

MAX PLANCK INSTITUTE FOR HUMAN
COGNITIVE AND BRAIN SCIENCES
Department of Neurophysics
Department of Neurology



DOCUMENTATION

JIST-pipeline for automated ultra-high resolution MRI image co-registration and segmentation at 7 T

created: 10.08.2015

last updated: 12.07.2018

Author: Daniel Sung-Hwan Hänel

E-mail address: haenelt@cbs.mpg.de

Updated by: Juliane Döhler

E-mail address: doehler@cbs.mpg.de

CONTENTS

1	Co-registration of structural and functional data	3
1.1	About	3
1.2	Software	4
1.3	Content of the folder Coregistration_BOLD	4
1.4	Getting Started	5
1.4.1	On your private work station...	5
1.4.2	On an institute's work station...	6
1.4.3	Further Steps	6
1.5	Considered pulse sequences	8
1.6	Single steps of the pipeline	9
1.6.1	Input data	9
1.6.2	Scanner space	10
1.6.3	Pre-processing of structural data	11
1.6.4	Pre-processing of functional data	13
1.6.5	Co-registration	13
2	Co-registration of structural data	15
2.1	About	15
2.2	Software	16
2.3	Special requirements to perform non-linear registration	16
	References	16

1 Co-registration of structural and functional data

1.1 About

The understanding of the relationship between structure and function in the brain using various imaging methods is one of the main interests in neuroscience. Thereby, co- registration, i.e. the alignment of functional and structural images, tends to be a crucial step in processing imaging data. State-of-the-art imaging techniques at 7 T can provide full brain in vivo data with enhanced signal-to-noise ratio (SNR) at the isotropic resolution of 0.4 to 0.5 mm (Bazin et al., 2014). Such high resolution images require efficient computational methods that scale well with the increase in data size, handle the various contrasts, and identify the newly visible structure. However, conventional available computational methods of image processing like FreeSurfer, FSL or SPM do not provide the right method to handle high resolution imaging data in some cases.

In the following, we present a fully automated routine to co-registrate ultra-high resolution fMRI data at 7 T using the Java Image Science Toolkit (JIST). The aim is to get a global alignment that does not exceed a matching error of one voxel (perfect align-ment is not realistic over the whole brain). The presented co-registration method has two major registration steps. First, we transform structural and functional data to the same isocenter in scanner space. Second, with a nonlinear method using a diffeomorphic mapping (Avants, Epstein, Grossman, & Gee, 2008), structural images which show less distortions are targeted to the functional images for co-registration.

△ In this co-registration pipeline, the **structural images are transformed onto the functional images** and not the other way around.

First tests with in vivo whole brain data yielded a good alignment in all anatomical planes (axial, coronal, and sagittal¹) with an averaged running time around 20 min. Structural data were acquired with a Magnetization Prepared 2 Rapid Acquisition Gradient Echoes sequence (MP2RAGE) whereas the functional images with an echo planar imaging sequence (EPI) with whole brain resolution of 1.5 mm.

¹The sagittal plane was less sufficient, which is caused by difficulties in image aquirement of the 7 T MR device.

1.2 Software

The pipeline was created with the following packages, which are designed to process large data sets of high resolution images: MIPAV², JIST³, TOADS-CRUISE⁴ and CBS high-res brain processing tools packages which are released as a set of plug-ins for the MIPAV software package and the JIST pipeline environment⁵. See for installation *CBS Tools for MIPAV & JIST: Quick Installation and Processing Guide* (n.d.). JIST is a graphical pipeline engine, see for more information Lucas et al. (2010). To perform a series of computations, one simply lines up the desired modules, connects inputs and outputs and sets their parameters. Each separate data set to process can then be added to source modules, and the completed layout can then be saved and passed to the JIST Process Manager for actual processing. An example for power and ability especially in processing multimodality studies is given in Landman et al. (2013).

Input and output files were processed using the common research-oriented NIFTI file format (.nii), but various other formats are also supported in the MIPAV package. The JIST pipeline is given in the layout form (.LayoutXML).

1.3 Content of the folder Coregistration_BOLD

- ANTs-1.9.x-Linux
- antsenv
- Coregistration_BOLD.LayoutXML
- embedded_SyN.sh

We shortly describe the different files in the folder Coregistration_BOLD. ANTs-1.9.x-Linux contains the project Advanced Normalization Tools (ANTs)⁶ Please note that it is recommended to use this version of ANTs as newer releases might cause compatibility problems.

△ That project is necessary for correct working of the JIST module *embedded_SyN* which has to have access to ANTs.

²Medical Image Processing Analysis & Visualization: MIPAV release 5.4.2

³Java Image Science Toolkit: JIST 2011

⁴Topologically consistent tissue classification algorithm-Cortical Reconstruction Using Implicit Surface Evolution: TOADS-CRUISE release R3c (23 march 2012) a usual plug-in for JIST which was not used in this pipeline.

