SIEMENS

Algorithm Validation Toolkit
AVT2EXT

[redacted] SCR

Algorithm Validation Toolkit
Functional Specification

R 1.0

Last Change: 4/30/2010 5:44 PM

Robert W. Schwanke
SCR-SAP

History

Document History

Version/ Status	Date of Issue	Author	Change and Reason of Change / Change Request/CHARM
R0.1/Draft	9-Nov-09	Robert Schwanke	First Draft
R0.3/Draft	2-Feb-10	Robert Schwanke	First complete draft, including requirements from RSNA and choices on standard sections to include or exclude.
R0.4/Approved	1-May-10	Robert Schwanke	Release candidate
R1.0/Released	10-May 10	Robert Schwanke	Released

History of released Versions

Version	Release date	Product Version
1.0	10-May-2010	AVT2EXT

Table of Contents

1 Introduction

1.1 Purpose of the document

The Functional Specification defines the use cases implemented in the current development cycle, and defines the user interface design of the product by means of a set of screen shots with textual descriptions. All requirements in the corresponding Requirement Specification[1] that are not deferred or rejected are traced to use cases in this document. Some use cases listed here are not implemented in the current development cycle; however, each leaf-level use case that is implemented is traced to a section of the test scripts[2] that test it.

1.2 Area of validity of the document

This document applies to Algorithm Validation Toolkit, version AVT2EXT.

1.3 Definitions and abbreviations

See *AVT2EXT Definitions and Abbreviations*, a separate document.

1.4 References

[1] Algorithm Validation Tookit, Requirement Specification, version 0.3

[2] Test Scripts for AVT2EXT, https://collab01a.scr.siemens.com/avtwiki/index.php/Test_Scripts_for_AVT2EXT

[3]

2 General description

2.1 Product goal

See Requirement Specfication

2.2 Product profile

See Requirement Specification

2.3 Required functionalities

See *AVT2EXT Vision, Scope, and Technical Overview.*

2.4 Interfaces and interactions

See *AVT2EXT Design Specification.*

2.5 Assumptions, dependences, limitations

See Requirement Specification.

2.6 Risk analysis

Not Applicable.

This product is not approved for clinical use, and hence has no safety concerns.

As a precaution, the user is required to acknowledge a terms-of-use statement to this effect.

3 SW system architecture

< Specifying the software system architecture >

<Process model: interfaces, communication; e.g. context diagram >

<Alternative data model: information structure, data structure; e.g. Entity Relationship diagram of the information modeling (IM)>

<Design limitations: programming language, standards, resource limits (e.g. memory, ...), HW configuration (e.g. processor, periphery), SW configuration (libraries, operating system, ...)>

