LIÈGE université
GIGA

Code versioning & Git

GIGA Doctorate School

Christophe Phillips, Ir Ph.D.

Program

- Why “Version Control” ?
- Basics of Version Control (VC)
- Git as a VC solution
- Being Git practical with GitHub/GitLab
- Conclusions & reference

What is “Version Control”

*In software engineering, version control (aka. revision control, source control, or source code management) is a class of systems responsible for **managing changes** to computer programs, documents, large web sites, or other collections of information.*

i.e. organize and control **revisions**.

https://en.wikipedia.org/wiki/Version_control

Program

- Why “Version Control” ?
- Basics of Version Control (VC)
- Git as a VC solution
- Being Git practical with GitHub/GitLab
- Conclusions & reference

Example 1, “last version?”

Real life example

- Hey can you send me the source of that article XYZ?
- Sure, …hum, well, …

`article.tar.bz2`	There it is!
`article_final.tar.bz2`	No this one is more recent
`article_final2.tar.bz2`	Wait, this is one even more so
`article_last.tar.bz2`	Hold on, that should be it
`article_20180705_bis.tar.bz2`	Or maybe…

- Poor man’s versioning → date & comment in archive file name

  BUT you do not know <u>what</u> is different from one version to the other!!!

Example 2, “conference abstract and presentation?”

Real life example #2,

Big international conference in October, with abstract/short-paper deadline in March

- in March create results, plots & graphs + write submission
- from March to October, keep on working *on code and data*
- in September, prepare your oral presentation or poster…
  - can you reproduce results, plots & graphs from March?
  - if different, which one is “correct” ? And why?
  - code difference is improvement, new bug or bug fix?

Example 3, “collaborate?”

- One person in charge

  Send an email with:

  “Changes made:

  - updated help part of file1.m
  - corrected a bug in file2.m
  - Added a new feature to handle .png images in file3.m

  See the attached files.”

- One shared file, e.g. through Dropbox or on server

  → Incompatible parallel versions, overwritten files, lost changes,… depending on “who saved last”

  And still no idea of what differs across versions!

Example 4, “mess with yourself!”

A simple way to “shoot oneself in the foot”:

1. Take a snap shot archive of current stable version
   commonly “copy your code in a new folder”.
2. Begin implementing your new crazy experimental idea.
3. Fix some bugs in old code, revealed during testing.
4. Your idea was crap, discard experimental version.
5. Start back from stable version archive.
6. You lost your bug fixes, which also applied to the stable version...
   Or was it ?

Why Version Control

Key questions:

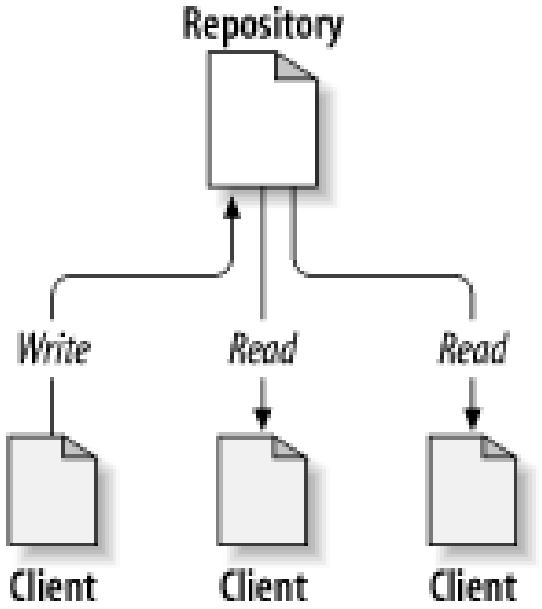
- Do you work in a team?
- Has it ever happened that you were working on a file, and someone else was working on the same file at the same time? Did you lose your changes to that file because of that? Or ended up with incompatible code?
- Have you ever saved a file, and then wanted to revert the changes you made? Have you ever wished you could see what a file looked like some time ago?
- Have you ever found a bug/error in your project and wanted to know when that bug got into your files?

If any “Yes”, then use a VC system !

Program

- Why “Version Control” ?
- Basics of Version Control (VC)
- Git as a VC solution
- Being Git practical with GitHub/GitLab
- Conclusions & reference

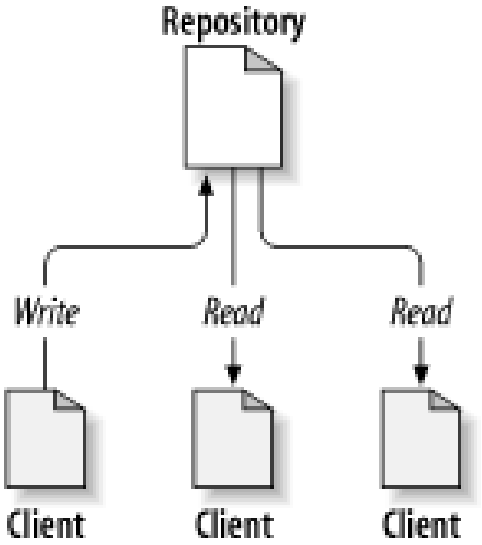
Centralized file management

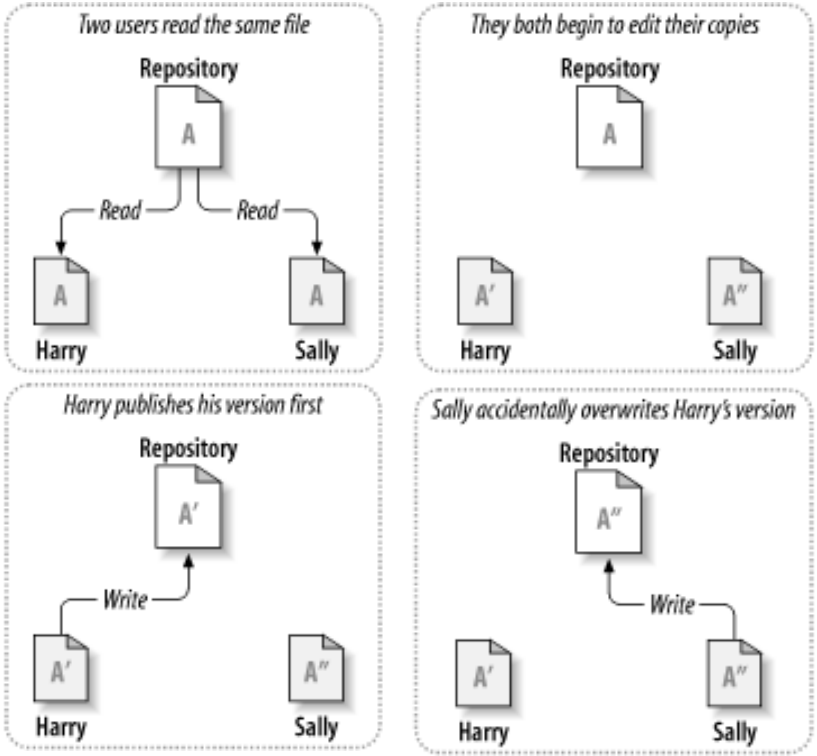


- One central **repository**, on a **server**.
- (Stores the files and their history.)
- Many **clients**, i.e. users, connecting to the repository.
- Each client has one or more **working copies**, i.e. a local copy of the files, where changes are made