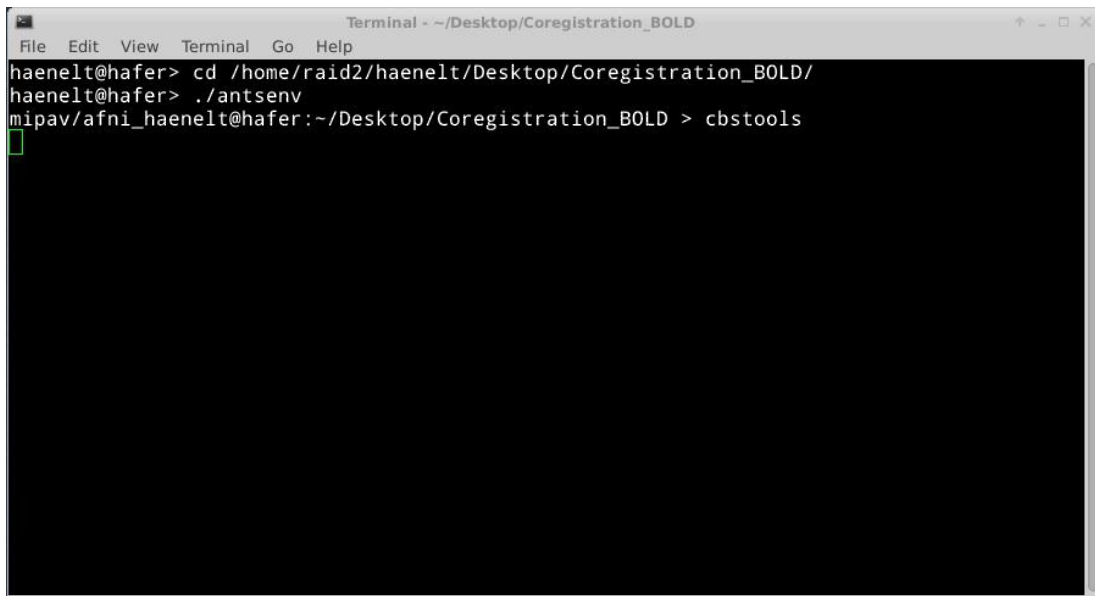
⁵See for more information <https://www.nitrc.org/projects/cbs-tools/> and <http://www.cbs.mpg.de/institute/software/cbs-hrt/index.html>

⁶The open-source project ANTs can be downloaded from [https://sourceforge.net/projects/advants/files/ANTs/ANTs 1.9.x/](https://sourceforge.net/projects/advants/files/ANTs/ANTs%201.9.x/). For tool description see <http://stnava.github.io/ANTs/>.

Also the script file *embedded_SyN.sh* is necessary for the correct working of that module. *antsenv* is a Bash startup script that enables JIST to get access to the ANTs project. The Layout file *Coregistration_BOLD.LayoutXML* contains the whole JIST pipeline for co-registration.

1.4 Getting Started

1.4.1 On your private work station.... We consider now the opening and running of the pipeline. After correct installation of the required software packages (MIPAV, CBS Tools, JIST) the input files should be converted into NIFTIs (*.nii*). In my example I saved the folder *Coregistration_BOLD* on the desktop of my home directory. In Fig. 1, I show just the way to open MIPAV. You have to go first to the location where you saved the folder *Coregistration_BOLD*. Then, you have to execute the script *antsenv*. Afterwards, you can open MIPAV, or more specific the *cbstools* environment, with the command *cbstools*.

A screenshot of a terminal window titled "Terminal - ~/Desktop/Coregistration_BOLD". The terminal shows the following commands and output:

```
haenelt@hafer> cd /home/raid2/haenelt/Desktop/Coregistration_BOLD/  
haenelt@hafer> ./antsenv  
mipav/afni_haenelt@hafer:~/Desktop/Coregistration_BOLD > cbstools
```

 A green cursor is visible on the line following the *cbstools* command.

Figure 1. Screenshot of the terminal which shows the procedure to open MIPAV.

⚠ That way of opening *cbstools* implies that all instruction must be performed in the same terminal.

But there are two things you have to pay attention carefully because the both scripts are saved locally. In the startup script *antsenv* the path of the ANTs project is stated. If the Folder *Coregistration_BOLD* is moved the new path has to be edited there as well. Also the path of the script *embedded_SyN.sh* has to be adjusted. That has to be changed directly in the JIST module *embedded_SyN* in the JIST Layout Tool. We come to that in a minute.

1.4.2 On an institute's work station.... You have to go first to the location where you saved the folder Coregistration_BOLD. Then, you have to execute the script antsenv (see prior section). Afterwards, in the same terminal or bash go to /afs/cbs.mpg.de/software/cbstools/3.0/ and run cbstools.

Running MIPAV for the first time? Then simply follow the instructions given here: <https://cbswiki.cbs.mpg.de/bin/view/Npsy/InitializeCBSToolsMIPAV>. Please be sure to set the JIST library directory correctly under the Global Preferences within the JIST Layout Tool after installing that plugin. To do so, update the "External Module Location" to the following path:

/afs/cbs.mpg.de/software/cbstools/3.0/module-library

You may need to close and restart MIPAV to see all available algorithms in the module viewer on the left side of the Layout Tool (see Fig. 2).