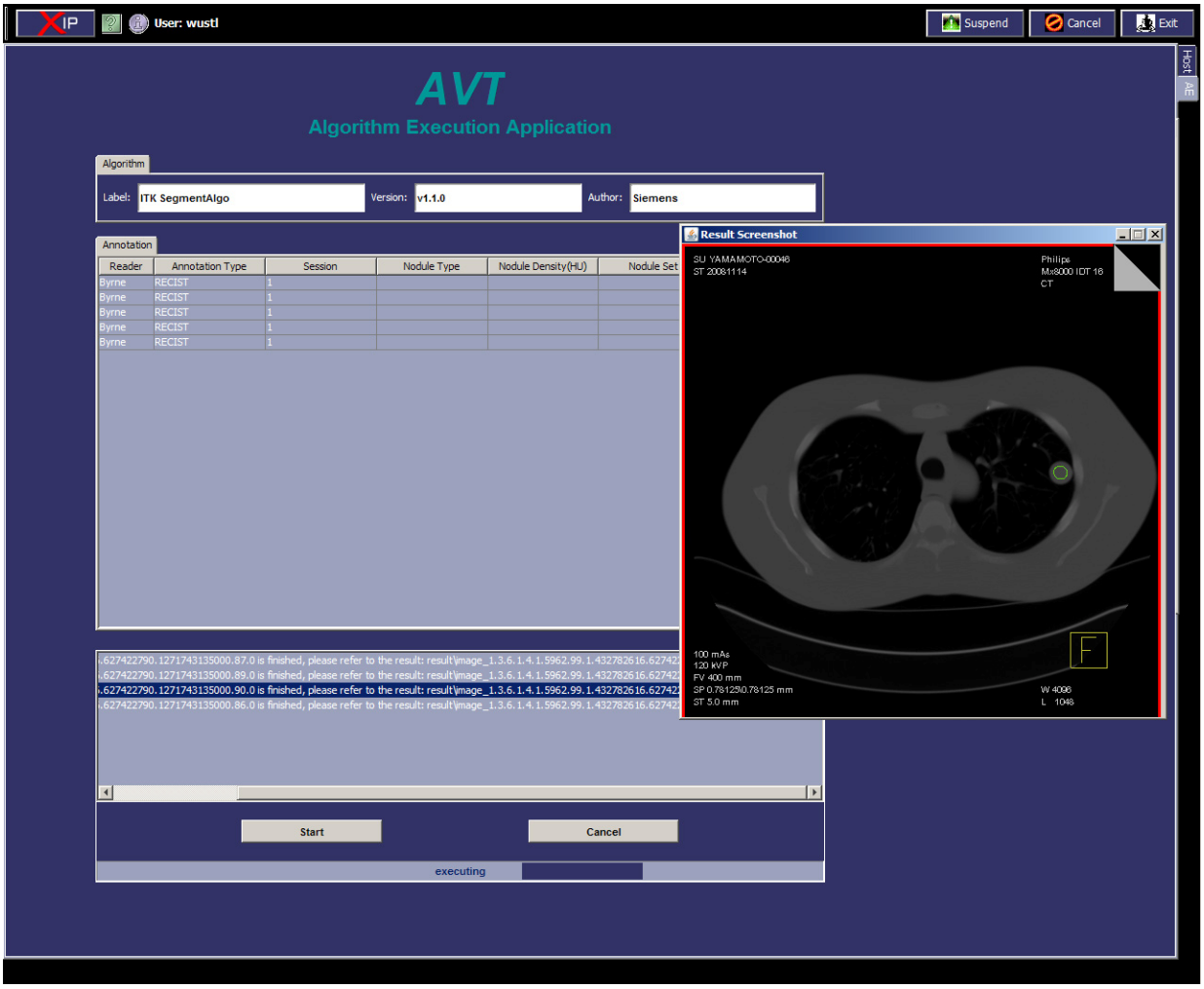
4 Interfaces

4.1 User interfaces

See

- AVT2EXT Image Annotation User Manual
- AVT2EXT Image Annotation User Manual

4.1.1 Algorithm Execution User Interface



The AE user interface is depicted in the figure above.

- The Algorithm section allows you to indicate the name and version of the algorithm and the author.
- The Annotation section lists the seed annotations (or RECIST annotations, which can substitute fore seed annotations) that will be fed to the algorithm.
- The Start button starts the batch execution.

- The progress bar across the bottom shows whether the algorithm is currently executing.
- The Cancel button cancels the batch at the end of the currently-running execution of the algorithm.
- The (unlabelled) transcript lists a sample image showing segmentation results for each run of the algorithm, when the run finishes. Double-clicking on the a line of the transcript window opens the sample image in a pop-up window for examination.
- The pop-up window shows one slice of the series with the contour produced for that slice by the algorithm.

4.2 Hardware interfaces

Not applicable.

4.3 Software interfaces

See *AVT2EXT Design Specification*, a separate document.

4.4 Communication interfaces

Not applicable.

5 Use Cases

AVT is being developed in an incremental, iterative process. Each iteration adds functionality to support another research case study. So far, three research *case-studies* (not "use-cases") have been or are being implemented:

- **APA**: Algorithm Performance Analysis
  - Analyzes liver tumor images and ground truth volume segmentations from MICCAI'08 Grand Challenge data.
  - Uses an automatic volume segmentation algorithm built from Itk components.
  - Carries out statistical analysis of volume errors to analyze strengths and weaknesses of algorithm.
  - In this document, APA is described the way it would be done today. At the time it was done (June 2009), not all of the functionality was available.
- **TCGA**: TCGA Radiology project, involving *glioblastoma multiforme* (GBM) segmentation
  - Analyzes GBM series from TCGA collection on NBIA.
  - Performs manual segmentation of individual slices of a series
  - Prototypes template-driven capture of Vasari characteristics of GBM tumors.
  - Exchanges AIM annotations with other Image Readers to check interoperability.
  - No measurement variability analysis.
- **SoVA**: Sources of Variation Analysis
  - Analyzes thoracic phantom images organized by CDRH, soon to be available in a collection on NBIA
  - Performs manual capture of seed annotations for use in algorithm execution.
  - Manual and automatic volume segmentation, as well as RECIST and WHO diameter measurements.
  - Performs statistical analysis that compares contributions to error from different independent variables.
  - Performs statistical analysis of reader variability (intra-reader and inter-reader).

