



# Using Git for OpenVSP Model Version Control

2020 OpenVSP Workshop, Virtual Broadcast

September 15-17, 2020

BRANDON LITHERLAND, AST  
NASA LANGLEY RESEARCH CENTER  
AERONAUTICS SYSTEMS ANALYSIS BRANCH



# Outline

- Why use Git?
- Nomenclature
- Git-Bash
  - Basic commands
- GitHub & GitLab
- GitHub Desktop
- Demo (time permitting)

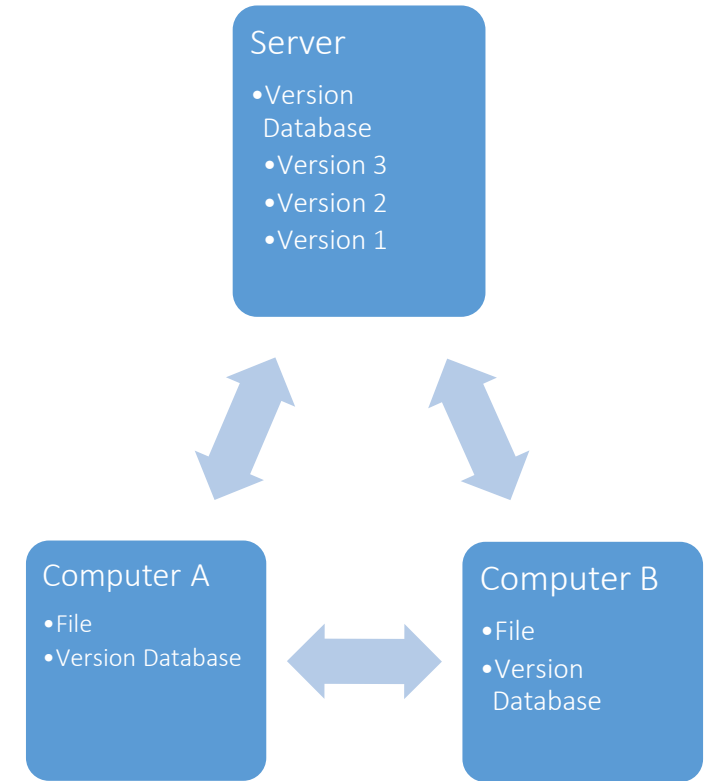


- We are discussing version control for models rather than for software source code.
  - This can include geometry, performance models, design inputs, notes, scripts, etc...
- Applies to single/multiple designers working on a model.
  - Each individual can modify the models as needed at the same time and merge changes when complete.
  - Enables extensive collaboration.
- Avoids saving multiple copies of the same model as the design progresses (“backups”).

# Why Use Git?



- Simple, clean, easy to use version control of various file types.
- You can save files as much as you want without affecting the stored version.
  - Your changes do not affect the controlled model until you “commit” the changes.
- You can revert to any previous version at any time without losing the most recent work.
- Highly collaborative environment on user-controlled resource.
- Can work locally on your laptop or online at GitHub or GitLab. Changes are tracked on both.
- You aren't just working with a local copy, you work with the entire history of the project.
  - Distributed Version Control System (DVCS)
  - There are multiple backups of the project in case something is corrupted or lost.
- OpenVSP source code is on GitHub!
  - <https://github.com/OpenVSP/OpenVSP>