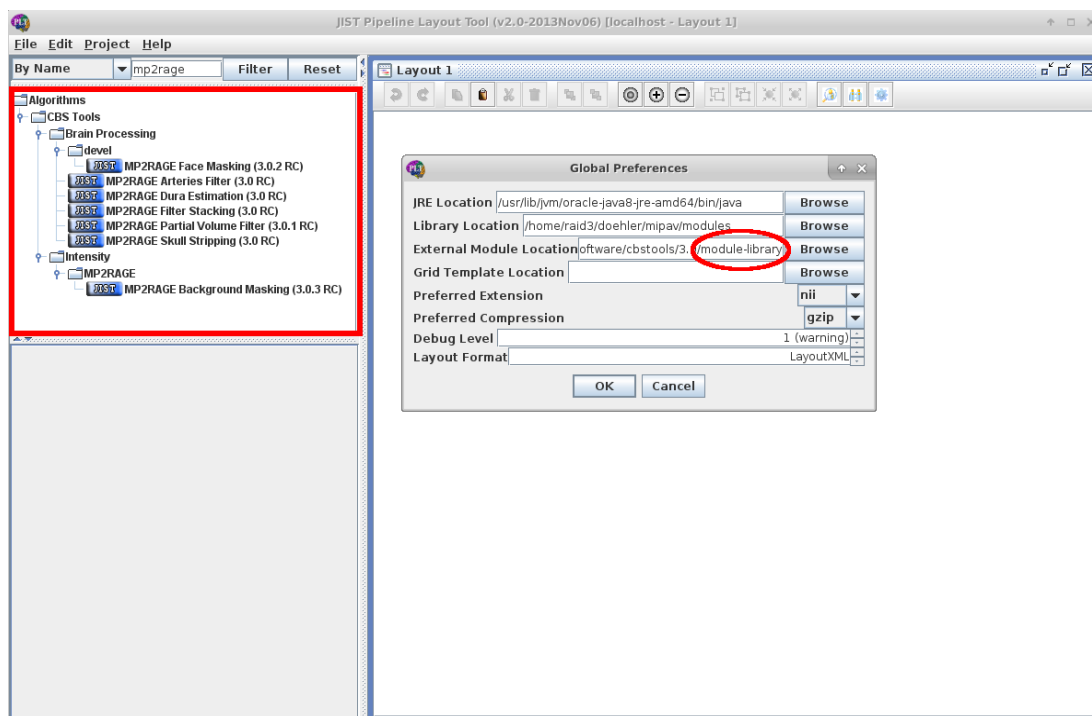


Figure 2. Available algorithms when the JIST library directory was set correctly.

1.4.3 Further Steps. With MIPAV you can open now the JIST Layout Tool if JIST is correctly installed as plug-in. On the menubar you go to Plugins → JIST → JISTLayoutTool. Then, a window like Fig. 3 should have opened.

You can open now the JIST pipeline Coregistration_BOLD.LayoutXML on the menubar with file → open. Then you open the pipeline which is located in the folder Coregistration_BOLD.

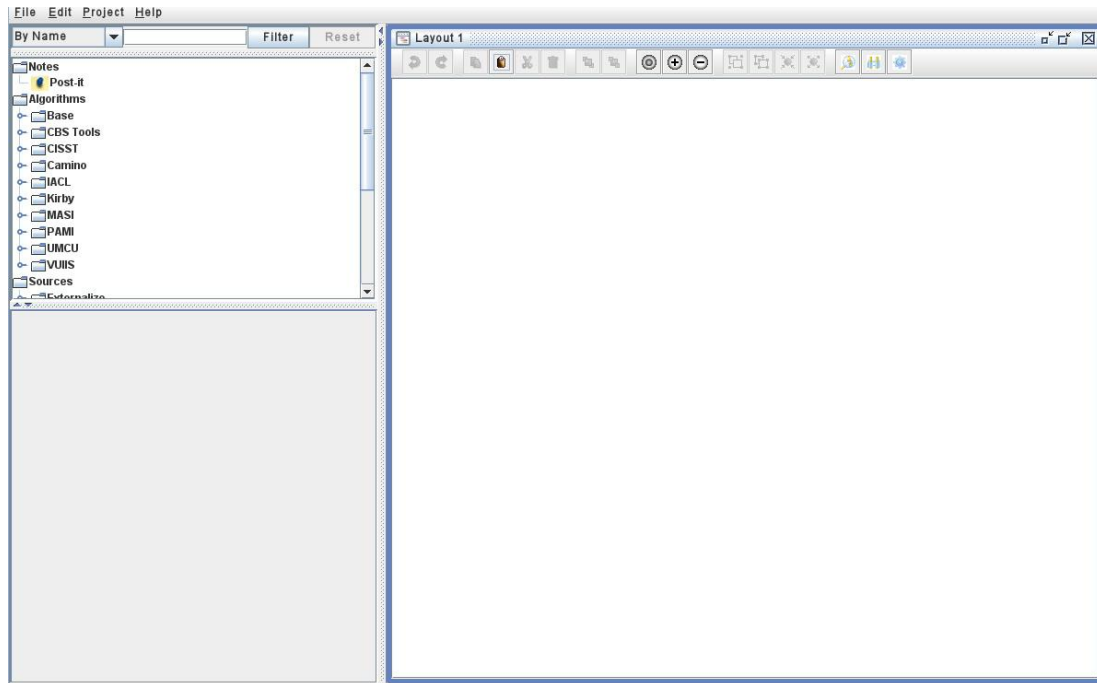


Figure 3. JIST Layout Tool with an empty layout file.

In Fig. 4 we show the opened pipeline in the JIST Layout Tool. With the upper blue modules you can select the processing data just by clicking on it and picking out the NIFTIs which shall be co-registered. But single modules will be explained explicitly in the last chapter.

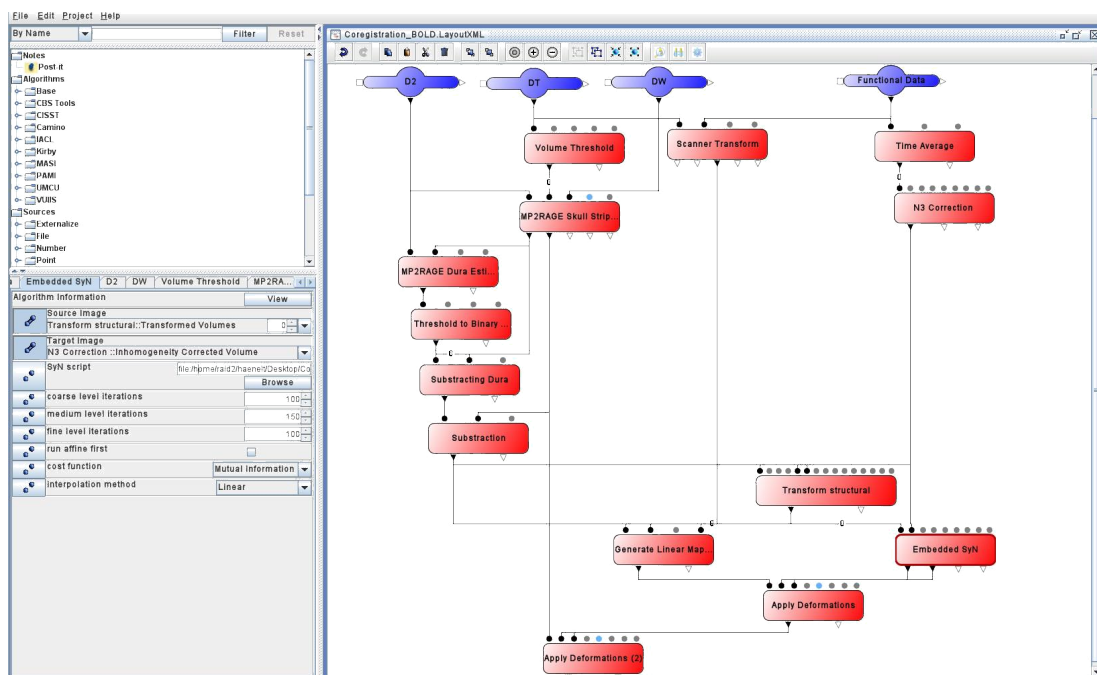


Figure 4. JIST Layout Tool with the opened JIST pipeline for co-registration.