This use-case specification is organized as a hierarchy of use cases, with annotations indicating which parts of each use case are used in each case-study.

Each use case is decomposed into parts, each related to the parent in one of three ways:

- Includes: The parent use case always includes this part
- Includes (Optional): The parent use case sometimes includes this part and sometimes does not. (This refers to the user's choice, not the development team's choice! Anything that is not going to be implemented in the current development cycle should be marked {deferred}.)
- Special Case: This part is a special case of the parent use case.

About *"optional"*: a part of a use case is marked optional to indicate that sometimes the parent activity includes that sub-activity and sometimes it does not. For example, some users of AVT will install their own image processing algorithm and some will just use the one that is already installed. So, in the diagram above and the specification below, "***AVT** includes (optionally)* ***Install Image Processing Algorithm***". If no part of the functionality of a step in a use-case is being implemented in the current development cycle, it should be marked {deferred}.

The use cases defined here do not specify the number of times each of their parts is executed. The parts are listed in a logical order, but a part might be executed one or more times, interspersed with the execution of other parts, depending on what the user wants to do.

Some of the parts are labeled "SUC" and hyper-linked to a section giving more details. The other section is a Sub-Use-Case of the present section. The other parts are simply described in-line in their parent use case's description.

The use case descriptions are also marked with requirement keys, listed in small_Arial_font, indicating correspondence between steps of use cases (or whole use cases) and requirements.

Finally, some of the use case items are marked with a square box, e.g.

- Filter Cases to Analyze

These lines should exactly match subsection headings in the test script in the Wiki, thus defining traces between use cases and test cases.

Traceability and Coverage

Every leaf-level use case step should either be marked with a requirement, with {deferred}, or with {OOPS} (Out of Project Scope). Those that are executable should be traced to test cases as above.

5.1 Main UC: AVT Tumor Measurement Research

This is the overall use case that includes everything that anyone does with AVT for tumor measurement research..

Includes

- Install AVT

INSTALL_installation_procedure
INSTALL_release_notes
INSTALL_binary_code
INSTALL_source_code
INSTALL_end_user_scenario_documentation
INSTALL_end_user_feature_documentation
XIPHost_default_working_directories

- Import DICOM Series into AD

MISC_grid_connectivity DEFERRED

DATA_DICOM_image_types DEFERRED

DATA_thoracic_phantom_images

AD_multiple_collections Deferred

AD_curation_operations *Deferred*

Includes (optional)

- SUC: Install Image Processing Algorithm
- SUC: Customize Image Reader to Annotation Protocol
- SUC: Customize Measurement Variability Tool
- Convert Ground Truth and Seed data to AIM {OOPS – unsupported scripts for the CDRH case study}

DATA_CDRH_annotation_import

- Import Ground Truth and Seed Data as AIM
- SUC: Create Seed Annotation
- SUC: Interactive Annotation
- SUC: Batch Annotation
- SUC: Measurement Variability Analysis
- Export AIM Objects {deferred}

AD_curation_operations *Deferred*

The following requirements apply to the entire use case above.

MISC_open_source
MISC_operational_pilot
DATA_seed_AIM_object

-

5.2 SUC: Install Image Processing Algorithm

One of the principal uses of AVT is to compare a new image processing algorithm to other methods (manual and/or automatic) of doing the same task.

Includes, optionally

- Install a C++ image processing algorithm in a scene graph library
  1. Embed the source code of an image processing algorithm in a subclass of a suitable XIP scene graph component class.
  2. Compile and link the scene graph component into a binary library, (e.g. a DLL in Windows)
- Install a non-C++ image processing algorithm in a scene graph library {deferred}
- Configure a custom scene graph library into AVT

- Configure a scene graph pipeline to implement an image processing algorithm

For APA and SoVA, the SCR team will perform this step on behalf of the eventual user. TCGA did not use image processing algorithms. The delivered documentation will describe how others can install their own image processing algorithms.

