frontiers in Neuroinformatics

TECHNOLOGY AND CODE
published: 11 February 2020
doi: 10.3389/fninf.2020.00005

An Automated Open-Source Workflow for Standards-Compliant Integration of Small Animal Magnetic Resonance Imaging Data

Horea-Ioan Ioanas[1]*, *Markus Marks*[2], *Clément M. Garin*[3], *Marc Dhenain*[3], *Mehmet Fatih Yanik*[2] *and Markus Rudin*[1]

[1] *Institute for Biomedical Engineering, ETH and University of Zurich, Zurich, Switzerland,* [2] *Institute of Neuroinformatics, ETH and University of Zurich, Zurich, Switzerland,* [3] *Centre National de la Recherche Scientifique (CNRS), Université Paris-Sud, Paris, France*

Large-scale research integration is contingent on seamless access to data in standardized formats. Standards enable researchers to understand external experiment structures, pool results, and apply homogeneous preprocessing and analysis workflows. Particularly, they facilitate these features without the need for numerous potentially confounding compatibility add-ons. In small animal magnetic resonance imaging, an overwhelming proportion of data is acquired via the ParaVision software of the Bruker Corporation. The original data structure is predominantly transparent, but fundamentally incompatible with modern pipelines. Additionally, it sources metadata from free-field operator input, which diverges strongly between laboratories and researchers. In this article we present an open-source workflow which automatically converts and reposits data from the ParaVision structure into the widely supported and openly documented Brain Imaging Data Structure (BIDS). Complementing this workflow we also present operator guidelines for appropriate ParaVision data input, and a programmatic walk-through detailing how preexisting scans with uninterpretable metadata records can easily be made compliant after the acquisition.

Keywords: MRI/fMRI, small animal imaging, repositing, FOSS, Bruker, ParaVision, Python

OPEN ACCESS

Edited by:
Jan G. Bjaalie,
University of Oslo, Norway

Reviewed by:
Reza M. Salek,
International Agency For Research On Cancer (IARC), France
Jason Lerch,
Toronto Centre for Phenogenomics, Canada
Remi Gau,
Catholic University of Louvain, Belgium

***Correspondence:**
Horea-Ioan Ioanas
ioanas@biomed.ee.ethz.ch;
chr@chymera.eu

Received: *31 May 2019*
Accepted: *16 January 2020*
Published: *11 February 2020*

Citation:
Ioanas H-I, Marks M, Garin CM, Dhenain M, Yanik MF and Rudin M (2020) An Automated Open-Source Workflow for Standards-Compliant Integration of Small Animal Magnetic Resonance Imaging Data. Front. Neuroinform. 14:5. doi: 10.3389/fninf.2020.00005

1. INTRODUCTION

Magnetic resonance imaging (MRI), and functional MRI (fMRI) are highly popular methods in the field of neuroscience. Their high tissue penetration makes them eminently suited for reporting features at the whole-brain level *in vivo*. High assay coverage is particularly relevant for an organ as holistic in its function as the brain, as it facilitates the interrogation of not only sensitivity but also regional specificity. However, MRI methods generate signal via nuclear spin polarization—which is commonly very weak—and characteristically posses low intrinsic sensitivity. Additionally, fMRI methods rely on highly indirect measures of neuronal activity, and are consequently susceptible to numerous confounding factors.

In animal fMRI in particular, subject preparation, and more specifically cerebrovascular parameters (Schroeter et al., 2016) and anesthesia (Schlegel et al., 2015; Bukhari et al., 2018) are widely known drivers of result variability. In order to integrate data which may be thus strongly

confounded—as well as in order to clarify the confounds themselves (Grandjean et al., 2019)—it is vital that data is shared in a raw state, i.e., having undergone no or as little processing as possible. Raw data sharing increases transparency and reproducibility, as data can be assumed to be free from undocumented "fixes." Such attempts at *ex post facto* data improvement may not just include data matrix manipulations, but also outlier (subject or session) filtering. While valid rationales for both outlier filtering and data editing exist, these processes are best performed in a transparent and well-documented fashion, leaving the raw data untouched as an ultimate recourse.

As published data is intended for reuse, it is reasonable to assume that it may be employed to explore hypotheses other than those under the constraints of which it was originally acquired. In such cases, it is vital that the shared entry and feature pool be as inclusive as possible. It is likely, above all in the effort of methodological comparison and improvement, that what is an artifact or outlier for the interrogation of a narrow hypothesis, may constitute a strong driver of the effect of another hypothesis. Therefore, it is the best choice for small animal MRI researchers to publish data in as raw a form as possible.

The documentation of directed information processing is known as data provenance, and in the effort of establishing a point of recourse it is helpful to map out data traces to the earliest record or earliest record in digital form. In **Figure 1**, a simplified summary of data provenance is showcased, based on the most common features of small animal MRI. While the rawest data is in theoretical terms the best possible recourse, the extent of this overview illustrates that the choice of a raw data origin point is also constrained by the scope of a researcher's work. Particularly the first step, reconstructing volumetric information from the time domain k-space record, is commonly not covered by modern MRI pipelines, and instead left to the original acquisition software.

Whether directly from the k-space file or from the reconstructed image, the data needs to be converted into a standard, vendor-independent form. Standards are a cornerstone of scientific collaboration, as they concomitantly enable result comparability and data integration. They can, however, also be potentially restrictive, since a standard may impose artificial limitations on features or artificial requirements not pertinent to a particular study. Such limitations may materialize in negative restrictions on software tools (e.g., proprietary standards may preclude data processing with non-proprietary tools), restrictions on hypotheses (e.g., data organized by one hierarchical principle may lead to loss or obfuscation of certain category correspondence relationships), or positive restrictions of technologies (e.g., data available in highly specific formats might require usage of and familiarity with highly specialized tools)—with the latter restriction being particularly relevant in database standards.