# Why Use Git?

- VSP models are XML-based code
- Few VSP files are unformatted.
- Traditionally used for software.
  - Works well with human-readable formats.
- You can version control *everything* in the repository:
  - VSP scripts
  - Python code/scripts
  - Design files
  - Variables
  - Configurations
  - Analysis input & output files
  - PDF, images, Excel, etc.
    - Unformatted or binary files are still tracked but you cannot see the differences. Reverting can be difficult.
- Documentation uses Markdown language. Useful for keeping a summary or description.

```

1  <?xml version="1.0"?>
2  <Vsp_Geometry>
3    <Version>4</Version>
4    <Vehicle>
5      <ParmContainer>
6        <ID>YGWDMTNJCK</ID>
7        <Name>Cessna_337_Skymaster</Name>
8        <BBox>
9          <X_Len Value="2.953802208208907100e+001" ID="RLRZFCBKUNB"/>
10         <X_Min Value="-1.300000000000000000e+000" ID="JBVJZKSHDTA"/>
11         <Y_Len Value="3.795936915246787900e+001" ID="WMFTWSAJFBP"/>
12         <Y_Min Value="-1.897968457623393900e+001" ID="DVBDDNNYCSSL"/>
13         <Z_Len Value="8.228481480423333500e+000" ID="NESDHAOCLBY"/>
14         <Z_Min Value="-3.110381480423333400e+000" ID="PDGATYWMJND"/>
15       </BBox>
16       <Design>
17         <Working_XDDM_Type Value="0.000000000000000000e+000" ID="LNBWCPTZDRD"/>
18       </Design>
19       <ExportFlag>
20         <CompGeom_CSV_Export Value="1.000000000000000000e+000" ID="ZMUKGNUYOFS"/>
21         <DegenGeom_CSV_Export Value="1.000000000000000000e+000" ID="JPICRTUYSIX"/>
22         <DegenGeom_M_Export Value="1.000000000000000000e+000" ID="GVYMNMTADEII"/>
23         <DragBuild_TSV_Export Value="1.000000000000000000e+000" ID="JCVTEMRKMPJU"/>
24       </ExportFlag>
25       <FitModel>
26         <Select_Box_Flag Value="0.000000000000000000e+000" ID="BIPTTOINLWB"/>
27         <Select_One_Flag Value="0.000000000000000000e+000" ID="AVNHDNTVURD"/>
28         <U_TargetPt Value="0.000000000000000000e+000" ID="JNQDQHDFFQYA"/>
29         <U_Type Value="1.000000000000000000e+000" ID="GFGLKORGTXI"/>
30         <W_TargetPt Value="0.000000000000000000e+000" ID="IHZPNQMLERK"/>
31         <W_Type Value="1.000000000000000000e+000" ID="UTKACGWTQQ"/>
32       </FitModel>

```

```

1  # DEGENERATE GEOMETRY CSV FILE
2
3  # NUMBER OF COMPONENTS
4  2
5
6  # DegenGeom Type, Name, SurfNdx, GeomID
7  LIFTING_SURFACE,WingGeom,0,JFCMTJETXJ
8  # DegenGeom Type,nXsecs, nPnts/Xsec
9  SURFACE_NODE,49,21
10 # x,y,z,u,w
11 1.000000000000000000e+00, 0.000000000000000000e+00,
12 9.838635136053281993e-01, 0.000000000000000000e+00,
13 9.462389668831039380e-01, 0.000000000000000000e+00,
14 8.645616039409491638e-01, 0.000000000000000000e+00,
15 7.118569384843369541e-01, 0.000000000000000000e+00,
16 4.924601613712297210e-01, 0.000000000000000000e+00,
17 2.724626653299470624e-01, 0.000000000000000000e+00,
18 1.192719059093479000e-01, 0.000000000000000000e+00,
19 3.904834319157425571e-02, 0.000000000000000000e+00,
20 6.189458957445818921e-03, 0.000000000000000000e+00,

```

```

Sref = 45.000000
Cref = 2.500000
Bref = 18.000000
X_cg = 0.000000
Y_cg = 0.000000
Z_cg = 0.000000
Mach = 0.233300
AoA = 6.000000
Beta = 0.000000
Vinf = 100.000000
Rho = 0.002377
ReCref = 10000000.000000
ClMax = -1.000000
MaxTurningAngle = -1.000000
Symmetry = Y
FarDist = -1.000000
NumWakeNodes = -1
WakeIters = 5
NumberOfRotors = 0
NumberOfControlGroups = 0
Preconditioner = Matrix
Vortex Lift = Y
LE Suction = N
Karman-Tsien Correction = Y