5.3 SUC: Customize Image Reader to Annotation Protocol

Typically, customization is needed to capture additional observations and markups, control the order in which annotation takes place, and arrange the AIM format in which the annotations are stored. At present, the Image Reader can only be customized by programming.

The TCGA Radiology project specified a template for annotation capture, which the AVT team implemented by hand for TCGA phase 1. In the future, we would like to implement a template-driven Image Reader interface to make it easier to configure the Image Reader to the needs of a new annotation protocol.

DATA_thoracic_phantom_tumor_AIM_object

IA_template_driven_symbolic_annotation *DEFERRED*

5.4 SUC: Customize Measurement Variability Tool

Customize MVT to support the particular kinds of calculation and exploration needed for the specific research project to which AVT is being applied. The AVT installation package supplies an example Measurement Variability Tool to copy and customize. It can be customized in several ways:

- At run-time, by selecting a subset of the available analysis methods to execute.
- At run-time, by writing scripts in the R statistical programming language following the pre-defined syntax , for things like summary statistics.
- At development time, by programming new analysis methods in R. {deferred}
- At development time, by extending the user interface with new ways to view the data, written in Java and/or C++. {deferred}

Includes (optional)

- Customize Independent Variables {by programming}

MVT_independent_variables

MVT_independent_variables_future *deferred*

MVT_suppress_irrelevant_variables

- Customize Dependent Variables {by programming}
- Customize Measures {by programming; deferred}
- Customize Summary Statistic
- Develop new statistical routine {by programming}

  - written as R package to fit MVT/R interface
  - Load R package into MVT

- Configure MVT report generators (or write new ones) { deferred , by programming}
- Configure MVT table views (or write new ones) { deferred , by programming}
- Configure MVT image views (or write new ones) { deferred , by programming}

DATA_thoracic_phantom_tumor_AIM_object

5.5 SUC: Create Seed Annotation

A seed annotation is used in AVT to define the starting context for a measurement task, which task could be done interactively or by a batch process. It is a special case of interactive annotation, in which the emphasis is on recording information that is already known, rather than on performing accurate measurements.

Rather than relying on esoteric concepts of inheritance, we have repeated the details of the parent use-case that apply to the sub-case. We have not repeated the requirement keys.

Includes

- Acknowledge Terms-of-use
- SUC: View Image as 3D-Multi-planar Reconstruction
- SUC: View Fused Volume Rendering
- Input User Information for Audit Trail

Includes (optional)

- Manually Mark RECIST Diameter
- Manually Mark WHO Diameter
- Manually Segment Tumor
- Add or Subtract Closed Contour
- Markup ROI Location
- Algorithmically Segment Tumor
- Semi-automatically Segment Tumor
- Automatically Segment the Tumor
- Expand a contour manually
- Reduce a contour manually
- Calculate maximal slice and RECIST and WHO diameters from segmentation.
- Classify Nodule Characteristics manually{For CDRH this includes:

- Nodule type: 10 mm sphere, 20 mm sphere, 20 mm ovoid, 10 mm lobulated, 10 spiculated
- Nodule Density: -10HU, +100HU
- Nodule Set (First, Second, etc., representing repeat scans)
- Reading Session (First, Second, Etc.)
- Measurement Method (RECIST, WHO, Volume)
- Case Name
- Nodule anatomical location (pick from list)}

Includes

- Add Audit Trail Comment
- Mark Annotation as Seed Annotation

IA_store_seed_annotations

- Save Annotation as AIM Object

This is a special case of SUC: Interactive Annotation

5.6 SUC: Interactive Annotation

Interactive segmentation includes manual, semi-automatic, and automatic identification of the boundary of a single tumor. In the current version of this model, segmentation means 3D segmentation, but when done manually, it is done by drawing contours on each of the image slices where the tumor is visible. Automatic segmentation may be stored in any of several forms, not specified here, but is displayed as a set of contours on slices, just as if it had been drawn manually.