In order to access the benefits and mitigate the pitfalls of standards compliance, data should be migrated to a form which is openly and thoroughly documented, and easily accessible for user manipulation. In the field of small animal MRI, the vast majority acquisition devices (∼80 %) are produced by the Bruker Corporation, and thus the largest segment of data is initially formatted according to the ParaVision standard. This standard is largely transparent, with most metadata stored in plain-text files. The data, however, are stored in a binary format, which strongly diverges from the *de facto* standard of NIfTI (Cox et al., 2004), and for which extensive documentation is not openly available. Conversion tools from the ParaVision standard to NIfTI exist (Ferraris et al., 2017; Rorden and Naveaum, 2018), and have more recently also been made available from the manufacturer (presently only in closed-source form and only in ParaVision 360, without backward compatibility). Contingent on the scope limitations of the NIfTI format itself, these tools can however not repackage the majority of metadata represented in ParaVision standard plain-text files. This situation exemplifies how the utility of standards is not only contingent on their suitability for use as a common origin format, but also on their flexibility to accommodate all relevant information.

The Brain Imaging Data Structure (BIDS) standard (Gorgolewski et al., 2016), is a prominent candidate for repositing small animal magnetic resonance imaging data. It is thoroughly tested and well-adopted in the field of human MRI, and its extensible and permissive nature makes it easily adaptable to small animal data—as well as generally accommodating for broad swathes of eclectic use cases. The standard builds upon the NIfTI format (Cox et al., 2004), one of the most widely used formats for high-level neuroimaging analysis, which is compact, as it offers on-the-fly lossless compression, and general-purpose, as its header only contains the minimal amount of metadata required for spatiotemporal image representation. In addition, BIDS offers an extendable specification for metadata, stored in sidecar text files. This separation of minimal and full metadata

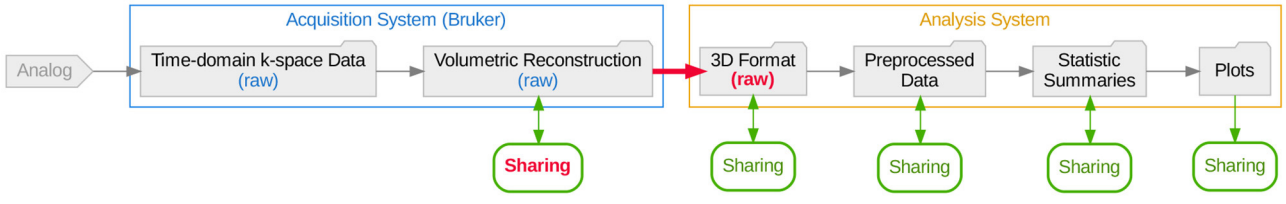

FIGURE 1 | Data provenance flowchart, with the leftmost data being the rawest. Folder nodes represent data states on disk, with nodes suitable as raw data recourse highlighted with the word raw in parentheses. The introduction of a standardized conversion process (red arrow edge) would permit the creation of a 3D format representation usable as raw data recourse, as well as the sharing of Bruker acquisition system volumetric reconstruction data (also highlighted in red).

space makes BIDS easily and incrementally accessible for new users and portable to further modalities, as the core requirements can rapidly be met and further relevant metadata fields can be processed and added as they become required for analysis. The standard's usage of plain-text metadata files also makes them accessible to ubiquitous, minimal, and Free and Open Source (FOSS) tools, e.g., the Standard GNU Utilities, or equivalent core utility implementations. Particularly, the representation of metadata as text allows data set versions with increments in metadata availability to be compared via `diff`-type commands (Hunt and MacIlroy, 1976), which is less feasible for binary data. The format is organized around a simple directory hierarchy, with key metadata fields captured in the file and path names. This makes BIDS data intuitive to access from both a console and a graphical user interface.

Given varying specifications, it is common for standards to not map fully onto each other (e.g., one metadata field may not have a clear correspondence relationship in both standards). As data conversion is always based only on the most recent parent format, this means that the risk of data (or more specifically, metadata) loss or obfuscation grows with each transition to a new standard. Thus, in the example of **Figure 1**, collaborative potential is best served if both the "3D Format" representation is infused with sufficient and sufficiently accessible metadata, *and* the original "Volumetric Reconstruction" is rendered shareable.

Both these goals can be attained by the introduction of an automated open-source workflow which can perform the standard transition. As such, all metadata fields which are identified as equivalent between the ParaVision and BIDS standards can be made accessible in the final form. Conversely, if ParaVision standard data is automatically interpretable as input for a concatenation of processing workflows, this original form can also serve as a shareable raw data recourse.

2. THE WORKFLOW

The workflow, entitled `bru2bids` (Bruker ParaVision to BIDS), is distributed as part of SAMRI (Ioanas et al., 2019b), a free and open source workflow package of the ETH and University of Zurich Institute for Biomedical Engineering. This repositing workflow can be used stand-alone, but also serves as a gateway to all the further workflows included in the SAMRI package (encompassing dedicated solutions for all analysis steps showcased in **Figure 1**). As such, the ParaVision-to-BIDS workflow not only permits users to convert data into a format which is more widely supported and flexible, but also easily links to reference implementations for BIDS-based small animal processing functionalities (e.g., registration; Ioanas et al., 2019c).