```

# Git Nomenclature

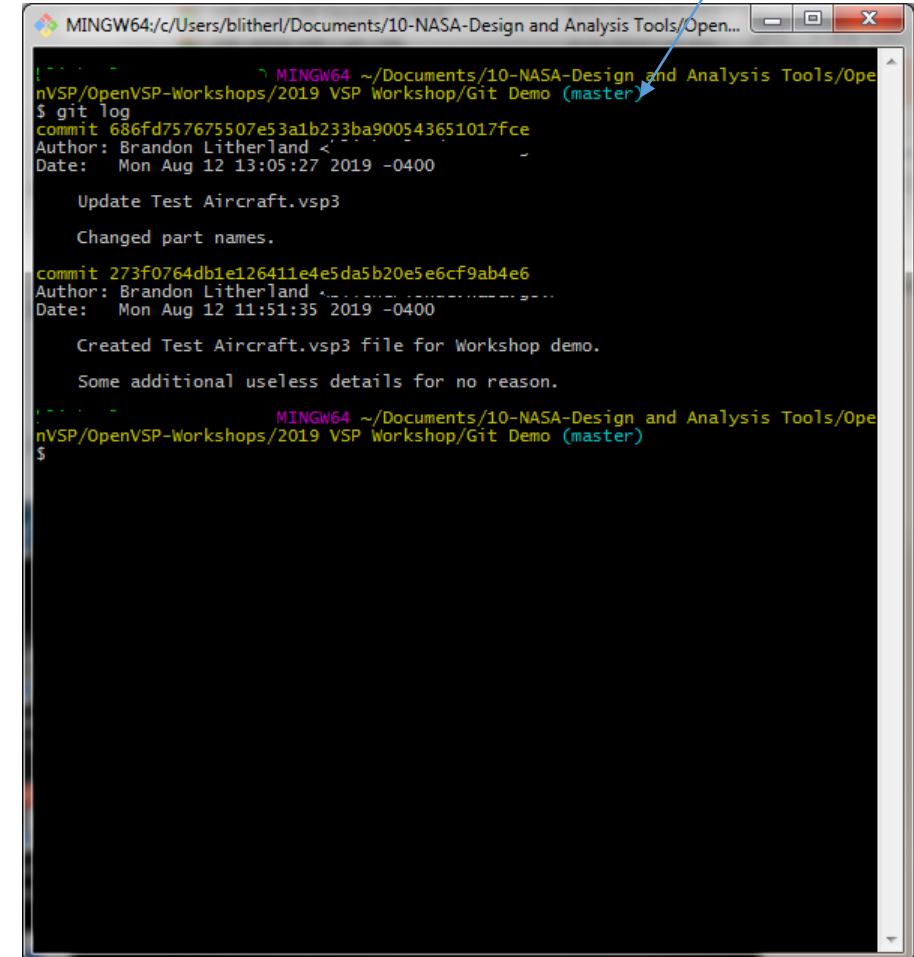


- **Branches** – Parallel versions of the project where you can safely make changes, experiment, fix bugs, etc...
  - “I’ve checked out a version of the project to work on.”
- **Fork** – GitHub (online only) term that implies making an online copy of the main repository of work to your user account.
  - “I’m working on MY copy of the project.”
- **Clone** – Copy the online repository to my local storage.
- **Checkout** – Work with a specific project branch.
- **Commit** – You update the controlled version of the project with the saved copies stored locally.
  - “I’m happy with my saved changes and want those changes added to the project and history.”
- **Push/Pull** – Either “push” changes from your local copy to the online repository or “pull” the information from one branch to another.
  - “I’m done making commits to the project for now and am ready to update the main branch. I push my local changes to the online repository”
  - “I request to pull the changes from my online branch to the main branch.” – When collaborating.