We want just to add that you have to adjust the correct path for embedded_SyN.sh in the JIST Layout Tool. So, you have to click on the module Embedded_SyN

and all parameters of that function will pop up on the left. You find there the option SyN script where you have to add the correct path of the script file.

In the menubar you can further choose the desired output directory under Project → Layout Preferences. So, if you do not want to change any parameters of the JIST pipeline you only have to add the input files and select the output directory in the JIST Layout Tool (we will come to a detailed explanation of the single processing steps in chapter 6).

Now, we want to run the pipeline. Therefore, we have to open the JIST Process Manager. You have to select Plugins in the MIPAV menubar and go to JIST → JISTProcess-Manager. Then, you have to open the pipeline by selecting file → open. If you saved the selected input data in the JIST Layout Tool the chosen data is already considered in the Process Manager. To run the pipeline you have to choose on the menubar Scheduler → Start Scheduler. If you do not get any error messages, the pipeline produces output files for each computational step and save it to your desired output directory.

1.5 Considered pulse sequences

Structural data were acquired with the MP2RAGE sequence, see for more information Marques, Kober, Zwaag, Kruegger, and Gruetter (2010). Large spatial inhomogeneities in transmit B_1^+ and receive B_1^- fields observable in human MR images at high static magnetic fields (B_0) greater than 3 T impair the image quality. MP2RAGE overcomes this effect in brain T_1 -weighted images. It is a modification of the MPRAGE sequence generating two different images at different inversion times D_1 and D_2 , i.e. two gradient echo blocks instead of one. By combining the two images in the complex domain stated in Eq. 1, it is possible to create T_1 -weighted images where the result image is free of proton density contrast, T_2^* contrast, reception bias field, and to first order, transmit field inhomogeneity (Bazin et al., 2014).

$$D_W = \frac{D_1 D_2}{D_1^2 + D_2^2} \quad (1)$$

The resulting image is normalized in $[-0.5, 0.5]$. In total, we get different images according to different inversion times with that pulse sequence, namely T_1 -weighted image (D_W), first inversion (D_1), second inversion (D_2) which can be considered as proton-density-weighted image, and T_1 -map (D_T).

The functional image was acquired with an EPI pulse sequence resulting in a functional 4D image.

1.6 Single steps of the pipeline

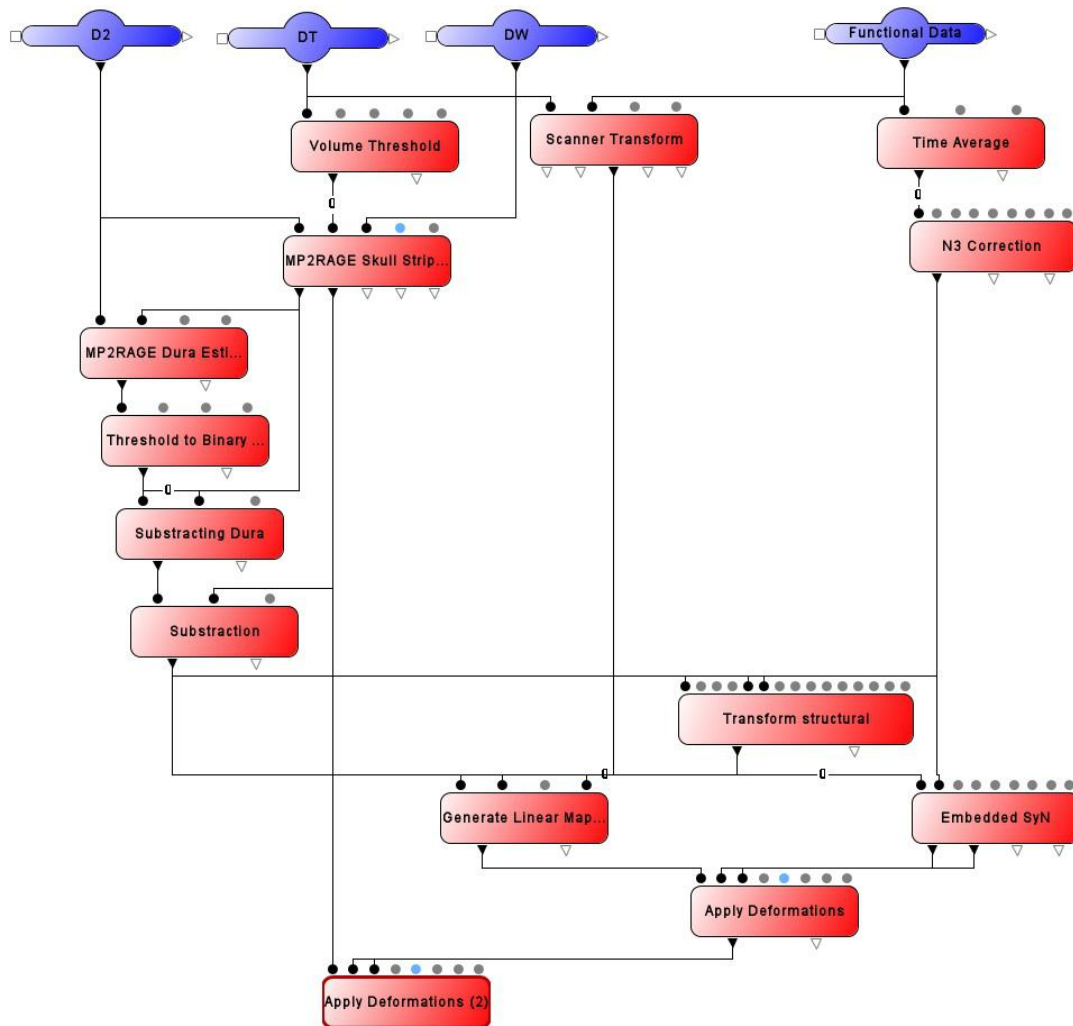


Figure 5. JIST pipeline for co-registration.