The workflow reposits data and metadata from the ParaVision standard into a BIDS-compliant form, notifying the user of BIDS validation possibilities upon completion. The repositing process automatically handles the conversion of data from ParaVision volumetric reconstruction files (`2dseq`) to NIfTI files. Additionally, it assigns metadata from the specific ParaVision text files to either the NIfTI header, the BIDS metadata files, or the BIDS directory hierarchy, as applicable. A simplified overview of this process is presented in **Figure 2A**. A more extensive break-down of metadata sourcing—showing the actual input and output files, and highlighting metadata fields represented in the data paths—is laid out in **Figure 2B**.

The repositing functionality described herein can be accessed from both Bash and Python, via `SAMRI bru2bids` or `samri.pipelines.reposit.bru2bids()`, respectively. Invocation variants are illustrated in **Figure 3**, and link to the same code implementation. The `bru2bids` function is highly parameterized, with the same parameter set available in either Bash or Python. A full list of parameters can be obtained by executing the `SAMRI bru2bids -help` command from the console. The current parameter listing for Bash is presented under **Figure S1**.

Notable parameters include the functional, diffusion-weighted, and structural scan specification. These three selectors each use an input dictionary (pairs of one *key*, which is a BIDS metadata string, and a list of accepted *values*), to identify which scans are to be reposited. Examples of such dictionaries are given in **Figure 3**, where scans with an "EPI" acquisition field are categorized as functional, and files with a "TurboRARE" acquisition field are categorized as structural. The repositing pipeline is run sequentially for all scan categories, as separate processes are required for each scan type (examples of the internal processing nodes are shown in **Figure 6**). Further parameters not included in the workflow to date, such as magnetic resonance spectroscopy (MRS) data selection, can easily be implemented, given the availability and familiarity with enough example data, by copying and editing the process instructions from present parameters inside the `samri.pipelines.reposit.bru2bids()` function.

2.1. Operator Guidelines

The process of scan categorization and metadata sourcing for BIDS conversion is contingent on the presence of operator input records interpretable by the workflow. As the ParaVision metadata files contain free-field input, adherence to a minimal set of guidelines is necessary to ensure unambiguous error-free conversion.

The acquisition of data in the Bruker ParaVision graphical user interface is commenced by creating a new study in the "Study Registration" window (**Figure S2**). In this window the operator should fill in the "Animal ID" entry corresponding to the intended BIDS subject identifier (i.e., the `sub` field in the resulting path and file names), and the "Study Name" entry corresponding to the intended BIDS session identifier (i.e., the `ses` field in the resulting path and file names). Notably, the values for both of these fields should be BIDS-compliant—meaning that they should contain only alphanumeric characters, necessarily excluding underscores and hyphens, which are used in the BIDS standard as field separators.

Once the study is created, scans should be renamed in the graphical user interface (before or after acquisition) to contain the additional relevant metadata information according to the BIDS standard. Thus, a resting-state scan acquired with an EPI sequence resolving BOLD contrast should contain the following string in the "Instruction Name" column of the