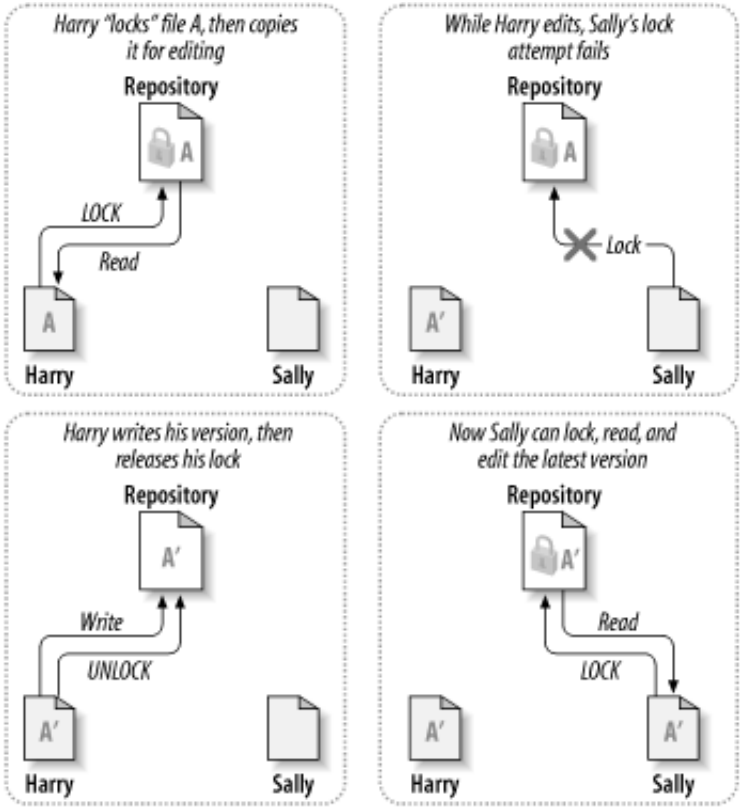
File sharing & Collaboration Problem

Centralized VC model

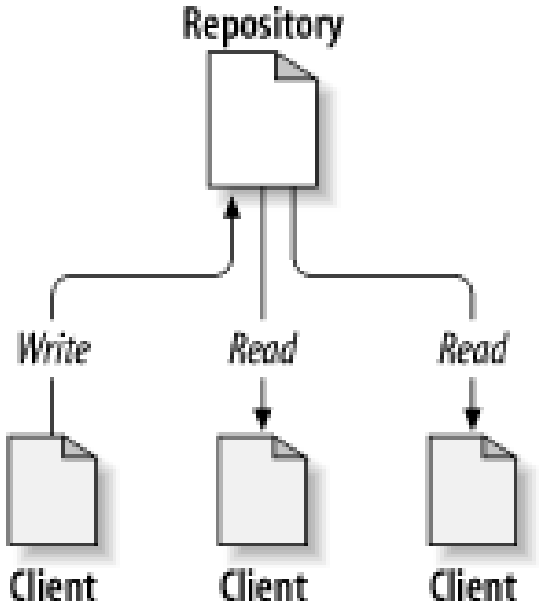




Locking solution



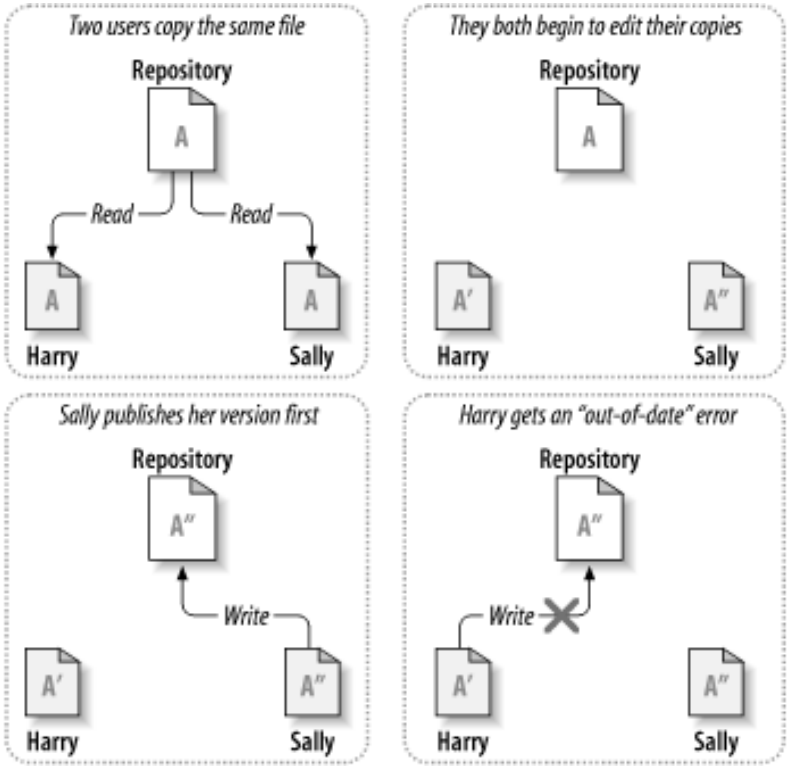
Centralized VC model



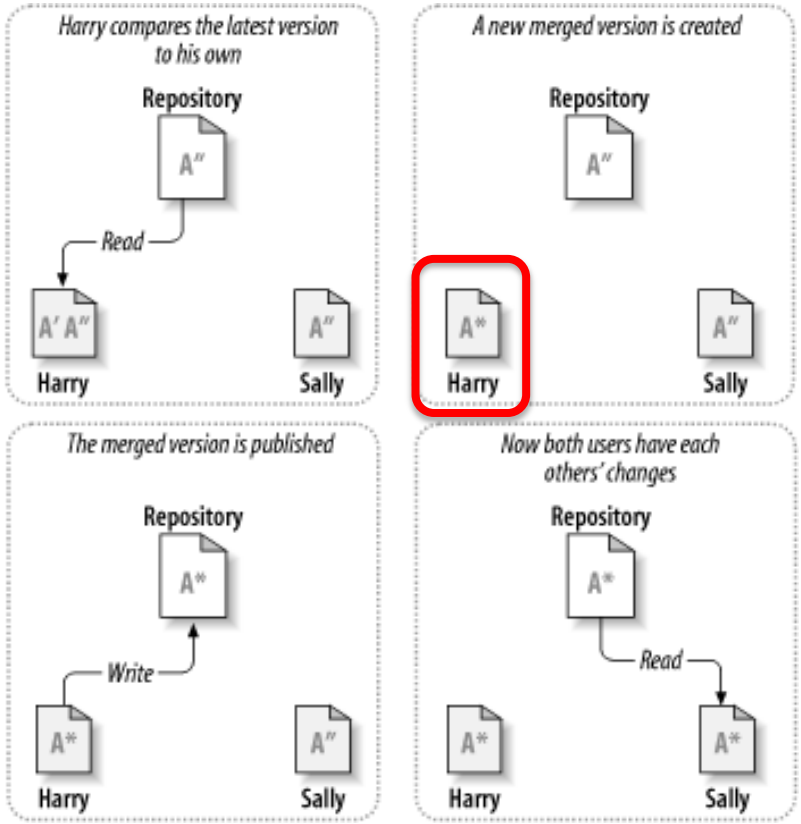
- One central **repository**, on a server.
- Stores the **files** and their **history**.
- Many **clients**, i.e. users connecting to the repo
- Each client has one or more **working copies**, i.e. a local copy of the files, where changes are made

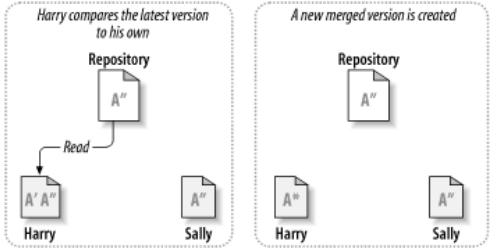
- A **revision** identifies a point in time of the repo, it is denoted by a number.

Copy-Modify-Merge Solution



Copy-Modify-Merge Solution





File merging & conflicts

When updating files are “updated” automatically.

- **Merged** files:

  all changes, yours & from server, are automatically merged into ***your*** files (if possible).

  → manual check recommended…

- **Conflicted** files:

  your changes and those on the server are NOT compatible, no automatic merging possible

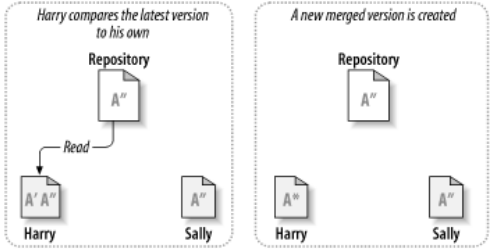
  → manual intervention necessary! **Your** responsibility.

Resolving conflicts

When updating your working copy:

- If some files have changed *both in the repository and in your working copy*, there can be a **conflict**
- It is **your responsibility** to fix conflicts, by inspecting the diverging changes and
  - choose your own version, or
  - choose repository version, or
  - choose previous version, or
  - mix both versions

Binary files…



- Merging works on text-based files (code/document)
- With binary files (images, `.ppt`, `.pdf`, `.doc`, `.xls`, …)

  → Updating overwrites the file…

  but previous versions still available in history!
- Use simple text (`.txt`), Markdown (`.md`) , Latex (`.tex`/`.bib`), comma-/tab-separated values (`.csv`/`.tsv`) or JSON (`.json`) files instead of Word or Excel files !

How to…

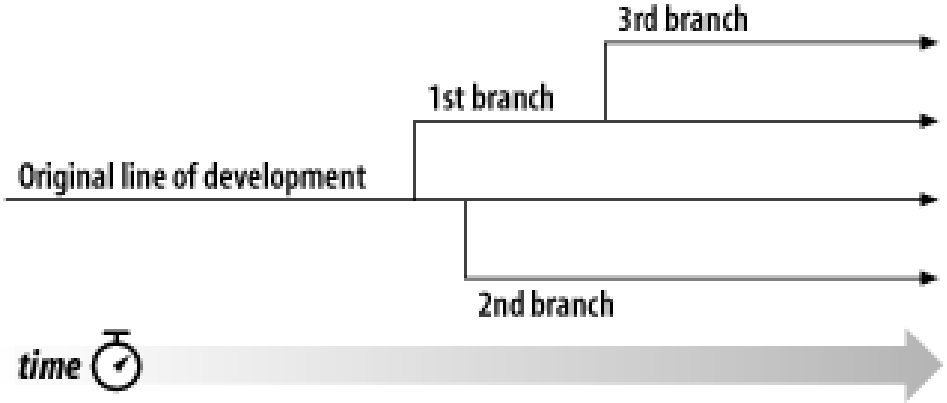
- Create repository or get code from repository:
  - check out/clone code, or update code
- Work on your code/files:
  - bug fixes and/or new features
- Publish your changes to the repository
  - re-updating and fixing conflicts, if necessary

Note:

- Split your commits into logical steps
- Add description!!!