## Git-bash - For those that enjoy command line interfaces: <https://git-scm.com/download/>

- Linux-style command structure for controlling local repositories and making changes online.
- Easy to learn and usually quicker than using a GUI.
- Current working branch is highlighted in blue.
- Several operations in Git can not be done online at GitHub. Local Git installation must be used.
- There is also a Git GUI but not very intuitive/useful.
- Git Cheat Sheet: <https://services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf>

Current Branch



```
MINGW64~/Documents/10-NASA-Design and Analysis Tools/Open...
MINGW64 ~/Documents/10-NASA-Design and Analysis Tools/Open...
nVSP/OpenVSP-Workshops/2019 VSP Workshop/Git Demo (master)
$ git log
commit 686fd757675507e53a1b233ba900543651017fce
Author: Brandon Litherland <...>
Date: Mon Aug 12 13:05:27 2019 -0400

    Update Test Aircraft.vsp3

    Changed part names.

commit 273f0764db1e126411e4e5da5b20e5e6cf9ab4e6
Author: Brandon Litherland <...>
Date: Mon Aug 12 11:51:35 2019 -0400

    Created Test Aircraft.vsp3 file for Workshop demo.

    Some additional useless details for no reason.

MINGW64 ~/Documents/10-NASA-Design and Analysis Tools/Open...
nVSP/OpenVSP-Workshops/2019 VSP Workshop/Git Demo (master)
$
```



## Useful Commands

**\$ git init [project-name]**

- Creates a new local repository with the specified name.
- “git init” makes the current directory the repository.
- Makes the repository in the current directory.

**\$ git clone [url]**

- Downloads a project and its entire version history.
- Makes the repository in the current directory.

=====

**\$ git branch**

- Lists all local branches in the current repository.

**\$ git branch [branch-name]**

- Creates a new branch.

**\$ git checkout [branch-name]**

- Switches to the specified branch and updates working directory.

**\$ git merge [branch-name]**

- Combines the specified branch’s history into the current branch.

**\$ git branch -d [branch-name]**

- Deletes the specified branch.
- Good for removing branches that are no longer needed.

**\$ git status**

- Lists all new or modified files to be committed.

**\$ git diff**

- Shows file differences not yet staged.

**\$ git add [file]**

- Snapshots the file in preparation for versioning.

**\$ git diff --staged**

- Shows file differences between staging and the last file version.

**\$ git reset [file]**

- Unstages the file, but preserves its contents.

**\$ git commit -m “[descriptive message]”**

- Records file snapshots permanently in version history.

**\$ git pull**

- Fetches the online repository to your local copy.

**\$ git push [remote] [branch]**

- Pushes local changes to the remote repository
- Ex: “git push origin master”

**\$ git log**

- Shows commit history



## Example Workflow: Local Files

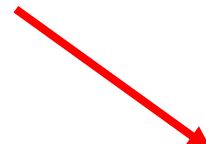
1. Navigate to desired directory.
2. `$ git init MyProject`
3. Add files to the repository.
  1. Create files using whatever software.
  2. Can have one or many files.
  3. Git is tracking changes as you save.
4. `$ git status`
  1. Uncommitted files are listed in **RED**.
5. `$ git add [filename]`
  1. Stages files for commit.
  2. Can add multiple files at once.
    1. Example: `$ git add *.vsp3` will add ALL VSP files at once.
    2. `$ git add .` will add ALL untracked or unstaged files!
  3. `$ git status` will now show staged files in **GREEN**.
6. `$ git commit -m "[comments]"`
  1. Changes are committed to the project and added to the history with [comments] describing the changes.