In Fig. 5 the whole JIST pipeline for co-registration of multimodal brain images at ultra-high fields is presented. For an explicit explanation of the pipeline we will go step by step through the pipeline and explain each module separately. To exemplify the computations we present real output data of one subject for each computational step.

1.6.1 Input data. The input data can be selected considering the *blue modules* in the first row. From left to right we have to include three structural images and one functional image (see Fig. 6).

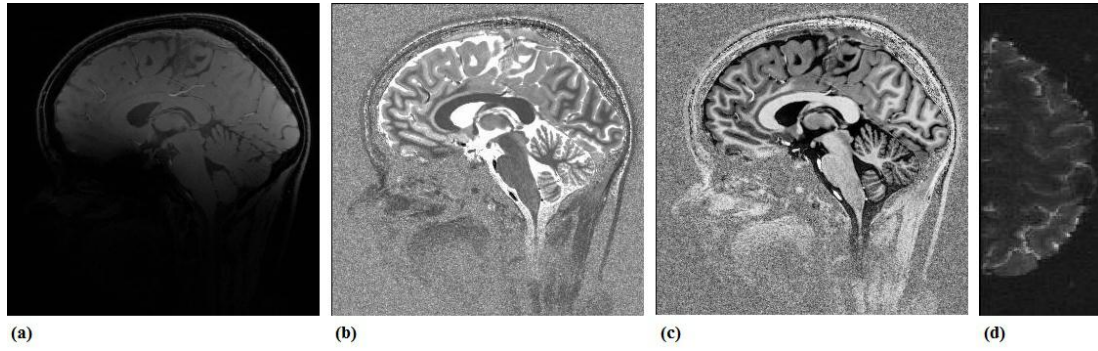


Figure 6. (a)-(c) One slide of the structural image in the sagittal plane of one subject. (a) Second inversion, (b) T_1 -map, and (c) T_1 -weighted image. (d) One slide of the functional image in the axial plane.

The structural images are different images from the MP2RAGE sequence. We used the following abbreviation D_2 (Second inversion), D_T (T_1 map) and D_W (T_1 -weighted map). The functional data comes from the EPI sequence. Select the input data by adding volumes to the activated module (to do so use the specification type box of the JIST Layout Tool on the left side of the window as it is shown in Fig. 7.).

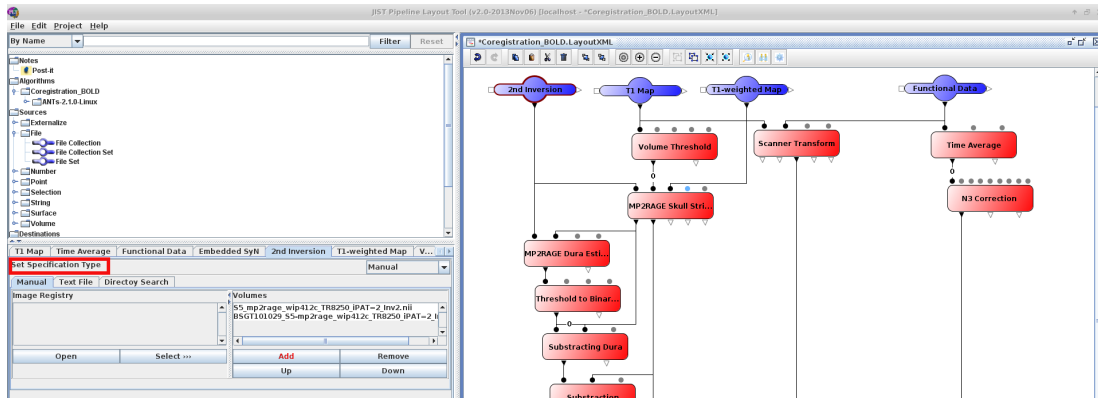


Figure 7. Selecting input data by adding new volumes to the activated modules via the JIST Layout Tool.

We exemplify the functioning of the pipeline with in vivo data of one subject. Structural whole brain data were acquired with MP2RAGE. Functional data of the right brain hemisphere were acquired with EPI which will be our target image.

1.6.2 Scanner space. The first and crucial step in the pipeline is the transformation of the image data of the T_1 map and the functional data in the same coordinate space, namely the scanner space of the MR device. We use the T_1 -map as source, targeting on the functional image. The module *Scanner Transform* generates an aligned source, transforming both images from subject space to the scanner space. It is important that this module has to be applied before any further transformations (rotation, translation, etc.) which would destroy information about the initial scanner space. For correct functioning no check marks should be made at the

parameters Ignore translations and Output in scanner space. Then, both images are brought to the same isocenter.

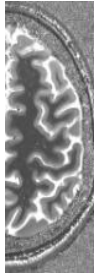


Figure 8. Slice of aligned T_1 -map with functional image.

In Fig. 8 the aligned source is shown. If one compares the aligned source with the slide of the functional image illustrated in Fig. 6 (d) already a good alignment can be observed. That is our first co-registration step. But the matching has to be improved. So, we have to pre-process further the structural and functional images separately. The transformation we got from this step gets important again if we apply the whole transformation on the structural data set.