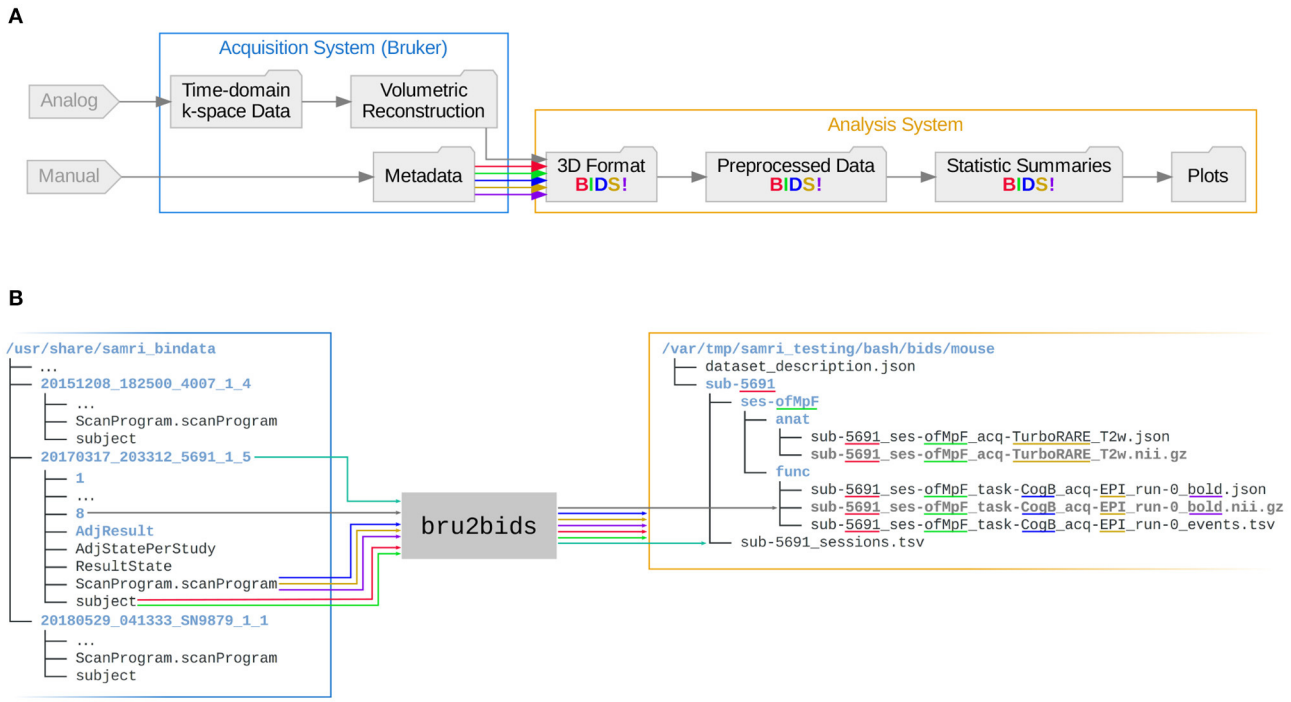

FIGURE 2 | The Paravision-to-BIDS repositing process automatically interprets the source data structure and determines corresponding variables in the BIDS standard. Depicted are color-coded overviews of data and metadata streams during the SAMRI `bru2bids` repositing process, with the data matrix content coded in gray, the subject field in red, the session field in green, the task field in blue, the acquisition field in ochre, and the modality suffix in purple. **(A)** Processing flowchart, breaking down the Bruker to BIDS repositing step (highlighted in **Figure 1**) to depict data-metadata integration, and downstream metadata encoding in the BIDS directory hierarchy and corresponding metadata files. **(B)** Directory tree overview of Bruker to BIDS repositing. Depicted are source and result directories, with arrows indicating which files the most relevant BIDS variable fields are sourced from. Date information is coded in cyan, and is sourced directly from the ParaVision scan directory name. The conversion presented herein shows the mouse data form the test data collection, and is performed given the instructions in **Figure 3A**: note that no BIDS entries from the 4007 and SN9879 subject directories were processed (as well as from any of the other species). This is happening by design and because no entries in any of the other scans match the selection criteria.

A

```
user@host ~ $ SAMRI bru2bids\
        -o /var/tmp/samri_testing/bash\
        -f '{"acquisition":["EPI"]}'\
        -s '{"acquisition":["TurboRARE"]}'\
        /usr/share/samri_bindata
```

B

```
from samri.pipelines.reposit import bru2bids
bru2bids('/usr/share/samri_bindata/',
        inflated_size=False,
        functional_match={'acquisition':['EPI']},
        structural_match={'acquisition':['TurboRARE']},
        out_base='/var/tmp/samri_testing/pytest/',
        )
```

FIGURE 3 | Both Bash and Python can be used to access the repositing functionality. As the Bash binding is auto-generated from the Python function, features become available synchronously, and inspection can be coherently performed regardless of the invocation language. Both code snippets specify the exact same instructions regarding the data source: they indicate that ParaVision standard data from `/usr/share/samri_bindata/` is to be reposited into a BIDS standard form, categorizing scans with an "EPI" acquisition string as functional, and files with a "TurboRARE" acquisition string as structural. Both of the above invocations are included in the package test suite. **(A)** Bash invocation, repositing data into a BIDS directory located under `/var/tmp/samri_testing/bash`. **(B)** Python invocation, repositing data into a BIDS directory located under `/var/tmp/samri_testing/pytest`.

ParaVision interface: `acq-seEPI_task-rest_bold`. The format is composed out of the BIDS short identifier (e.g., `acq` for acquisition), followed by a hyphen and the desired metadata field value (e.g., `seEPI` for spin-echo echo-planar imaging). The level of detail in these fields is at the discretion of the operator, as per the flexible nature of the standard. If recognizing the EPI variant at-a-glance is deemed irrelevant for the data at hand, this field may simply be assigned a value of `EPI`, or could conversely be expanded to include an arbitrary amount of additional detail. Pairs of BIDS short identifiers and desired values should be separated by underscores, with the modality suffix appended at the end after a final underscore separator.

Additional BIDS fields such as the run ordinal number (e.g., `run-0`, as seen in **Figure 2B**), are automatically determined by

the bru2bids workflow. The modality suffix, if not explicitly specified by the operator, can also be automatically assigned in a small number of cases where it is unambiguous (e.g., a scan with a FLASH acquisition but with no operator defined modality will be assigned a T1w modality suffix, and similarly, one with a TruboRARE acquisition will be assigned a T2w suffix).

Entering metadata in this fashion during scanner operation creates records which can directly serve as input for analysis workflows, and are thus immediately ready for sharing or analysis. Examples for such interpretable metadata scans are available in the SAMRI ParaVision testing data archive, and the example strings specified in this section are sourced from the rat data identified in the archive by the 20171106_184345_21_1_1 directory name.

2.2. Preexisting Data Compliance

Preexisting data sets—acquired with operator metadata input divergent from the recommended guidelines—can also be redressed, via a small number of plain text editing operations. Performing such edits is highly recommended, as it is still easier than manually repositing data in the BIDS standard, and also permits direct ParaVision data sharing, as shown in **Figure 1**.

The relevant parts of a ParaVision scan directory which need to be edited are single lines in the subject and Scanprogram.scanProgram files. Editing the subject identifier in the ParaVision directory name is also advisable, though only for ease of overview—since the subject field is not read from the directory name by the repositing workflow.

Before editing, value names for all available metadata fields must be chosen, such as respect the constraints of the BIDS standard. This means that subject, session, task, and all other desired identifiers need to contain only alphanumeric characters (digits 0 through 9, and lower and uppercase letters). Once all identifiers are properly chosen, they can be replaced or retroactively entered into the ParaVision metadata fields.