\*Note: `$ git commit` will bring up the VIM editor where you can make a summary AND detailed description. The "Insert" key will let you edit the text. "Esc" will take you out of insert mode. To write the text and exit the editor, use `:"wq"` – Read as colon write quit. THIS IS MUCH BETTER PRACTICE FOR DESCRIBING COMMITS!!!

```
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)


        modified:   Test Aircraft.vsp3

no changes added to commit (use "git add" and/or "git commit -a")
```



```
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   Test Aircraft.vsp3
```



```
$ git log
commit 17cbd9f8505a14e24533a2f27303b3b5c77279c8
Author: Brandon Litherland <...>
Date:   Wed Aug 21 14:01:29 2019 -0400

    First line is the commit summary.  Example: This is a commit message.

    Subsequent lines are part of the detailed description.  This can be
    arranged in any way that makes sense in text.  For example:

    Commit Feature 1 - This commit is just a test.
    Commit Feature 2 - More commit details.
    Commit Feature 3 - Ni

    - Example bulleted commit detail.

    =====
    Separate visually for easy reading.

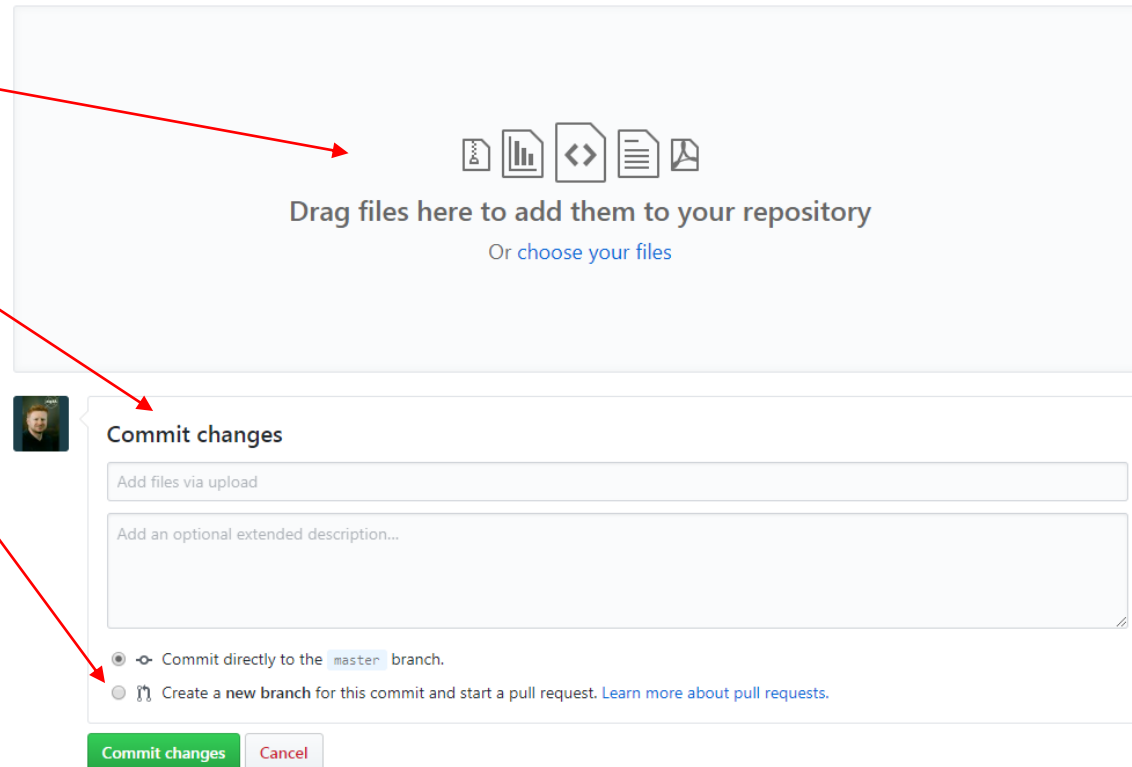
    Remeber to Save!  ":"wq"
```



- This is where the online version control and collaboration happens (mostly).
- Similar to other systems, a multitude of helpful online tutorials are present and available.
  - Example: <https://guides.github.com/activities/hello-world/>
- Step-by-step instructions on navigating Git can be found, including operations in git-bash.
- Most operations here are simple point-and-click steps.
- Very helpful in examining the status of the project and seeing what versions are checked out.