Includes

- Invoke Image Reader from XIPHost
- Acknowledge Terms-of-use

MISC_IA_clinical_use_disclaimer

- Read Annotation Instructions *{deferred}*
- SUC: View Image as 3D-Multi-planar Reconstruction

IA_dynamic_viewport_layout

- SUC: View Fused Volume Rendering
- Input User Information for Audit Trail

AUDIT_user_role DEFERRED

Includes (optional)

- Load AIM Annotation
- Navigate to Tumor

- Change the Pane Arrangement
- Create an Observation
- Delete an Observation
- Create Multiple AIM Annotations
- Manually Mark RECIST Diameter

SEG_IA_manual_diameter_on_a_2D_slice

- Manually Mark WHO Diameter

SEG_IA_manual_orthogonal_diameters_on_a_2D_slice

- Manually Segment Tumor

SEG_IA_manual_3D_volume_segmentation

- Expand a contour manually
- Reduce a contour manually

SEG_IA_edit_volume_segmentation_contours

- Add a Seed Line

SEG_IA_mark_tumor_with_seed
SEG_automatic_3D_volume_segmentation
SEG_ITK_volume_segmentation_algorithm
SEG_IA_invoke_automatic_volume_segmentation

- Semi-automatically Segment Tumor
- Automatically Segment the Tumor
- Expand a contour manually
- Reduce a contour manually
- Calculate maximal slice and RECIST and WHO diameters from segmentation. {deferred}

SEG_diameter_calculations_from_volume_segmentation *DEFERRED*

- Describe Tumor as specified in protocol.
- Describe GBM Tumor – Vasari Protocol {TODO: document previous work}
- Rate User's Confidence in Annotation

IA_label_confidence_scale_from_RadLex ~~CHEAP~~

- Add Audit Trail Comment

AUDIT_comments

- Save Annotation as AIM Object
- Exit the Image Reader

DATA_meaningful_AIM_file_name
IA_store_pan_zoom_window_level DISCUSSING
IA_save_warnings CHEAP

Has special case

- SUC: Create Seed Annotation

The following requirements keys apply to the entire sub-use-case above.

XIPHOST_query_seeds
DATA_load_and_display_AIM_annotations
IA_navigate_to_markup
IA_SoV_case_study
IA_multiple_AIM_annotations_per_image_reader_session
IA_store_annotations
AUDIT_image_reader
AUDIT_create_annotation
AUDIT_algorithm_name_and_version

5.7 SUC: View Image as 3D-Multi-planar Reconstruction

This functionality, being already implemented, is not described in detail here. The image below illustrates the following requirements.