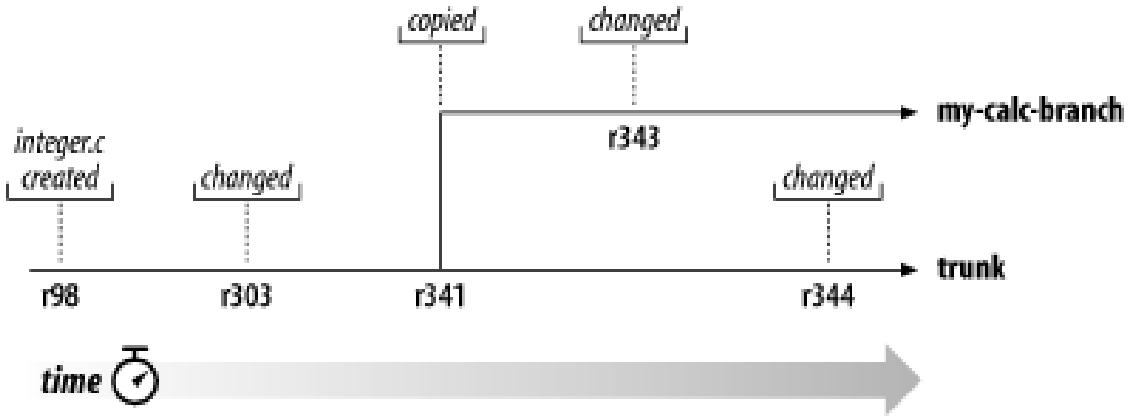
Code branch

"…a line of *development that exists independently of another line, yet still shares a common history if you look far enough back in time. A branch always begins life as a copy of something, and moves on from there, generating its own history.*"



Branching

- Work on a branch as you would on any other folder, e.g. `code_v1`, `code_v2`,…
- File histories in branches also stored!



Branch merging

= synchronizing two branches

- When developing a branch, you'll want to synch with "main trunk" from time to time (e.g. for bug fixes)
- When merging, you can encounter conflicts, to be resolved as before
- If you want to integrate a branch back to "main trunk", you can merge it back (e.g. adding new features).

Program

- Why “Version Control” ?
- Basics of Version Control (VC)
- Git as a VC solution
- Being Git practical with GitHub/GitLab
- Conclusions & reference

What is “Git” ?

- currently the most popular distributed versioning system
- free open-source software
- cross-platform (originally for Linux but now also on MacOS and Windows)
- very efficient, very powerful but can be very complex
- some GUIs and IDEs plugins
- no global revision numbers, “hashes” instead
- created by Linus Torvalds, 1st release in 2005

Git, pro's & con's

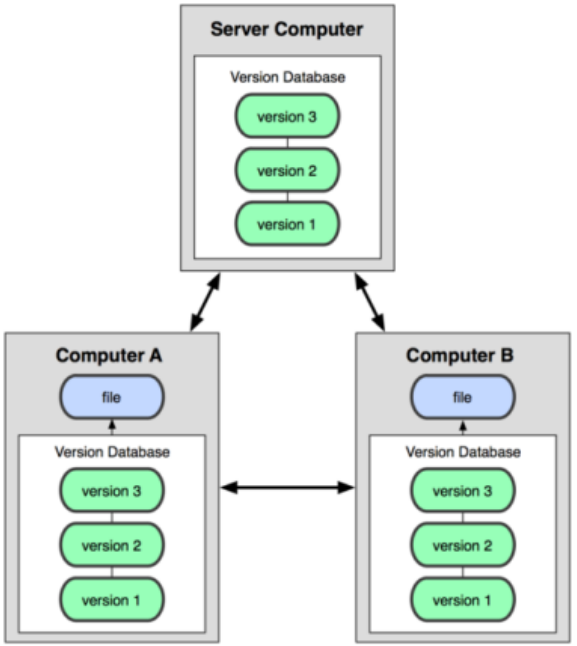
Pro's

- Every working copy is a full backup of the data
- You can work off-line
- You can do micro-commits
- Allows private work, eases experimental jump in

Cons

- More complex (decentralized → "parallel worlds")
- Less control on project evolution
- Less sharing?

Decentralized model



Git Cheat Sheet

Git Basics

`git init <directory>`	Create empty Git repo in specified directory. Run with no arguments to initialize the current directory as a git repository.
`git clone <repo>`	Clone repo located at `<repo>` onto local machine. Original repo can be located on the local filesystem or on a remote machine via `HTTP` or `SSH`.
`git config user.name <name>`	Define author name to be used for all commits in current repo. Devs commonly use `--global` flag to set config options for current user.
`git add <directory>`	Stage all changes in `<directory>` for the next commit. Replace `<directory>` with a `<file>` to change a specific file.
`git commit -m "<message>"`	Commit the staged snapshot, but instead of launching a text editor, use `<message>` as the commit message.
`git status`	List which files are staged, unstaged, and untracked.
`git log`	Display the entire commit history using the default format. For customization see additional options.
`git diff`	Show unstaged changes between your index and working directory.

Undoing Changes

`git revert <commit>`	Create new commit that undoes all of the changes made in `<commit>`, then apply it to the current branch.
`git reset <file>`	Remove `<file>` from the staging area, but leave the working directory unchanged. This unstages a file without overwriting any changes.
`git clean -n`	Shows which files would be removed from working directory. Use the `-f` flag in place of the `-n` flag to execute the clean.

Rewriting Git History

`git commit --amend`	Replace the last commit with the staged changes and last commit combined. Use with nothing staged to edit the last commit's message.
`git rebase <base>`	Rebase the current branch onto `<base>`. `<base>` can be a commit ID, a branch name, a tag, or a relative reference to `HEAD`.
`git reflog`	Show a log of changes to the local repository's `HEAD`. Add `--relative-date` flag to show date info or `--all` to show all refs.

Git Branches

`git branch`	List all of the branches in your repo. Add a `<branch>` argument to create a new branch with the name `<branch>`.
`git checkout -b <branch>`	Create and check out a new branch named `<branch>`. Drop the `-b` flag to checkout an existing branch.
`git merge <branch>`	Merge `<branch>` into the current branch.

Remote Repositories

`git remote add <name> <url>`	Create a new connection to a remote repo. After adding a remote, you can use `<name>` as a shortcut for `<url>` in other commands.
`git fetch <remote> <branch>`	Fetches a specific `<branch>`, from the repo. Leave off `<branch>` to fetch all remote refs.
`git pull <remote>`	Fetch the specified remote's copy of current branch and immediately merge it into the local copy.
`git push <remote> <branch>`	Push the branch to `<remote>`, along with necessary commits and objects. Creates named branch in the remote repo if it doesn't exist.

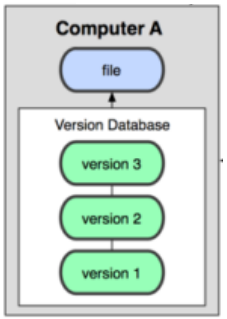
Git, notes

- If the git repository only exist on your machine or one single computer/drive, then
  - you are at risk of losing everything!
  - no easy collaboration

  ⇒ **use an external server to sync' with**
- Only text files or *light* (<10MB) binary files

  ⇒ **No dataset or heavy binary files !**

  (use other tools)