The subject file defines both the subject and the session of the scan. To enter the subject identifier, the line *below* the line containing the string ##$SUBJECT_id= must be edited. This following line should contain the subject identifier between greater than and less than characters, e.g., for a subject identified as Mc365A, the line should read <Mc365A>. Analogously, to enter the session identifier, the line *below* the line containing the string ##$SUBJECT_study_name= needs to be edited to read e.g., <R02> for a session identified as R02 within the study, as shown in **Figure 4B**.

The Scanprogram.scanprogram file is used to define other BIDS metadata fields. Within this file, individual lines containing the string ␣(E (with the open box character representing a space) are used both to record the ParaVision "Instruction Name" and to establish correspondence with the respective numbered ParaVision scan directory. On such lines, after the open <displayName> tag, and up to the space character before the open parenthesis, an arbitrary sequence of BIDS short identifiers and value pairs, separated by underscores, can be inserted—as shown in **Figure 4A**.

Examples for manually redressed data are included in the SAMRI ParaVision testing data archive, and are in their end form indistinguishable from data acquired along the lines of the operator recommendations. In **Figure 5**, we show the changes which were required to render mouse lemur testing data compliant with the repositing workflow.

2.3. Package Management

The workflow becomes available upon installation of the SAMRI software package. The dependency list of the package is documented inside the SAMRI repository and version archive (Ioanas et al., 2019b) in accordance with the Package Manager Specification (Bennett et al., 2017). Automatic installation of the entire dependency stack has been made available for package managers conforming to this specification (Ioanas et al., 2017), including the standard package manager of Gentoo Linux and its derivatives (Ioanas et al., 2017). The package design is based on the widely-adopted Python setuptools functionality, and is thus easily accessible to maintainers using further distributions.

3. RESULTS

To demonstrate the capabilities of the workflow, we have compiled a versioned reference archive of multi-species small animal MRI data. The resulting package, samri_bindata

A

```
@@ -3225 +3225 @@
-<displayName>MSME_200um (E6)</displayName>
+<displayName>acq-MSME200um_T2w (E6)</displayName>
@@ -4218 +4218 @@
-<displayName>GE_EPI_sat_mat96_1mm (E7)</displayName>
+<displayName>acq-geEPI_task-rest_bold (E7)</displayName>
```

B

```
@@ -10,2 +10,2 @@
 ##$SUBJECT_id=( 60 )
-<MD1704-Mc285AB-P02>
+<Mc285AB>
@@ -28,2 +28,2 @@
 ##$SUBJECT_study_name=( 64 )
-<MD1704_RsLemurs_Rs02>
+<R02>
```

FIGURE 4 | Editing operations on only up to four lines are required to render preexisting data compatible with repositing via the workflow at hand. Depicted are file differences for the 20171024_165248_MD1704_Mc285AB_P02_1_1 testing dataset scan, in a patch syntax. The line numbers identifying the position of the text segment (before and after editing) are highlighted between arobase characters and in cyan. Deleted and added lines are highlighted in red or green, and prefixed with a minus or plus, respectively. Conserved lines are printed in black. **(A)** A total of two lines need to be edited in the ScanProgram.scanProgram file in order to render acquisition, task, and contrast automatically interpretable. **(B)** A total of two lines need to be edited in the subject file in order to make session and subject information automatically interpretable.

```
/usr/share/samri_bidsdata
├── 20151208_182500_4007_1_4
│   ├── 10
│   ├── 2
│   ├── 3
│   ├── 4
│   ├── 5
│   ├── 7
│   ├── 8
│   ├── 9
│   ├── AdjResult
│   ├── AdjStatePerStudy
│   ├── ResultState
│   ├── ScanProgram.scanProgram
│   └── subject
├── 20170317_203312_5691_1_5
│   ├── 1
│   ├── 2
│   ├── 3
│   ├── 4
│   ├── 5
│   ├── 6
│   ├── 7
│   ├── 8
│   ├── AdjResult
│   ├── AdjStatePerStudy
│   ├── ResultState
│   ├── ScanProgram.scanProgram
│   └── subject
├── 20171024_165248_MD1704_Mc285AB_P02_1_1
│   ├── 5
│   ├── 6
│   ├── 7
│   ├── AdjResult
│   ├── AdjStatePerStudy
│   ├── ResultState
│   ├── ScanProgram.scanProgram
│   └── subject
├── 20171106_184345_21_1_1
│   ├── 1
│   ├── 12
│   ├── 13
│   ├── 16
│   ├── 17
│   ├── 18
│   ├── 22
│   ├── 3
│   ├── 4
│   ├── 6
│   ├── 9
│   ├── AdjResult
│   ├── AdjStatePerStudy
│   ├── ResultState
│   ├── ScanProgram.scanProgram
│   └── subject
└── 20180529_041333_SN9879_1_1
    ├── 1
    ├── 2
    ├── 4
    ├── 5
    ├── 6
    ├── 8
    ├── AdjResult
    ├── AdjStatePerStudy
    ├── ResultState
    ├── ScanProgram.scanProgram
    └── subject
```

```
user@host ~ $ SAMRI bru2bids\
        -o /var/tmp/samri_testing/bash/mouse\
        -f '{"acquisition":["EPI"]}'\
        -s '{"acquisition":["TurboRARE"]}'\
        /usr/share/samri_bindata
```

```
/var/tmp/samri_testing/bash/mouse
└── bids
    ├── dataset_description.json
    └── sub-5691
        ├── ses-ofMpF
        │   ├── anat
        │   │   ├── sub-5691_ses-ofMpF_acq-TurboRARE_T2w.json
        │   │   └── sub-5691_ses-ofMpF_acq-TurboRARE_T2w.nii.gz
        │   └── func
        │       ├── sub-5691_ses-ofMpF_task-CogB_acq-EPI_run-0_bold.json
        │       ├── sub-5691_ses-ofMpF_task-CogB_acq-EPI_run-0_bold.nii.gz
        │       └── sub-5691_ses-ofMpF_task-CogB_acq-EPI_run-0_events.tsv
        └── sub-5691_sessions.tsv
```

```
user@host ~ $ SAMRI bru2bids\
        -o /var/tmp/samri_testing/bash/rat\
        -f '{"acquisition":["seEPI"]}'\
        /usr/share/samri_bindata
```

```
/var/tmp/samri_testing/bash/rat/
└── bids
    ├── dataset_description.json
    └── sub-21
        ├── ses-noFUSr0
        │   └── func
        │       ├── sub-21_ses-noFUSr0_task-MhBu_acq-seEPI_run-1_bold.json
        │       ├── sub-21_ses-noFUSr0_task-MhBu_acq-seEPI_run-1_bold.nii.gz
        │       ├── sub-21_ses-noFUSr0_task-MhBu_acq-seEPI_run-1_events.tsv
        │       ├── sub-21_ses-noFUSr0_task-rest_acq-seEPI_run-0_bold.json
        │       ├── sub-21_ses-noFUSr0_task-rest_acq-seEPI_run-0_bold.nii.gz
        │       └── sub-21_ses-noFUSr0_task-rest_acq-seEPI_run-0_events.tsv
        └── sub-21_sessions.tsv
```

```
user@host ~ $ SAMRI bru2bids\
        -o /var/tmp/samri_testing/bash/lemur\
        -f '{"acquisition":["geEPI"]}'\
        -s '{"acquisition":["MSME200um"]}'\
        /usr/share/samri_bindata
```

```
/var/tmp/samri_testing/bash/lemur/
└── bids
    ├── dataset_description.json
    └── sub-Mc285AB
        ├── ses-R02
        │   ├── anat
        │   │   ├── sub-Mc285AB_ses-R02_acq-MSME200um_T2w.json
        │   │   └── sub-Mc285AB_ses-R02_acq-MSME200um_T2w.nii.gz
        │   └── func
        │       ├── sub-Mc285AB_ses-R02_task-rest_acq-geEPI_run-0_bold.json
        │       ├── sub-Mc285AB_ses-R02_task-rest_acq-geEPI_run-0_bold.nii.gz
        │       └── sub-Mc285AB_ses-R02_task-rest_acq-geEPI_run-0_events.tsv
        └── sub-Mc285AB_sessions.tsv
```

