is tested on the bench by setting the autopilot to Manual mode and then running through all the previous bench tests, ensuring that the plane does NOT enter fail safe mode.

- Test flights of a single plane followed by small groups of planes before introducing modified software to large swarms. Objective of these flights is to verify autopilot control software and all fail safes are still functional.
  - Currently, geofence failsafe must be verified by a test flight (both simulated and operational) at this time (cannot be tested on the bench).

When a new behavior is introduced to the autopilot or ground control station by NPS developers:

- The ability to fall back to the previous “known good” version must be present and accessible.
- Only one new behavior at a time will be tested. Before the new behavior is tested, a series of tests will be performed to ensure the modification has not unexpectedly altered a plane’s performance.
- The package of unit and regression tests that ensures that the plane continues to function within limits includes:
  - Roll, pitch, and yaw limits are obeyed.
  - Heading is held during straight line flight.
  - Racetrack flight does not result in S turns.
  - Waypoints are obeyed within a 75m tolerance in racetrack flight (overly wide turns are a sign of degradation in positive control).
  - Commanded altitude is maintained during straight, waypoint, and loiter flight.
  - Throttle does not oscillate outside of known tolerances during straight flight.