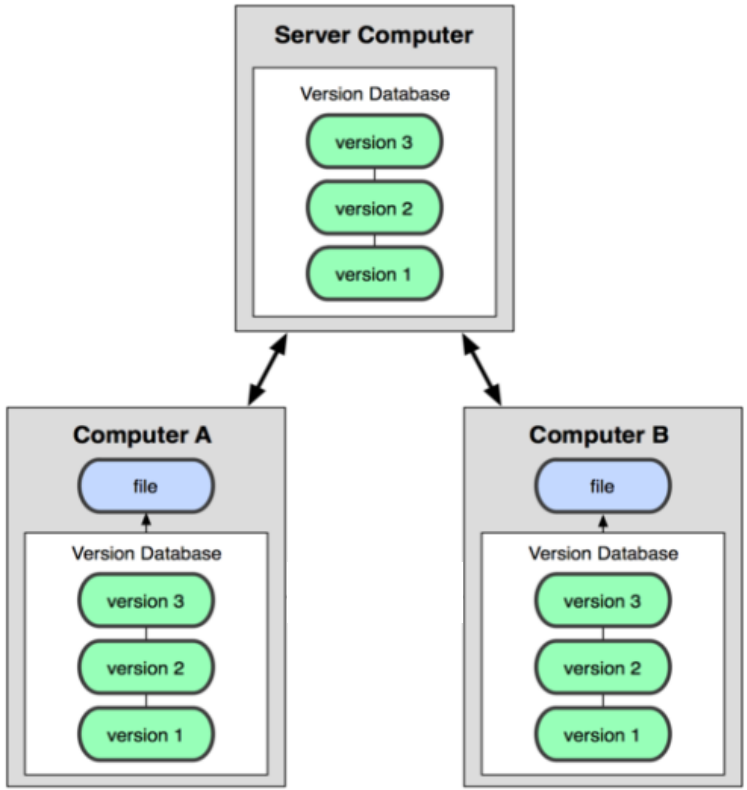
Program

- Why “Version Control” ?
- Basics of Version Control (VC)
- Git as a VC solution
- Being Git practical with GitHub/GitLab
- Conclusions & reference

Git & GitHub/GitLab

https://github.com
https://gitlab.uliege.be/

- Git
  - “version control system” software
  - language with its commands
- GitHub.com (& GitLab.com)
  - web-based Git repository hosting system
  - servers from a *private company*
- GitLab.uliege.be
  - web-based Git repository hosting system
  - hosted at ULiège. ☺

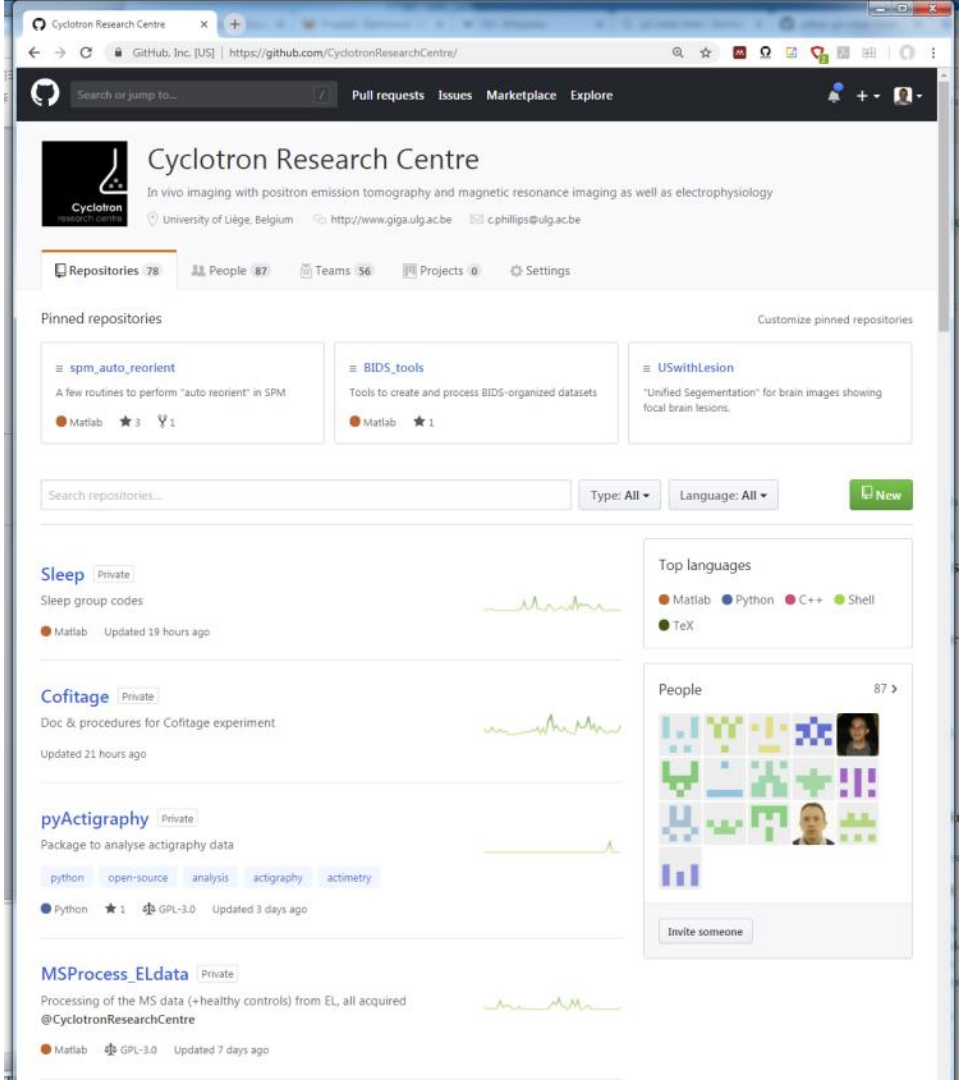


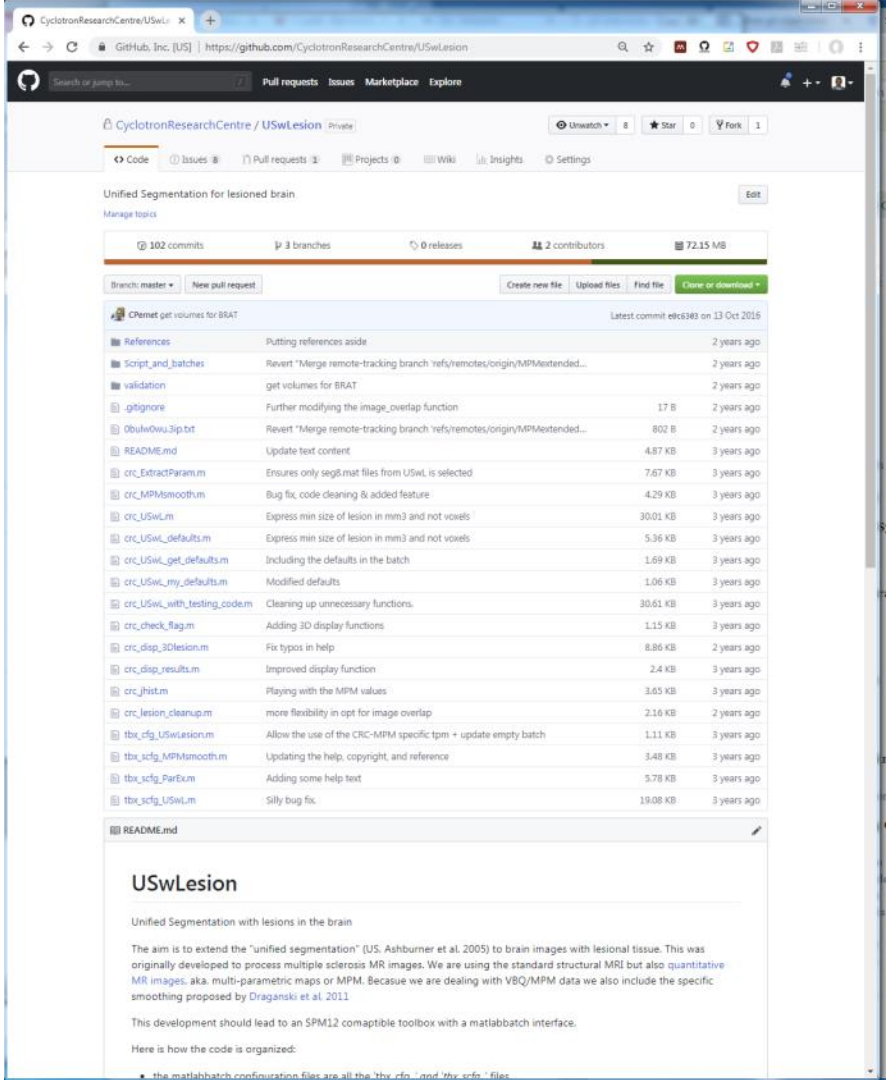
GitHub & GitLab features

- Code versioning
  + branching, merging, releases

And more…

- Code documentation and Wiki
  → build knowledge for the team
- Issue tracking
  → discuss problems & requests in a forum, keep track of decisions!
- Management
  → access rights, visibility, groups/teams, …