FIGURE 5 | The workflow can automatically select, convert, and reposit ParaVision data from a general collection directory into dedicated shareable data sets, formatted for the BIDS standard. The left column shows the contents of the source directory, analogous to a collection directory on a server or scanner, and packaged by us as `samri_bindata`. The right column contains the Bash commands needed to produce dedicated mouse, rat, and lemur data sets (in that order). Below each Bash command shell, the resulting BIDS directory contents are shown. Directories are highlighted in blue. As ParaVision provides no way of integrating stimulatioin information, the workflow creates empty events files (`*_events.tsv`), which the user can fill with the appropriate stimulation content, delete, or simply ignore (if empty the files carry no meaning in BIDS).

(Ioanas et al., 2019a), includes multi-center ParaVision standard scans from mice, rats, and mouse lemurs, and serves as testing data for the SAMRI package, including the repositing workflow. The data in this archive is based on scans acquired with originally compliant operator input, as well as on preexisting scans rendered compliant *ex post facto*, as detailed in this article.

In **Figure 5**, we test the capability of the workflow to correctly and automatically source specified datasets from a diverse collection, and reposit them in the BIDS standard. The resulting three separate and shareable BIDS data sets pass the BIDS validator with no errors. The commands listed in this section are included in the SAMRI test suite, and monitored for continued quality assurance.

4. METHODS FOR IMPLEMENTATION

The data of our reference ParaVision testing archive, `samri_bindata` (Ioanas et al., 2019a), were acquired at three separate centers, using three different scanner types, and in two rodent species (mouse and rat) and one primate species (mouse lemur)—with ParaVision `6.0.1`. Mouse data were acquired at the Animal Imaging Center of the ETH and University of Zurich, using a Bruker Biospec 70/16 system, or a Bruker Biospec 94/30 system. Rat data were acquired at the Neurotechnology Group of the University of Zurich Neuroinformatics Institute, using a Bruker PharmaScan 70/16 system. Mouse lemur data were acquired at the Molecular Imaging Research Center (MIRCen) of the Commissariat à l'Énergie Atomique et aux Énergies Alternatives (CEA), using a 11.7 Tesla Bruker BioSpec system.

The workflow is implemented as a function in the Python programming language, and uses the Nipype package (Gorgolewski et al., 2011) for workflow execution, overview generation, parallelization, and access to non-Python tools. Bash bindings are auto-generated based on the Python function definition and documentation string by the Argh package.

Data conversion from the ParaVision `2dseq` format to NIfTI is performed by the `Bru2` function from the Bru2Nii package (Rorden and Naveaum, 2018). Preliminary to workflow execution, ParaVision metadata parsing and BIDS metadata assignment is performed by Python utility functions implemented in the SAMRI package. These functions iterate through the lines of the relevant metadata files: `subject` and `ScanProgram.scanProgram`, falling back to `<scan_number>/acqp` if the scan program file is corrupted. Metadata detection is performed via regular expressions, to afford a maximum of flexibility, and avoid dependency on more convoluted higher-level tools.