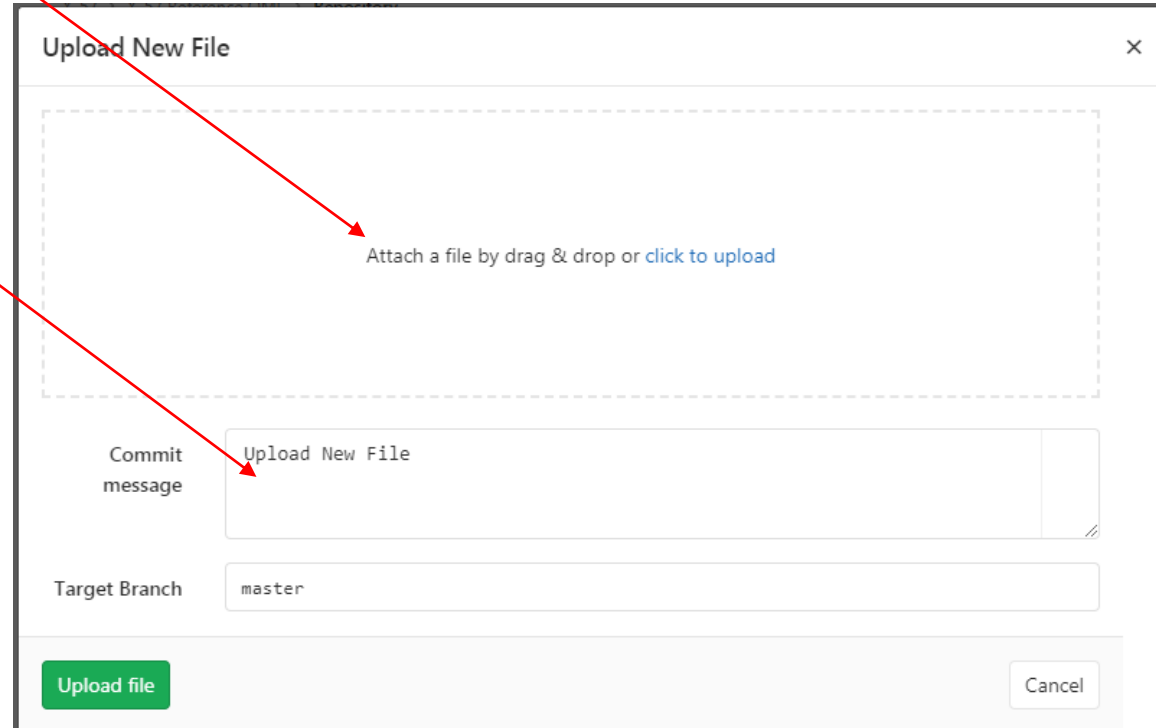
## GitHub ([github.com](https://github.com)): Online collaboration and backup version control

- Can add saved files
  - New versions of existing files or add new files.
- Perform Commits
- Initiate Pull Requests
  
- *All without using command line.*
- *Some GitHub Enterprise systems have increased in price but may now hold SBU/ITAR code.*
  - *Your experience may differ*



## GitLab ([gitlab.com](https://gitlab.com)): Online collaboration and backup version control

- Can add saved files
  - New versions of existing files or add new files.
- Perform Commits
- Track Project To-Do Lists
- Manage Merge Requests
- Similar to GitHub



Upload New File

Attach a file by drag & drop or [click to upload](#)

Commit message: Upload New File

Target Branch: master

Upload file Cancel

# GitHub Desktop



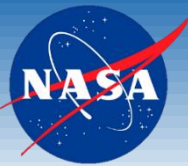
## Making your life easier...

- Most of the git-bash command line operations in a handy GUI!
- Quick navigation of repository.
- Automatic visualization of version differences.
- Found here:

<https://desktop.github.com/>

The screenshot shows the GitHub Desktop application interface. The top menu bar includes File, Edit, View, Repository, Branch, and Help. The current repository is 'Git Demo' and the current branch is 'master'. The 'Publish repository' button is visible. The 'History' tab is selected, showing a list of commits. The most recent commit is 'Update Test Aircraft.vsp3' by Brandon Litherland, committed just now. Below the commit list, the diff view for 'Test Aircraft.vsp3' is shown. The diff highlights changes in the 'RotationX', 'RotationY', and 'RotationZ' values. The 'RotationX' value has changed from 8.391749365233920344e+01 to 6.588286409235161045e+01. The 'RotationY' value has changed from 7.172886398410054198e+01 to -4.056907995273338940e+01. The 'RotationZ' value has changed from 8.414457012148544379e+01 to -2.903649336936903680e+01. The diff also shows changes in the 'ViewPortX', 'ViewPortY', and 'Zoom' values.