Cyclotron Research Centre
GitHub, Inc. [US] | https://github.com/CyclotronResearchCentre/
Search or jump to...
Pull requests
Issues
Marketplace
Explore
Cyclotron Research Centre
In vivo imaging with positron emission tomography and magnetic resonance imaging as well as electrophysiology
University of Liège, Belgium
http://www.giga.ulg.ac.be
c.phillips@ulg.ac.be
Repositories 78
People 87
Teams 56
Projects 0
Settings
Pinned repositories
Customize pinned repositories
spm_auto_reorient
A few routines to perform "auto reorient" in SPM
Matlab 3 1
BIDS_tools
Tools to create and process BIDS-organized datasets
Matlab 1
USwithLesion
"Unified Segementation" for brain images showing focal brain lesions.
Search repositories...
Type: All
Language: All
New
Sleep Private
Sleep group codes
Matlab Updated 19 hours ago
Cofitage Private
Doc & procedures for Cofitage experiment
Updated 21 hours ago
pyActigraphy Private
Package to analyse actigraphy data
python open-source analysis actigraphy actimetry
Python 1 GPL-3.0 Updated 3 days ago
MSProcess_ELdata Private
Processing of the MS data (+healthy controls) from EL, all acquired @CyclotronResearchCentre
Matlab GPL-3.0 Updated 7 days ago
Top languages
Matlab Python C++ Shell TeX
People 87 ›
Invite someone



CyclotronResearchCentre/USwLe

GitHub, Inc. [US] | https://github.com/CyclotronResearchCentre/USwLesion

Pull requests Issues Marketplace Explore

CyclotronResearchCentre / USwLesion Private

Unwatch 8 Star 0 Fork 1

Code Issues 8 Pull requests 1 Projects 0 Wiki Insights Settings

Unified Segmentation for lesioned brain

Edit

Manage topics

102 commits 3 branches 0 releases 2 contributors 72.15 MB

Branch: master New pull request Create new file Upload files Find file Clone or download

CPernet get volumes for BRAT Latest commit e8c6383 on 13 Oct 2016

References	Putting references aside		2 years ago
Script_and_batches	Revert "Merge remote-tracking branch 'refs/remotes/origin/MPMextended...		2 years ago
validation	get volumes for BRAT		2 years ago
.gitignore	Further modifying the image_overlap function	17 B	2 years ago
0bulw0wu.3ip.txt	Revert "Merge remote-tracking branch 'refs/remotes/origin/MPMextended...	802 B	2 years ago
README.md	Update text content	4.87 KB	3 years ago
crc_ExtractParam.m	Ensures only seg8.mat files from USwL is selected	7.67 KB	3 years ago
crc_MPMsmooth.m	Bug fix, code cleaning & added feature	4.29 KB	3 years ago
crc_USwL.m	Express min size of lesion in mm3 and not voxels	30.01 KB	3 years ago
crc_USwL_defaults.m	Express min size of lesion in mm3 and not voxels	5.36 KB	3 years ago
crc_USwL_get_defaults.m	Including the defaults in the batch	1.69 KB	3 years ago
crc_USwL_my_defaults.m	Modified defaults	1.06 KB	3 years ago
crc_USwL_with_testing_code.m	Cleaning up unnecessary functions.	30.61 KB	3 years ago
crc_check_flag.m	Adding 3D display functions	1.15 KB	3 years ago
crc_disp_3Dlesion.m	Fix typos in help	8.86 KB	2 years ago
crc_disp_results.m	Improved display function	2.4 KB	3 years ago
crc_jhist.m	Playing with the MPM values	3.65 KB	3 years ago
crc_lesion_cleanup.m	more flexibility in opt for image overlap	2.16 KB	2 years ago
tbx_cfg_USwLesion.m	Allow the use of the CRC-MPM specific tpm + update empty batch	1.11 KB	3 years ago
tbx_scfg_MPMsmooth.m	Updating the help, copyright, and reference	3.48 KB	3 years ago
tbx_scfg_ParEx.m	Adding some help text	5.78 KB	3 years ago
tbx_scfg_USwL.m	Silly bug fix.	19.08 KB	3 years ago

README.md

USwLesion

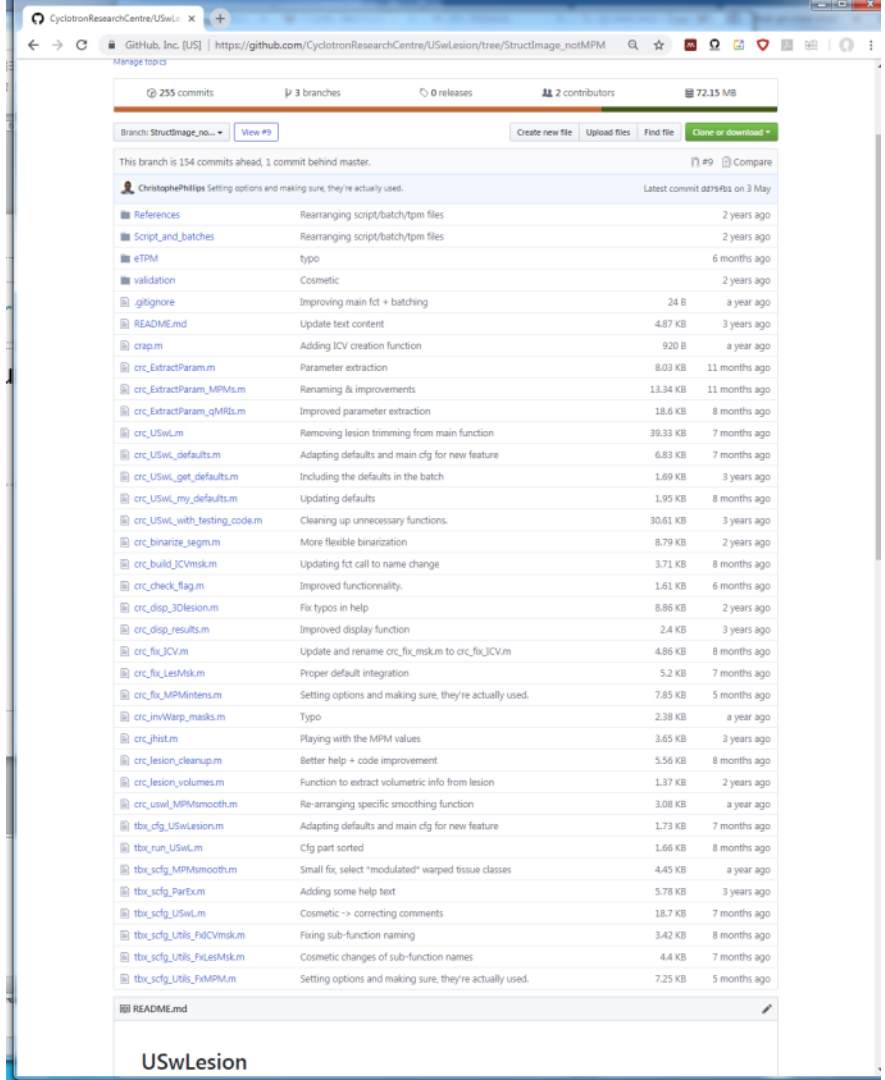
Unified Segmentation with lesions in the brain

The aim is to extend the "unified segmentation" (US. Ashburner et al. 2005) to brain images with lesional tissue. This was originally developed to process multiple sclerosis MR images. We are using the standard structural MRI but also quantitative MR images, aka. multi-parametric maps or MPM. Becasue we are dealing with VBQ/MPM data we also include the specific smoothing proposed by Draganski et al. 2011

This development should lead to an SPM12 comaptible toolbox with a matlabbatch interface.

Here is how the code is organized:

- the matlabbatch configuration files are all the 'tbx_cfg_*.mat' and 'tbx_scfg_*' files



CyclotronResearchCentre/USwLe

GitHub, Inc. [US] | https://github.com/CyclotronResearchCentre/USwLesion/tree/StructImage_notMPM

Manage topics

255 commits 3 branches 0 releases 2 contributors 72.15 MB

Branch: StructImage_no... View #9 Create new file Upload files Find file Clone or download