1.6.3 Pre-processing of structural data. We now discuss the single modules to pre-process the structural images. We set the module Volume Threshold as filter function to get rid of outliers. That module has a lower threshold 1.0 and an upper threshold 4000.0. Thus, this function just filters out all binary zeros from the image.

The next step consists in skull stripping (brain extraction). This task is crucial, as any structure accidentally removed cannot be recovered. Therefore, we used the module *MP2RAGE Skull Stripping*. All three structural images are taken as input data. That function has three images as output which is illustrated in Fig. 9.

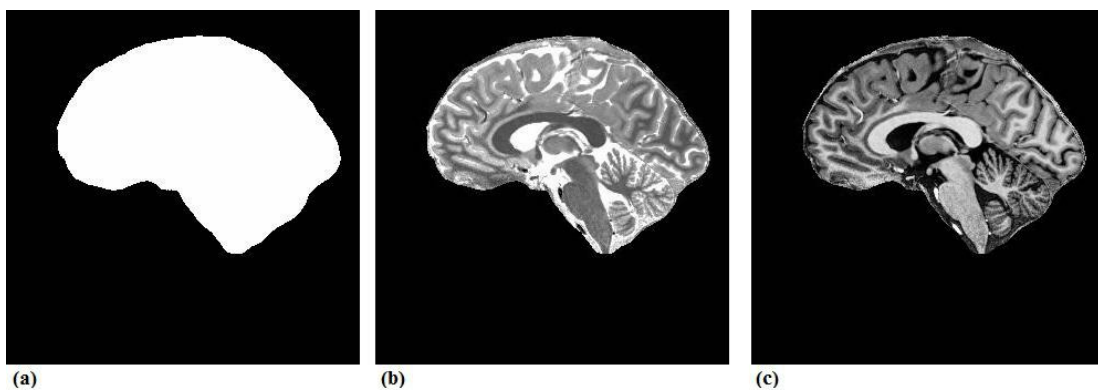


Figure 9. Brain extraction of structural images. (a) Brain mask image, (b) masked T_1 map image, and (c) masked T_1 -weighted image.

In that pipeline we only use the brain mask and the masked T_1 image. But in different tests we revealed that the co-registration can be enhanced drastically by

considering a dura estimation. It is known that at 7 T, the improved resolution makes the dura more visible (Bazin et al., 2014). The problem is that in the processing of brain extraction the dura is not removed sharply enough. Therefore, in the nonlinear co-registration step the structural images were shrunk to fit to the functional target which yielded in a bigger matching error. So, we estimate the dura with the module *MP2RAGE Dura Estimation*. Input data are the second inversion D2 and the brain mask image. In Fig. 10 (a) the estimated dura is shown. With the module *Threshold to Binary* a binary image of the estimated dura is calculated which is shown in Fig. 10 (b). That has to be done to merge the dura with the brain mask image. The binary threshold was set manually to the lower threshold of 0.8. The upper threshold is set as maximum value of the range. We also tested lower thresholds which resulted in a broader dura. But that estimate is too generously and already cut important parts of the cortex. We found 0.8 as reasonable but with other data sets this value has probably to be adjusted.

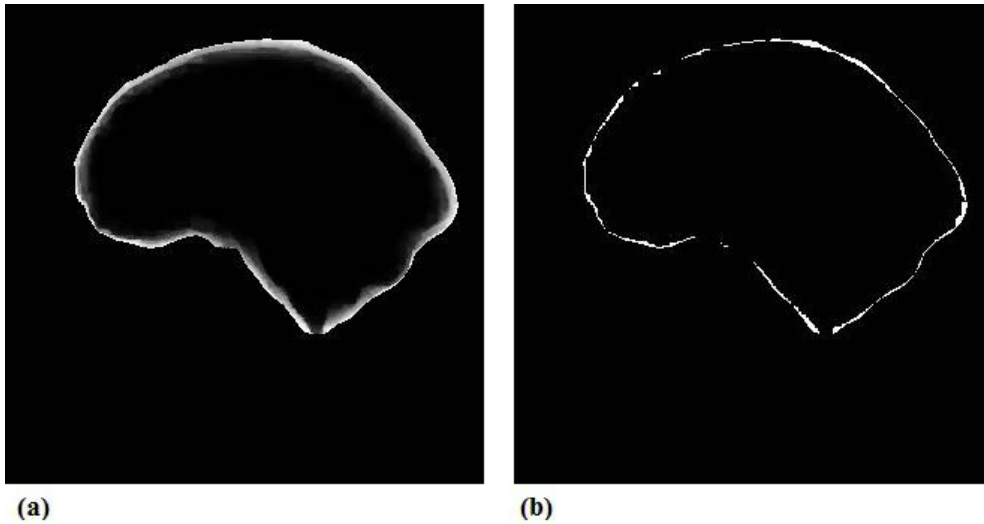


Figure 10. (a) Dura estimation to get a more accurate brain mask. (b) Binary threshold of the dura.

The next step consists in subtracting the dura estimate from the examined brain mask image which is done with the module *Subtract Dura*. That gives us a brain mask image without the estimated dura illustrated in Fig. 11 (a). Then we want to get rid of the dura in our T_1 map image. Therefore, we multiply the resolved brain mask image with our T_1 map using the module *Subtraction*. That cuts out all voxels which were determined in our binary image of the dura estimation. The result is shown in Fig. 11 (b).