# Git Desktop Example



Current repository: Git Demo | Current branch: master | Publish repository: Publish this repository to GitHub

Changes 1 | History | Test Aircraft.vsp3

1 changed file

- Test Aircraft.vsp3

List of changed files

Commit Summary: Update Test Aircraft.vsp3

Commit Details: Changed part names.

Commit to master

Committed 6 minutes ago | Created Test Aircraft.vsp3 file for W... | Undo

```
@@ -11,9 +11,9 @@
11      <CORZ Value="0.0000000000000000e+00" ID="NOKXVJUBJEL"/>
12      <PanX Value="0.0000000000000000e+00" ID="YRVZBFITVJ"/>
13      <PanY Value="0.0000000000000000e+00" ID="JQOPAXHJOEU"/>
14      - <RotationX Value="8.391749365233920344e+01" ID="SAHTS...
15      - <RotationY Value="7.172886398410054198e+01" ID="ZNOJ...
16      - <RotationZ Value="8.414457012148544379e+01" ID="QWJCS...
14      + <RotationX Value="6.588286409235161045e+01" ID="SAHTSTGMAUJ"/>
15      + <RotationY Value="-4.056907995273338940e+01" ID="ZNOJ...
16      + <RotationZ Value="-2.903649336936903680e+01" ID="QWJCS...
17      <ViewportX Value="0.0000000000000000e+00" ID="MCOQKNM...
18      <ViewportY Value="0.0000000000000000e+00" ID="KALEVPF...
19      <Zoom Value="3.201108798384666443e-02" ID="RYHADKDKAUB"/>
@@ -544,7 +544,7 @@ void Scale(double curr_scale )
544      <Geom>
545      <ParmContainer>
546      <ID>VIXZPQJGFO</ID>
547      - <Name>WingGeom</Name>
547      + <Name>Wing</Name>
548      <Attach>
549      <Rots_Attach_Flag Value="0.0000000000000000e+00" ID="JIZVIPSJQLS"/>
550      <Trans_Attach_Flag Value="0.0000000000000000e+00" ID="HNYFYDAQUFU"/>
@@ -898,7 +898,7 @@ void Scale(double curr_scale )
898      <Geom>
899      <ParmContainer>
900      <ID>NHVYQDVLU</ID>
901      - <Name>WingGeom</Name>
901      + <Name>Vert</Name>
```

I changed the view rotation...

and changed part names.

# Git Desktop Example



The screenshot shows the Git Desktop application interface. At the top, the menu bar includes File, Edit, View, Repository, Branch, and Help. Below the menu, the current repository is 'Git Demo' and the current branch is 'master'. A 'Publish repository' button is visible. The main area is divided into three panes: 'Changes', 'History', and 'Commit Summary'. The 'History' pane shows a list of commits, with the most recent one highlighted: 'Update Test Aircraft.vsp3' by Brandon Litherland, committed just now. The 'Commit Summary' pane shows the commit message 'Update Test Aircraft.vsp3' and the commit hash '686fd75'. The 'Commit Details' pane shows the diff for the file 'Test Aircraft.vsp3', highlighting changes to rotation values. A terminal window in the bottom left shows the output of the 'git log' command, with the full commit hash (SHA) '686fd757675507e53a1b233ba900543651017fce' highlighted. Callouts point to various elements: 'Commit Log' points to the commit history, 'Commit Summary' points to the commit message, 'Commit Hash ID (partial)' points to the commit hash in the commit summary, 'Commit Details' points to the diff view, and 'Full Commit Hash (SHA)' points to the full hash in the terminal.

# Demo



1. Example Repository
2. Open Model
3. Change and Save
4. View changes in Git
5. Stage > Commit
6. View History





# Thank you!

Questions?

---

BRANDON LITHERLAND, AST  
NASA LANGLEY RESEARCH CENTER  
AERONAUTICS SYSTEMS ANALYSIS BRANCH

---