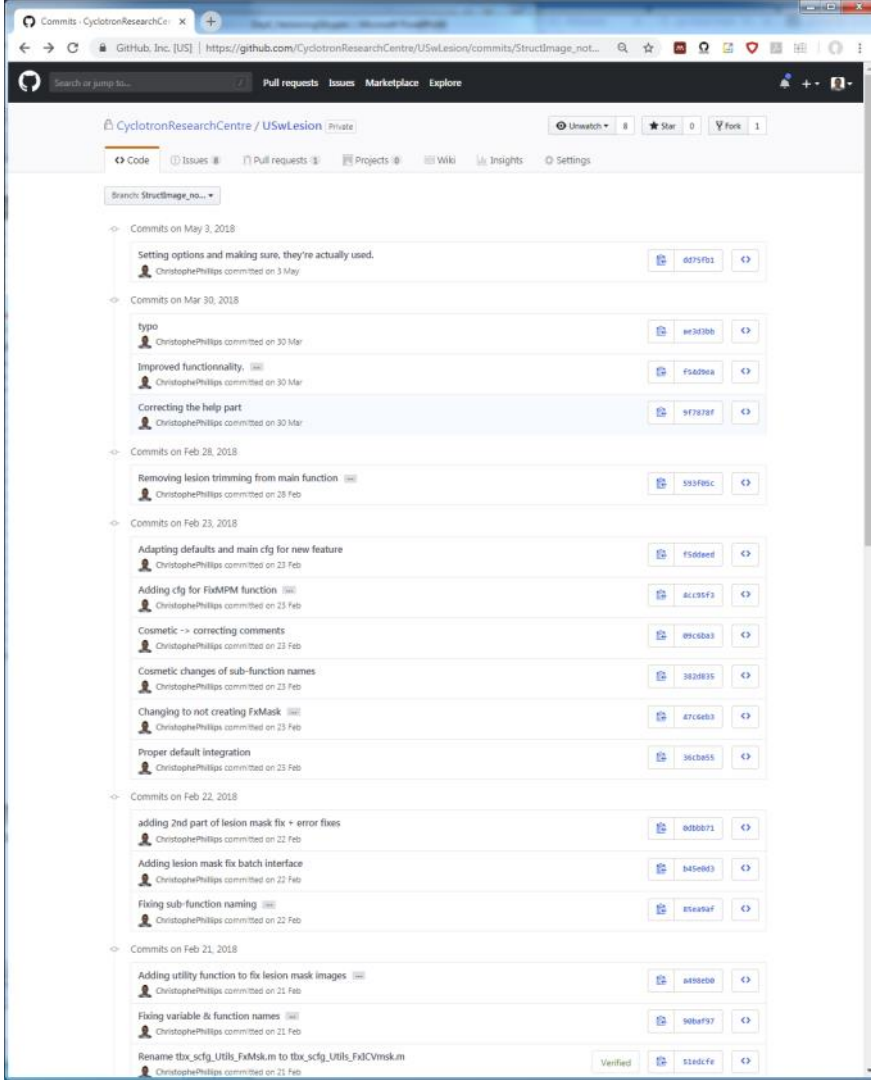
This branch is 154 commits ahead, 1 commit behind master. #9 Compare

ChristophePhilips Setting options and making sure, they're actually used. Latest commit dd7bfb1 on 3 May

References	Rearranging script/batch/tpm files		2 years ago
Script_and_batches	Rearranging script/batch/tpm files		2 years ago
eTPM	typo		6 months ago
validation	Cosmetic		2 years ago
.gitignore	Improving main fct + batching	24 B	a year ago
README.md	Update text content	4.87 KB	3 years ago
crap.m	Adding ICV creation function	920 B	a year ago
crc_ExtractParam.m	Parameter extraction	8.03 KB	11 months ago
crc_ExtractParam_MPMs.m	Renaming & improvements	13.34 KB	11 months ago
crc_ExtractParam_qMRIs.m	Improved parameter extraction	18.6 KB	8 months ago
crc_USwL.m	Removing lesion trimming from main function	39.33 KB	7 months ago
crc_USwL_defaults.m	Adapting defaults and main cfg for new feature	6.83 KB	7 months ago
crc_USwL_get_defaults.m	Including the defaults in the batch	1.69 KB	3 years ago
crc_USwL_my_defaults.m	Updating defaults	1.95 KB	8 months ago
crc_USwL_with_testing_code.m	Cleaning up unnecessary functions.	30.61 KB	3 years ago
crc_binarize_segm.m	More flexible binarization	8.79 KB	2 years ago
crc_build_ICVmsk.m	Updating fct call to name change	3.71 KB	8 months ago
crc_check_flag.m	Improved functionnality.	1.61 KB	6 months ago
crc_disp_3Dlesion.m	Fix typos in help	8.86 KB	2 years ago
crc_disp_results.m	Improved display function	2.4 KB	3 years ago
crc_fix_ICV.m	Update and rename crc_fix_msk.m to crc_fix_ICV.m	4.86 KB	8 months ago
crc_fix_LesMsk.m	Proper default integration	5.2 KB	7 months ago
crc_fix_MPMintens.m	Setting options and making sure, they're actually used.	7.85 KB	5 months ago
crc_invWarp_masks.m	Typo	2.38 KB	a year ago
crc_jhist.m	Playing with the MPM values	3.65 KB	3 years ago
crc_lesion_cleanup.m	Better help + code improvement	5.56 KB	8 months ago
crc_lesion_volumes.m	Function to extract volumetric info from lesion	1.37 KB	2 years ago
crc_uswl_MPMsmooth.m	Re-arranging specific smoothing function	3.08 KB	a year ago
tbx_cfg_USwLesion.m	Adapting defaults and main cfg for new feature	1.73 KB	7 months ago
tbx_run_USwL.m	Cfg part sorted	1.66 KB	8 months ago
tbx_scfg_MPMsmooth.m	Small fix, select "modulated" warped tissue classes	4.45 KB	a year ago
tbx_scfg_ParEx.m	Adding some help text	5.78 KB	3 years ago
tbx_scfg_USwL.m	Cosmetic -> correcting comments	18.7 KB	7 months ago
tbx_scfg_Utils_FxICVmsk.m	Fixing sub-function naming	3.42 KB	8 months ago
tbx_scfg_Utils_FxLesMsk.m	Cosmetic changes of sub-function names	4.4 KB	7 months ago
tbx_scfg_Utils_FxMPM.m	Setting options and making sure, they're actually used.	7.25 KB	5 months ago

README.md

USwLesion



Commits · CyclotronResearchCentre / USwLesion

GitHub, Inc. [US] | https://github.com/CyclotronResearchCentre/USwLesion/commits/StructImage_not...

Pull requests Issues Marketplace Explore

CyclotronResearchCentre / USwLesion Private

Unwatch 8 Star 0 Fork 1

Code Issues 8 Pull requests 1 Projects 0 Wiki Insights Settings

Branch: StructImage_no...

Commits on May 3, 2018

Setting options and making sure, they're actually used. — ChristophePhillips committed on 3 May — dd75fb1

Commits on Mar 30, 2018

typo — ChristophePhillips committed on 30 Mar — ae3d3bb

Improved functionnality. — ChristophePhillips committed on 30 Mar — f5a09ea

Correcting the help part — ChristophePhillips committed on 30 Mar — 9f787df

Commits on Feb 28, 2018

Removing lesion trimming from main function — ChristophePhillips committed on 28 Feb — 593f05c

Commits on Feb 23, 2018

Adapting defaults and main cfg for new feature — ChristophePhillips committed on 23 Feb — f5ddaed

Adding cfg for FixMPM function — ChristophePhillips committed on 23 Feb — 4cc95f3

Cosmetic -> correcting comments — ChristophePhillips committed on 23 Feb — 09c6ba3

Cosmetic changes of sub-function names — ChristophePhillips committed on 23 Feb — 382d835

Changing to not creating FxMask — ChristophePhillips committed on 23 Feb — 47c6eb3

Proper default integration — ChristophePhillips committed on 23 Feb — 36cba55

Commits on Feb 22, 2018

adding 2nd part of lesion mask fix + error fixes — ChristophePhillips committed on 22 Feb — 0dbbb71

Adding lesion mask fix batch interface — ChristophePhillips committed on 22 Feb — b45e0d3

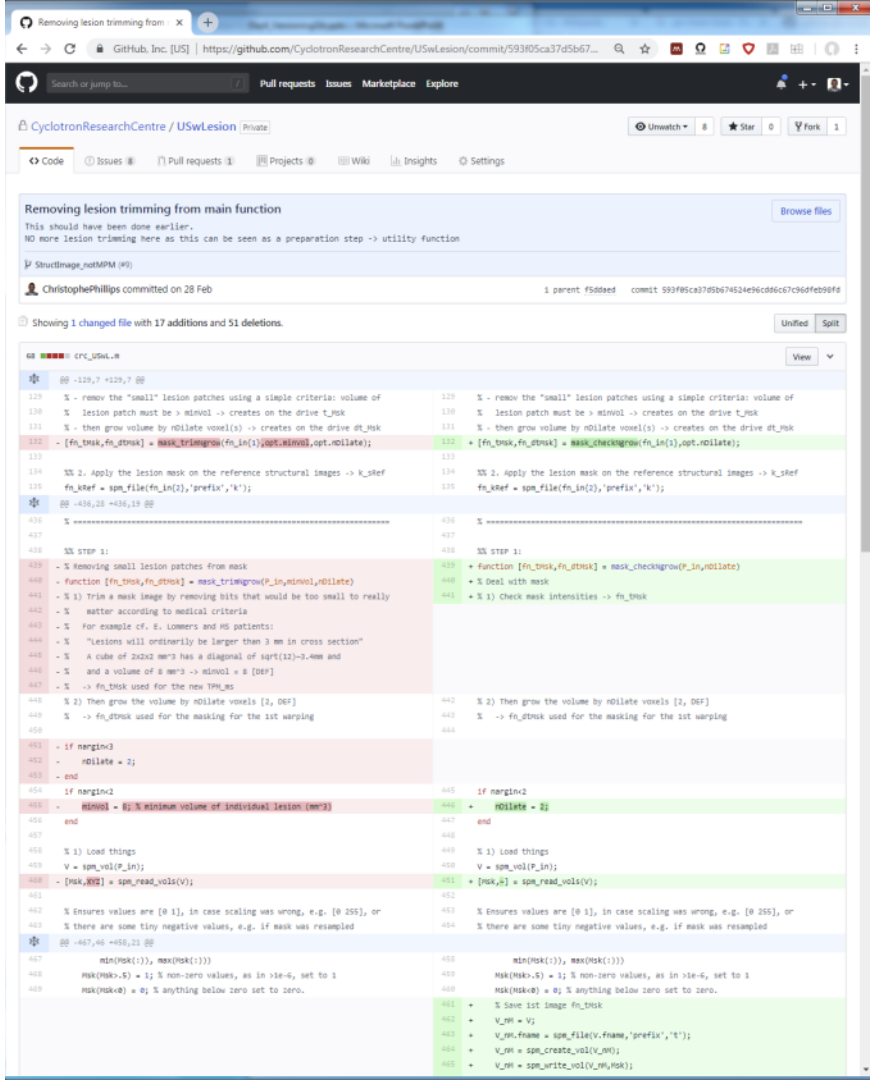
Fixing sub-function naming — ChristophePhillips committed on 22 Feb — 85ea9af