The processed metadata is recorded in a Pandas Dataframe (McKinney, 2010), which is both used internally and written to disk in the "work directory" to permit debugging. This record is then divided along the lines of the supported scan categories (structural, functional, and diffusion-weighted), and the resulting selections are used to initiate the respective workflow iterations, as seen in **Figure 6**. Following successful

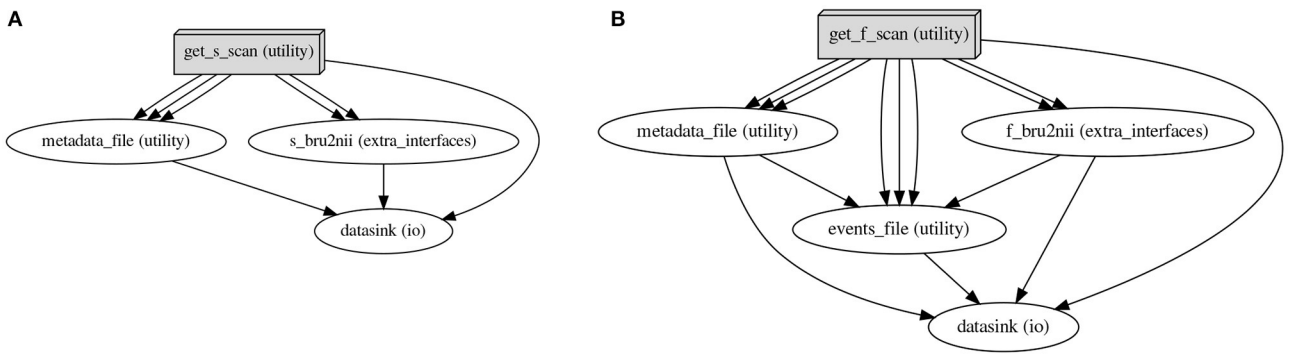

FIGURE 6 | Dedicated workflows are set up for each scan category. Depicted are directed acyclic graphs, as produced by the workflow engine, Nipype. The `bru2bids` function iteratively executes e.g., structural **(A)** and functional **(B)** repositing workflows, contingent on data availability. Node names specify the source code identifiers, with the text in parentheses indicating, which modules they are implemented in: `utility` and `extra_interfaces` are modules of the SAMRI package, and `io` is a module of the Nipype package. **(A)** Structural scan category repositing graph. The `get_s_scan` node queries the base Pandas Dataframe for scan entries selected as structural, and computes data descriptors, which are further piped in the following fashion: (1) the `metadata_file` node, responsible for creating a JSON BIDS metadata file is supplied with (i) a bundled subject and session identifier, (ii) the exact ParaVision scan path including the numbered subdirectory, and (iii) the computed BIDS output metadata file name; (2) the `s_bru2nii` node, responsible for creating a BIDS-named NIfTI data file is supplied with (i) the exact scan path including the numbered subdirectory and (ii) the computed BIDS output file name; (3) the `datasink` node, responsible for creating a BIDS-style directory hierarchy is supplied with a bundled subject and session identifier. **(B)** Functional scan category repositing graph. The `get_f_scan` node queries the base Pandas Dataframe for scan entries selected as functional, and computes data descriptors, which are further piped in the following fashion: (1) the `metadata_file` node, responsible for creating a JSON BIDS metadata file is supplied with (i) a bundled subject and session identifier, (ii) the exact ParaVision scan path including the numbered subdirectory, and (iii) the computed BIDS output metadata file name; (2) the `f_bru2nii` node, responsible for creating a BIDS-named NIfTI data file is supplied with (i) the exact scan path, and (ii) the BIDS output file name; (3) the `events_file` node, responsible for creating a BIDS-style events file is supplied with (i) the exact scan path, (ii) the task identifier, (iii) the BIDS output event file name, (iv) the metadata file (to query delay entries), and (v) the functional file path (to determine the repetition time necessary to perform eventfile adjustments in light of the metadata); (4) the `datasink` node, responsible for creating a BIDS-style directory hierarchy is supplied with a bundled subject and session identifier.

workflow execution, the BIDS standard `*_sessions.tsv` file is separately generated, recording the onset acquisition times for each session.

The guidelines regarding operator input and preexisting data compliance presented in this document are based on the ParaVison `6.x` release series. The parsing functionality of the workflow in the current SAMRI version (`0.4.1`) is based on the ParaVison `6.x` release series and the BIDS `1.x` specification.

5. DISCUSSION

The `bru2bids` workflow presented herein is a significant first step in rendering data in the Bruker ParaVision standard automatically interpretable for high-level analysis pipelines. This is done by repositing the ParaVision data according to the BIDS standard, which offers superior legibility, as well as integration with community analysis tools, specifically with tools adapted from human fMRI. The BIDS reposited data form can serve as a mere intermediary, facilitating data usage with standardized workflows, but can also be used in and of itself as raw data recourse—if data management expediency is prioritized over the larger pool of accessible information in the full ParaVision standard.

To demonstrate and persistently track compliance, we release a versioned archive of Bruker ParaVision testing data, diverse in terms of both animal species and acquisition protocols. We test the performance of the workflow on the dataset, and report compliance with the target standard. These demonstrated capabilities of the workflow are rendered accessible to the community by procedural instructions addressed to Bruker MRI scanner operators. Additional accessibility is conferred by a detailed walk-through, which allows custodians of Bruker ParaVision data to render preexisting records compatible as a workflow input. Further, we describe the general principles of the software implementation, which in conjunction with the documentation internal to the software enable collaborators to inspect, debug, augment, or create derivations based on our work.