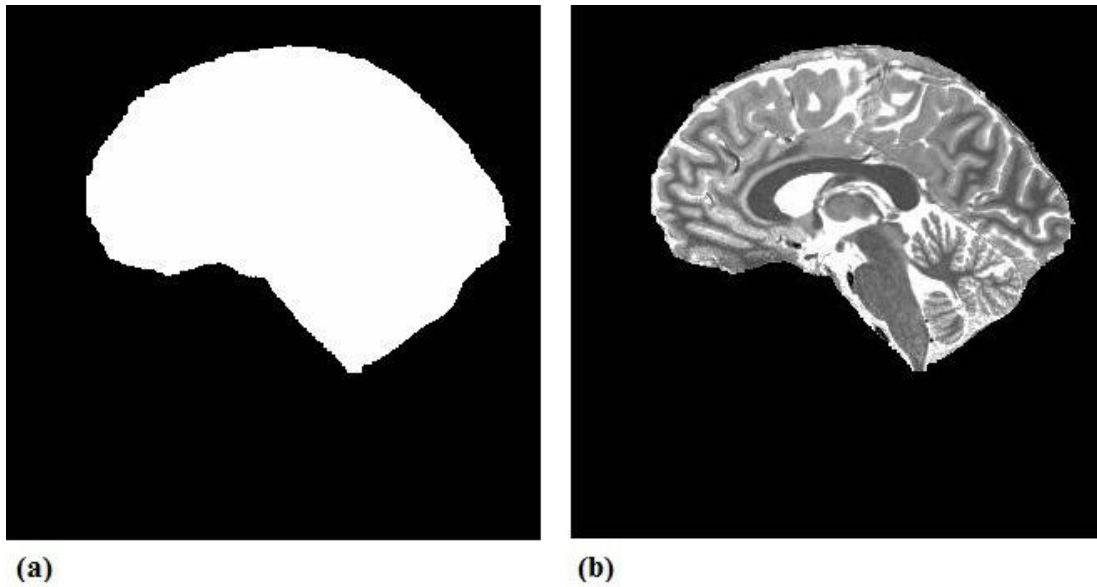


Figure 11. (a) Brain mask image without dura estimate. (b) T_1 map image without dura estimate.

This structural T_1 -map image is free of skull and dura and thus ready to be registered to the functional data.

1.6.4 Pre-processing of functional data. We turn now our concern to the right column of the pipeline, namely the pre-processing of the functional data. The functional data set is a 4D image, three dimension in space and one dimension in time. First, we want to get rid of the time dependency. So, with the module *Time Average* we average over the whole time line to get an averaged functional 3D image in space. The module *N3 Correction* executes a shading correction and get rid of B-field inhomogeneities.

1.6.5 Co-registration. We now come to the final co-registration steps. The module *Transform Structural* applies a given transformation matrix to a volumetric image. The transformation matrix in this case comes from the first source transformation into the scanner space. The target image is the N3-corrected functional image and we use the pre-processed T_1 -map image as source volume. We use a window-sinc function as interpolation function which shall improve both robustness and noise for grey and white matter (Bazin et al., 2014). As output we get the transformed volume which is illustrated in Fig. 12 (a).

Because we want to add now a nonlinear transformation, we have to be a little bit more careful. To be able to merge both transformation matrices (one from scanner transform and the other from embedded SyN) we have to generate a linear mapping for the transformed volume. So, we use the module *Generate Linear Mapping* and use the pre-processed T_1 -map image as source image again. But this time the transformed source volume is taken as reference image. We also include

the transformation matrix from the first transformation to scanner space but that is optional. That module generates now a coordinate mapping which allows us to apply this transformation together with the nonlinear transformation which will be explained now.

For the nonlinear transformation we use the module *Embedded SyN* (Avants et al., 2008) which consists in a deformable image registration using a symmetric diffeomorphic⁷ algorithm with the development of a symmetric image normalization method (SyN). That technique is able to deal with both small and large deformation problems. Here, the registration is restricted on diffeomorphic space with homogeneous boundary conditions, i.e. the assumption that rigid and scaling transformations have been factored out and image borders maps to itself. The final registration is calculated by symmetric normalization of both images, i.e. the algorithm searches for a finding correspondance with equal consideration of both images.

That function uses an intensity based approach using normalized mutual information as similarity metric, which has shown to be advantageous in registration across different modality (Landman et al., 2013; McAullife, 2008). The number of iterations which is set in our case to 100 (coarse level iterations) do not yield a better result if we increase the number because the function already exits the loop if a good alignment is reached. So, more iteration neither yields better nor worse results. But in other cases the number of iteration may be adjusted.

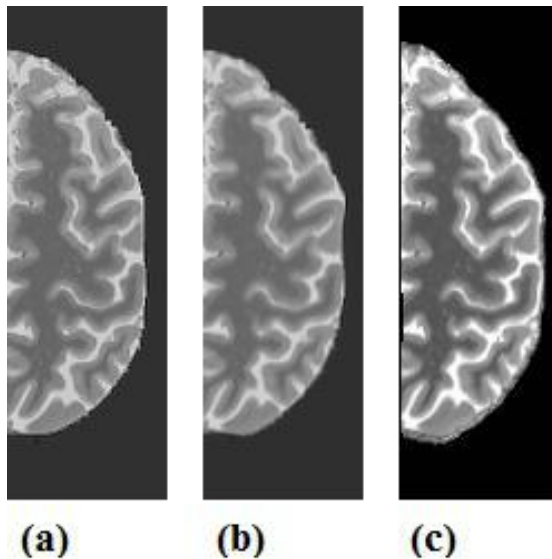


Figure 12. (a) Structural image after transformation to the same icocenter, (b) structural image after nonlinear transformation, and (c) both transformations together.

Fig. 12 (b) shows the nonlinear transformation on the structural image. A difference to Fig. 12 (a) is difficult to notice but crucial in our pipeline. The final

⁷A diffeomorphism is a differentiable map with a differentiable inverse.

step consists in putting both transformations together. In the module *Apply Deformations* we put in both transformation matrices. That module contains the whole transformation which can be applied to any source volume.

To exemplify the result we apply that transformation to our initial T_1 -map image (with dura!) using the module *Apply Deformations (2)* which is illustrated in Fig. 12 (c).

I hope that this little description was helpful and enjoyable. But more important, I only tested this pipeline on a few data sets where it worked pretty well. So, I hope that this remains true in other cases and that this pipeline simplify the co-registration of functional data at ultra-high resolution images a little bit.