Commits on Feb 21, 2018

Adding utility function to fix lesion mask images — ChristophePhillips committed on 21 Feb — a498eb0

Fixing variable & function names — ChristophePhillips committed on 21 Feb — 90baf97

Rename tbx_scfg_Utils_FxMsk.m to tbx_scfg_Utils_FxJCVmsk.m — ChristophePhillips committed on 21 Feb — Verified — 51edcfe



Removing lesion trimming from main function

GitHub, Inc. [US] | https://github.com/CyclotronResearchCentre/USwLesion/commit/593f05ca37d5b67...

Pull requests Issues Marketplace Explore

CyclotronResearchCentre / USwLesion Private

Unwatch 8 Star 0 Fork 1

Code Issues 8 Pull requests 1 Projects 0 Wiki Insights Settings

Removing lesion trimming from main function — Browse files

This should have been done earlier.
NO more lesion trimming here as this can be seen as a preparation step -> utility function

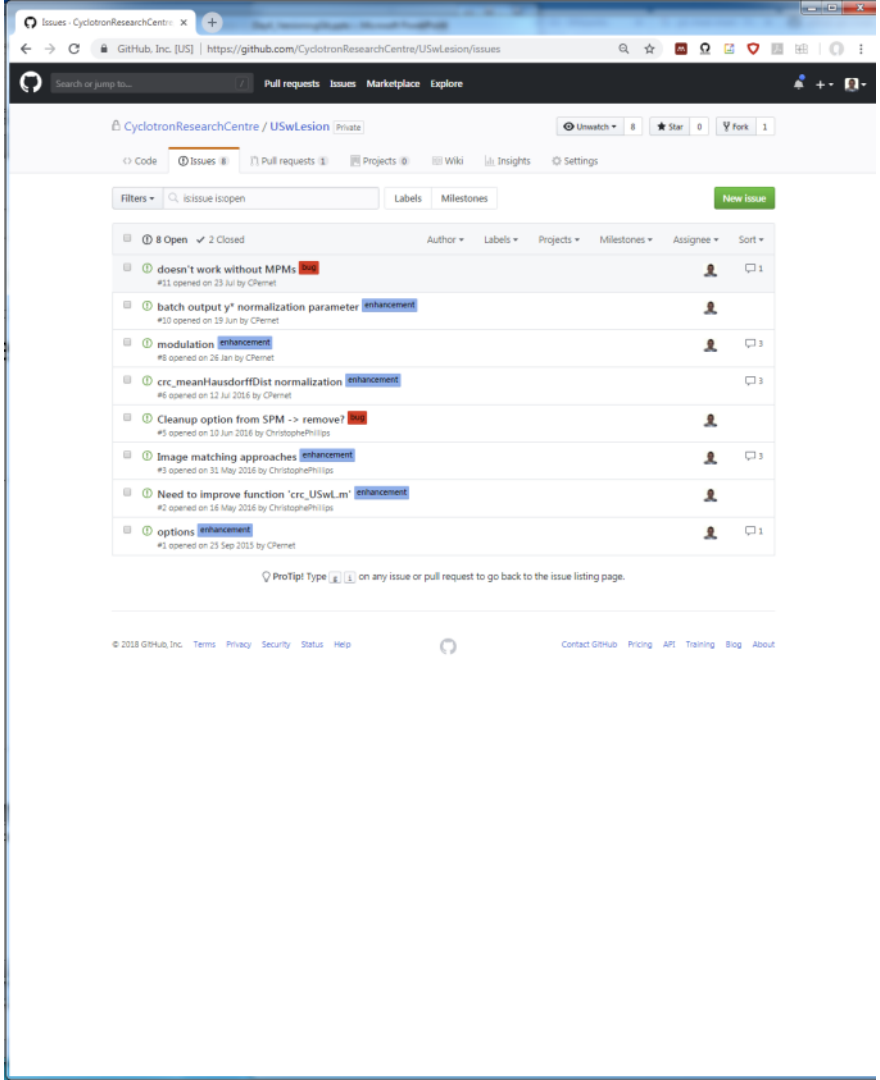
StructImage_notMPM (#9)

ChristophePhillips committed on 28 Feb — 1 parent f5ddaed — commit 593f05ca37d5b674524e96cdd6c67c96dfeb98fd

Showing 1 changed file with 17 additions and 51 deletions. — Unified | Split

crc_USwL.m — View

```
@@ -129,7 +129,7 @@
  % - remov the "small" lesion patches using a simple criteria: volume of
  %   lesion patch must be > minVol -> creates on the drive t_Msk
  % - then grow volume by nDilate voxel(s) -> creates on the drive dt_Msk
- [fn_tMsk,fn_dtMsk] = mask_trimNgrow(fn_in(1),opt.minVol,opt.nDilate);
+ [fn_tMsk,fn_dtMsk] = mask_checkNgrow(fn_in(1),opt.nDilate);

  %% 2. Apply the lesion mask on the reference structural images -> k_sRef
  fn_kRef = spm_file(fn_in(2),'prefix','k');
@@ -436,28 +436,19 @@
  %% STEP 1:
- % Removing small lesion patches from mask
- function [fn_tMsk,fn_dtMsk] = mask_trimNgrow(P_in,minVol,nDilate)
- % 1) Trim a mask image by removing bits that would be too small to really
- %    matter according to medical criteria
- %  For example cf. E. Lommers and HS patients:
- %  "Lesions will ordinarily be larger than 3 mm in cross section"
- %  A cube of 2x2x2 mm^3 has a diagonal of sqrt(12)~3.4mm and
- %  and a volume of 8 mm^3 -> minVol = 8 [DEF]
- %  -> fn_tMsk used for the new TPM_ms
+ function [fn_tMsk,fn_dtMsk] = mask_checkNgrow(P_in,nDilate)
+ % Deal with mask
+ % 1) Check mask intensities -> fn_tMsk

  % 2) Then grow the volume by nDilate voxels [2, DEF]
  %  -> fn_dtMsk used for the masking for the 1st warping

- if nargin<3
-     nDilate = 2;
- end
  if nargin<2
-     minVol = 8; % minimum volume of individual lesion (mm^3)
+     nDilate = 2;
  end

  % 1) Load things
  V = spm_vol(P_in);
- [Msk,XYZ] = spm_read_vols(V);
+ [Msk,~] = spm_read_vols(V);

  % Ensures values are [0 1], in case scaling was wrong, e.g. [0 255], or
  % there are some tiny negative values, e.g. if mask was resampled
@@ -467,46 +458,21 @@
          min(Msk(:)), max(Msk(:)))
      Msk(Msk>.5) = 1; % non-zero values, as in >1e-6, set to 1
      Msk(Msk<0) = 0; % anything below zero set to zero.
+     % Save 1st image fn_tMsk
+     V_nM = V;
+     V_nM.fname = spm_file(V.fname,'prefix','t');
+     V_nM = spm_create_vol(V_nM);
+     V_nM = spm_write_vol(V_nM,Msk);
```