The demonstrated performance of this workflow, its position at the transition from the most popular small animal MRI acquisition format into the most popular MRI data sharing format, as well as its transparent free and open source nature, make the SAMRI's `bru2bids` a strong foundation for the rapid and collaborative improvement of fMRI data analysis methodology.

DATA AVAILABILITY STATEMENT

All datasets generated for this study are included in the article/**Supplementary Material**.

ETHICS STATEMENT

Ethical review and approval was not required for the animal study because this study concerns an ex post facto technological method, for which only preexisting animal data was used.

AUTHOR CONTRIBUTIONS

The final submitted form of this document was edited by H-II. The document was drafted by H-II and MM. The text review process encompassed all authors. The software was authored by H-II and MM. Data was acquired by H-II, MM, and CG. Technology access and institutional backing was provided by MR, MY, and MD. Material support, supervision, and computational resources, as required during pipeline development, were offered by MR and MY.

FUNDING

The workflow development work was supported by Swiss National Science Foundation (SNF) (Grant Nos. 310030-160310 and 310030-179257) granted to MR.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fninf.2020.00005/full#supplementary-material

REFERENCES

Bennett, S. P., Faulhammer, C., McCreesh, C., Müller, U., and Gentoo Linux. (2017). *Package Manager Specification*. Available online at: https://projects.gentoo.org/pms/6/pms.html

Bukhari, Q., Schroeter, A., and Rudin, M. (2018). Increasing isoflurane dose reduces homotopic correlation and functional segregation of brain networks in mice as revealed by resting-state fMRI. *Sci. Rep*. 8:10591. doi: 10.1038/s41598-018-28766-3

Cox, R. W., Ashburner, J., Breman, H., Fissell, K., Haselgrove, C., Holmes, C. J., et al. (2004). "A (sort of) new image data format Standard: NIfTI-1," in *OHBM* (Budapest, HU).

Ferraris, S., Shakir, D. I., Merwe, J. V. D., Gsell, W., Deprest, J., and Vercauteren, T. (2017). Bruker2nifti: Magnetic Resonance Images converter from Bruker ParaVision to NIfTI format. *J. Open Source Softw*. 2:354. doi: 10.21105/joss.00354

Gorgolewski, K., Burns, C. D., Madison, A., Clark, D., Halchenko, Y. O., Waskom, M. L., et al. (2011). Nipype: a flexible, lightweight and extensible neuroimaging data processing framework in Python. *Front. Neuroinform*. 5. Available from: http://dx.doi.org/10.3389/fninf.2011.00013

Gorgolewski, K. J., Auer, T., Calhoun, V. D., Craddock, R. C., Das, S., Duff. E. P., et al. (2016). The brain imaging data structure, a format for organizing and describing outputs of neuroimaging experiments. *Sci. Data* 3:160044. doi: 10.1038/sdata.2016.44

Grandjean, J., Canella, C., Anckaerts, C., Ayrancı, G., Bougacha, S., Bienert, T., et al. (2019). *Common Functional Networks in the Mouse Brain Revealed by Multi-Centre Resting-State fMRI Analysis*. Available online at: https://doi.org/10.1101/541060

Hunt, J. W., and MacIlroy, M. D. (1976). *An Algorithm for Differential File Comparison*. Available online at: http://www.cs.dartmouth.edu/~doug/diff.pdf

Ioanas, H. I., Marks, M., Garin, C. M., Dhenain, M., Yanik, M. F., and Rudin, M. (2019a). *ParaVision Mouse, Rat, and Lemur Testing Data for SAMRI*. Available online at: https://doi.org/10.5281/zenodo.3234924

Ioanas, H. I., Marks, M., Segessemann, T., Schmidt, D., Aymanns, F., and Rudin, M. (2019b). *SAMRI—Small Animal Magnetic Resonance Imaging*. Available online at: https://doi.org/10.5281/zenodo.3566868

Ioanas, H. I., Marks, M., Yanik, M. F., and Rudin, M. (2019c). An optimized registration workflow and standard geometric space for small animal brain imaging. *bioRxiv*.

Ioanas, H. I., Saab, B., and Rudin, M. (2017). Gentoo Linux for Neuroscience — a replicable, flexible, scalable, rolling-release environment that provides direct access to development software. *Res. Ideas Outcomes.* 3:e12095. doi: 10.3897/rio.3.e12095

McKinney, W. (2010) "Data structures for statistical computing in Python," in *Proceedings of the 9th Python in Science Conference*, eds S. van der Walt and J. Millman (Austin, TX), 51–56.

Rorden, C., and Naveaum M. (2018). *Bruker ParaVision to NIfTI Conversion*. Available online at: https://web.archive.org/web/20180611113306; https://github.com/neurolabusc/Bru2Nii

Schlegel, F., Schroeter, A., and Rudin, M. (2015). The hemodynamic response to somatosensory stimulation in mice depends on the anesthetic used: implications on analysis of mouse fMRI data. *Neuroimage* 116, 40–49. doi: 10.1016/j.neuroimage.2015.05.013

Schroeter, A., Grandjean, J., Schlegel, F., Saab, B. J., and Rudin, M. (2016). Contributions of structural connectivity and cerebrovascular parameters to functional magnetic resonance imaging signals in mice at rest and during sensory paw stimulation. *J. Cereb. Blood Flow Metab*. 37, 2368–2382. doi: 10.1177/0271678x16666292

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2020 Ioanas, Marks, Garin, Dhenain, Yanik and Rudin. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.