2 Co-registration of structural data

2.1 About

The understanding of microstructural relations in the brain that underpin certain behaviors is one of the main foci in neuroscience. Thereby, co- registration of structural images, i.e. the alignment of slab and whole brain images of different resolutions, tends to be a crucial step in processing imaging data. State-of-the-art imaging techniques at 7 T can provide brain in vivo data with enhanced signal-to-noise ratio (SNR) at the isotropic resolution of 0.4 to 0.5 mm (*CBS Tools for MIPAV & JIST: Quick Installation and Processing Guide*, n.d.). Such high resolution images require efficient computational methods that scale well with the increase in data size, handle the various contrasts, and identify the newly visible structure. However, conventional available computational methods of image processing like FreeSurfer, FSL or SPM do not provide the right method to handle high resolution imaging data in some cases.

In the following, we present a fully automated routine to co-registrate ultra-high resolution MRI data at 7 T using the Java Image Science Toolkit (JIST). The aim is to get a global alignment that does not exceed a matching error of one voxel (perfect alignment is not realistic over the whole brain). The presented co-registration method has two major registration steps. First, we transform the slab and whole brain structural data to the same isocenter in scanner space, and perform a linear (rigid) registration process that roughly aligns our two images of interest (e.g., slab and whole brain). Second, with a nonlinear method using a diffeomorphic mapping [2], the slab image is precisely targeted to the resampled whole brain image.

⚠ In this co-registration pipeline, the **slab images are transformed onto the resampled whole brain images** and not the other way around.

First tests with in-vivo whole brain data yielded a good alignment in all anatomical planes (axial, coronal, and sagittal⁸) with an averaged running time around 20 min. Structural brain data were acquired with a Magnetization Prepared 2 Rapid Acquisition Gradient Echoes sequence (MP2RAGE) with a isotropic resolution of 0.7 mm for the whole brain image and a isotropic resolution of 0.5 mm for the slab image.

2.2 Software

The pipeline was created with the following packages, which are designed to process large data sets of high resolution images: MIPAV⁹, JIST¹⁰, TOADS-CRUISE¹¹ and CBS high-res brain processing tools packages which are released as a set of plug-ins for the MIPAV software package and the JIST pipeline environment¹². See for installation [3]. JIST is a graphical pipeline engine, see for more information [4]. To perform a series of computations, one simply lines up the desired modules, connects inputs and outputs and sets their parameters. Each separate data set to process can then be added to source modules, and the completed layout can then be saved and passed to the JIST Process Manager for actual processing. An example for power and ability especially in processing multimodality studies is given in [5].

Input and output files were processed using the common research-oriented NIFTI file format (.nii), but various other formats are also supported in the MIPAV package. The JIST pipeline is given in the layout form (.LayoutXML).

2.3 Special requirements to perform non-linear registration

- ANTs-1.9.x-Linux
- antsenv
- embedded_SyN.sh

Here, we shortly describe the different files. ANTs-1.9.x-Linux contains the project6 Advanced Normalization Tools (ANTs).

△ That project is necessary for correct working of the JIST module embedded_SyN which has to have access to ANTs.

⁸The sagittal plane was less sufficient, which is caused by difficulties in image aquirement of the 7 T MR device.

⁹Medical Image Processing Analysis & Visualization: MIPAV release 5.4.2

¹⁰Java Image Science Toolkit: JIST 2011

¹¹Topologically consistent tissue classification algorithm-Cortical Reconstruction Using Implicit Surface Evolution: TOADS-CRUISE release R3c (23 march 2012) a usual plug-in for JIST which was not used in this pipeline.

¹²See for more information <https://www.nitrc.org/projects/cbs-tools/> and <http://www.cbs.mpg.de/institute/software/cbs-hrt/index.html>

References

- Avants, B. B., Epstein, C. L., Grossman, M., & Gee, J. C. (2008). Symmetric diffeomorphic image registration with cross-correlation: evaluating automated labeling of elderly and neurodegenerative brain. *Medical image analysis*, 12(1), 26–41. doi:10.1016/j.media.2007.06.004
- Bazin, P.-L., Weiss, M., Dinse, J., Schäfer, A., Trampel, R., & Turner, R. (2014). A computational framework for ultra-high resolution cortical segmentation at 7 tesla. *NeuroImage*, 93, Part 2, 201–209. In-vivo Brodmann Mapping of the Human Brain. doi:10.1016/j.neuroimage.2013.03.077
- CBS Tools for MIPAV & JIST: Quick Installation and Processing Guide. (n.d.). Retrieved from <https://www.cbs.mpg.de/169392/cbs-tools-userguide.pdf>
- Landman, B. A., Bogovic, J. A., Carass, A., Chen, M., Roy, S., Shiee, N., ... Prince, J. L. (2013). System for integrated neuroimaging analysis and processing of structure. *Neuroinformatics*, 11(1), 91–103. doi:10.1007/s12021-012-9159-9
- Lucas, B. C., Bogovic, J. A., Carass, A., Bazin, P.-L., Prince, J. L., Pham, D., & Landman, B. A. (2010). The Java Image Science Toolkit (JIST) for Rapid Prototyping and Publishing of Neuroimaging Software. *Neuroinformatics*, 8(1), 5–17. doi:10.1007/s12021-009-9061-2
- Marques, J. P., Kober, T., Zwaag, v. d. W., Kruegger, G., & Gruetter, R. (2010). MP2RAGE, a self-biased corrected sequence for improved segmentation and T 1 -mapping at high field. *NeuroImage*, 49, 1271–1281. doi:10.1016/j.neuroimage.2009.10.002
- McAullife, M. (2008). *MIPAV: Medical Image Processing Analysis & Visualization. User's Guide*. Rockville, Maryland. Retrieved from <https://mipav.cit.nih.gov/documentation/userguide/Vol2Algorithms.pdf>