Issues · CyclotronResearchCentre

GitHub, Inc. [US] | https://github.com/CyclotronResearchCentre/USwLesion/issues

Pull requests Issues Marketplace Explore

CyclotronResearchCentre / USwLesion Private

Unwatch 8 Star 0 Fork 1

Code Issues 8 Pull requests 1 Projects 0 Wiki Insights Settings

Filters is:issue is:open Labels Milestones New issue

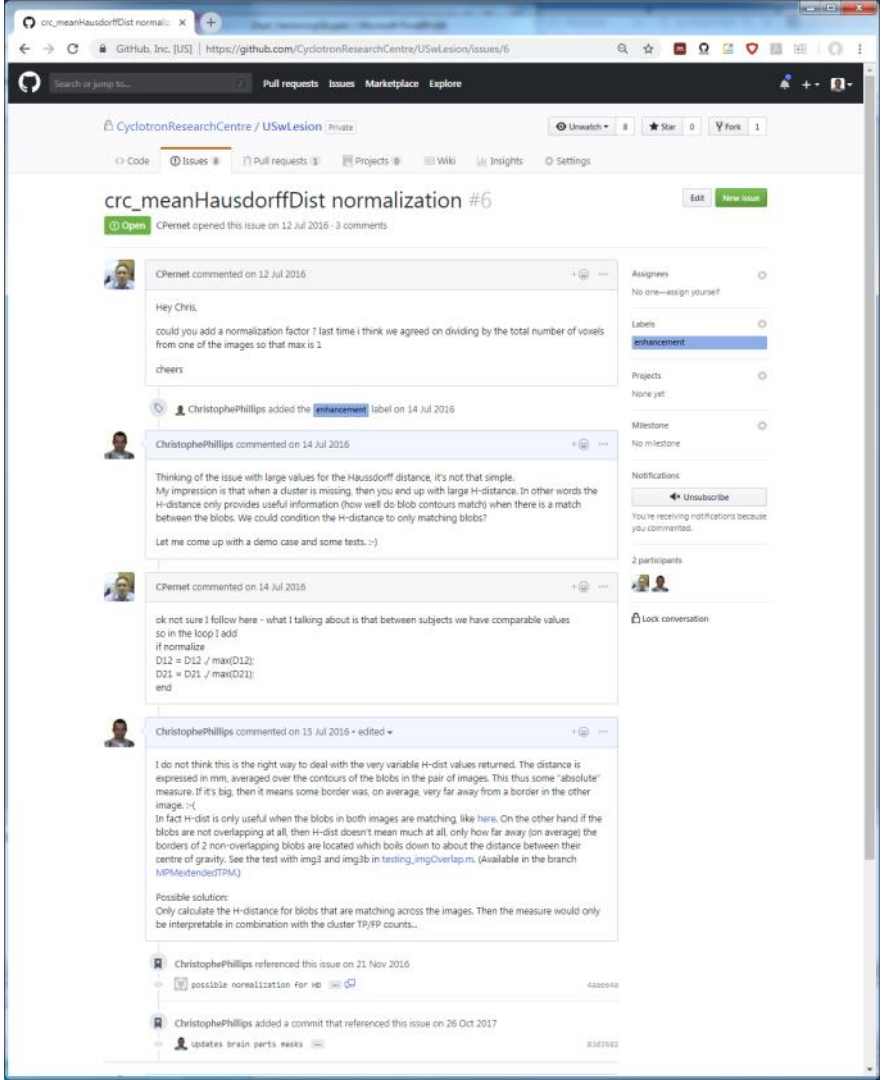
8 Open ✓ 2 Closed Author Labels Projects Milestones Assignee Sort

- doesn't work without MPMs bug
  #11 opened on 23 Jul by CPernet
- batch output y* normalization parameter enhancement
  #10 opened on 19 Jun by CPernet
- modulation enhancement
  #8 opened on 26 Jan by CPernet
- crc_meanHausdorffDist normalization enhancement
  #6 opened on 12 Jul 2016 by CPernet
- Cleanup option from SPM -> remove? bug
  #5 opened on 10 Jun 2016 by ChristophePhillips
- Image matching approaches enhancement
  #3 opened on 31 May 2016 by ChristophePhillips
- Need to improve function 'crc_USwL.m' enhancement
  #2 opened on 16 May 2016 by ChristophePhillips
- options enhancement
  #1 opened on 25 Sep 2015 by CPernet

ProTip! Type g i on any issue or pull request to go back to the issue listing page.

© 2018 GitHub, Inc. Terms Privacy Security Status Help

Contact GitHub Pricing API Training Blog About



crc_meanHausdorffDist normali

GitHub, Inc. [US] | https://github.com/CyclotronResearchCentre/USwLesion/issues/6

Pull requests Issues Marketplace Explore

CyclotronResearchCentre / USwLesion Private

Unwatch 8 Star 0 Fork 1

Code Issues 8 Pull requests 1 Projects 0 Wiki Insights Settings

crc_meanHausdorffDist normalization #6

Edit New issue

Open CPernet opened this issue on 12 Jul 2016 · 3 comments

CPernet commented on 12 Jul 2016

Hey Chris,

could you add a normalization factor ? last time i think we agreed on dividing by the total number of voxels from one of the images so that max is 1

cheers

ChristophePhillips added the enhancement label on 14 Jul 2016

ChristophePhillips commented on 14 Jul 2016

Thinking of the issue with large values for the Haussdorff distance, it's not that simple.
My impression is that when a cluster is missing, then you end up with large H-distance. In other words the H-distance only provides useful information (how well do blob contours match) when there is a match between the blobs. We could condition the H-distance to only matching blobs?

Let me come up with a demo case and some tests. :-)

CPernet commented on 14 Jul 2016

ok not sure I follow here - what I talking about is that between subjects we have comparable values
so in the loop I add
if normalize
D12 = D12 ./ max(D12);
D21 = D21 ./ max(D21);
end

ChristophePhillips commented on 15 Jul 2016 • edited

I do not think this is the right way to deal with the very variable H-dist values returned. The distance is expressed in mm, averaged over the contours of the blobs in the pair of images. This thus some "absolute" measure. If it's big, then it means some border was, on average, very far away from a border in the other image. :-(
In fact H-dist is only useful when the blobs in both images are matching, like here. On the other hand if the blobs are not overlapping at all, then H-dist doesn't mean much at all, only how far away (on average) the borders of 2 non-overlapping blobs are located which boils down to about the distance between their centre of gravity. See the test with img3 and img3b in testing_imgOverlap.m. (Available in the branch MPMextendedTPM.)

Possible solution:
Only calculate the H-distance for blobs that are matching across the images. Then the measure would only be interpretable in combination with the cluster TP/FP counts...

ChristophePhillips referenced this issue on 21 Nov 2016

possible normalization for HD 4bb6648

ChristophePhillips added a commit that referenced this issue on 26 Oct 2017

updates brain parts masks 03d3582

Assignees
No one—assign yourself

Labels
enhancement

Projects
None yet

Milestone
No milestone

Notifications
Unsubscribe
You're receiving notifications because you commented.

2 participants

Lock conversation

GitHub.com vs GitLab.uliege.be

- GitHub.com (& GitLab.com)
  - useful for international projects & collaboration
  - ensures international visibility
  - can be more than just code (workshop, home page, CV,…)
- GitLab.uliege.be
  - hosted at Uliège by SeGI → safe & secure
  - easy local collaboration
  - lab knowledge with issues & wiki
  - still international visibility

Key difference is audience and membership management.

Program

- Why “Version Control” ?
- Basics of Version Control (VC)
- Git as a VC solution
- Being Git practical with GitHub/GitLab
- Conclusions & reference

Any good “reasons” not to VC ?

- “It’s only a small bit of code to try out an idea on my data…”

  → *This how breakthroughs happen and papers follow!*

- “Nobody else will ever be interested in this…”

  → *If you are, someone else will necessarily be!*

- “My code is not ready yet…”

  → *The ULTIMATE reason to actually version your code!*

Major hurdle is **psychological** or **carelessness**.

Some wisdom

"Writing software as if we are the only person that ever has to comprehend it is one of the biggest mistakes and false assumptions that can be made."

- Karolina Szczur

Code Versioning conclusion

- **Absolutely necessary to manage any project that relies on code, script, batch, text,…**
- Useful to keep track of changes, improvements & bug fixes over time
- Even more so with multiple developers/users

  → start alone → team interest → available to the community
- Open science → paper + code + data accessible

References

- J. D. Blischak et al., *A Quick Introduction to Version Control with Git and GitHub*, PLOS Computational Biology, 12(1): e1004668, 2016 http://dx.doi.org/10.1371/journal.pcbi.1004668
- https://en.wikipedia.org/wiki/Version_control
- https://en.wikipedia.org/wiki/Git
- https://git-scm.com/docs
- https://services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf
- http://github.com/ & http://gitlab.com
- https://gitlab.uliege.be/
- Git GUI: https://desktop.github.com/ & https://gitahead.github.io/gitahead.com/
- https://www.campus.uliege.be/cms/c_9096862/fr/services-internet-intranet-offerts

Finally

"Programming is like pinball.
The reward for doing it is
the opportunity of doing it again."

– Unknown

LIÈGE université
GIGA

Thank you for your attention!


LIÈGE université
GIGA