Includes

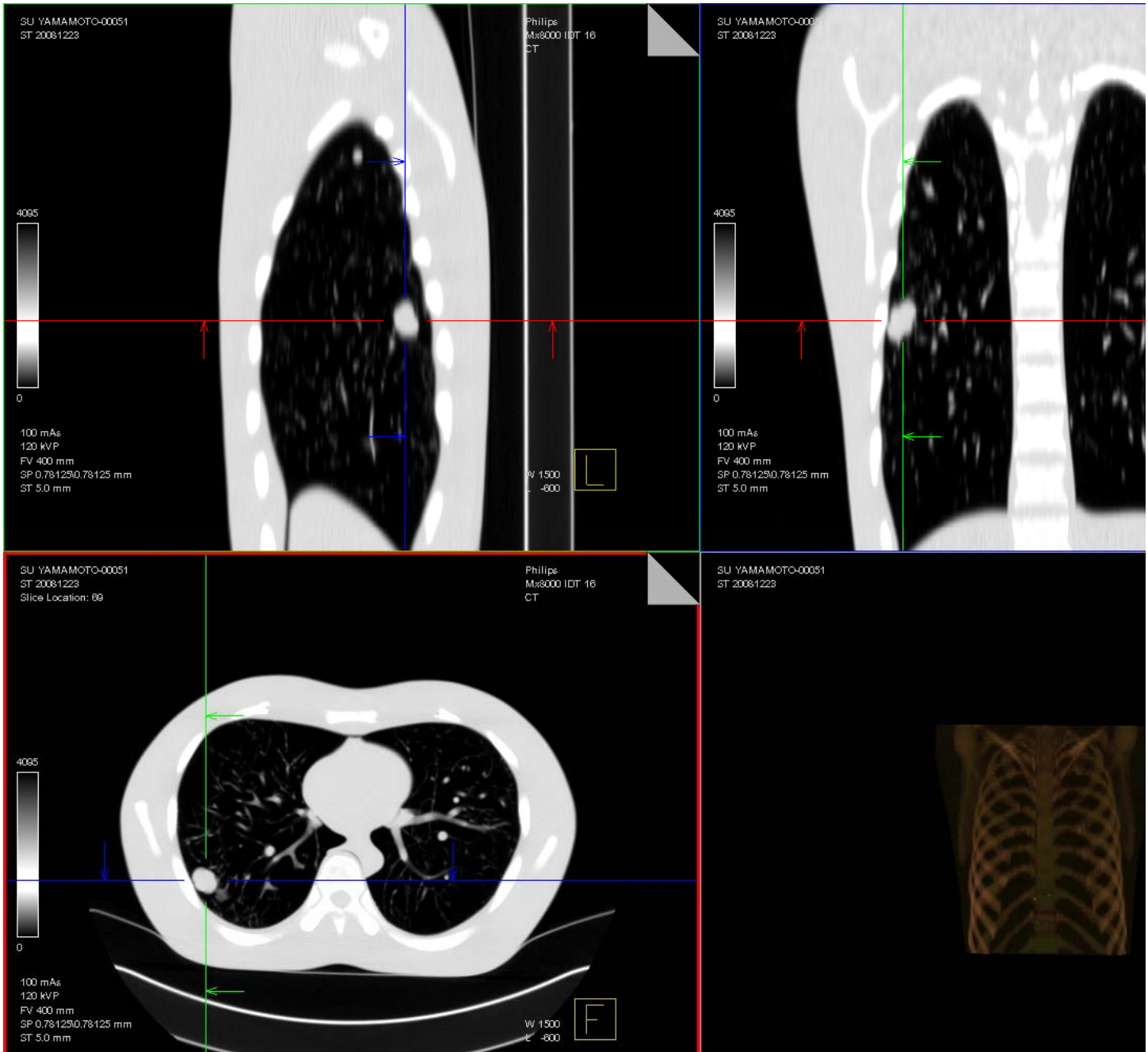
- View 3D-MPR Image

VIEW_double_oblique_MPR_viewing
VIEW_text_overlay
VIEW_orientation_cube

VIEW_scale *DEFERRED*

VIEW_CT_window_and_level_settings
VIEW_window_and_level_presets

VIEW_phantom_presets *DEFERRED*



VIEW_window_and_level_adjustment

VIEW_MR_auto_windowing DEFERRED

VIEW_pan_zoom
VIEW_mouse_cursor_feedback_on_adjustment_tools
VIEW_volume_segmentation_display_as_contours_on_slices
VIEW_volume_segmentation_display_on_alternate_planes

5.8 SUC: View Fused Volume Rendering

The figure above depicts volume rendering in the lower right quadrant.

Includes

- View Volume Rendering

VIEW_fused_volume_rendering

5.9 SUC: Batch Annotation

Batch segmentation means applying a single segmentation algorithm to each of a potentially large number of cases, where a case is specified by a seed annotation and the associated image series. Once the user initiates the batch, no further supervision is required, although the user can monitor progress and abort the execution if something desired.

Includes

- Customize AE To Use Desired Algorithm (See *AVT2EXT Programming Guide*)

AE_algorithm_Plug_in_Interface

- Select Seed AIM Objects
  - in XIPHost (and their corresponding Series)

AE_input_cases

- Submit Batch to AE

HOST_query_exclude_series

- (Optional) Monitor AE Progress

AE_progress_indicator

- (Optional) Cancel Batch Execution

AE_cancellation

- Review AE Results Summary

AE_results_summary

- The following requirements keys apply to the entire sub-use-case above.

AE_batch_segmentation
AE_WG23_hosted_application
XIPHOST_query_seeds
AE_store_results
DATA_meaningful_AIM_file_name
AUDIT_algorithm_execution
AUDIT_create_annotation
AUDIT_algorithm_name_and_version
AUDIT_AVT_version_date_and_time

5.10 SUC: Measurement Variability Analysis

Measurement Variability Analysis is the "meat" of AVT. It is a carefully integrated workflow in which the experimental measurements are compared to one another and to Nominal Ground Truth, statistical measures are calculated, outliers identified and examined, graphs plotted, reports generated, and so forth. All the parts are optional because the may be performed or skipped depending on characteristics of the data and the purpose of the analysis.

Includes

- Query Cases To Analyze

MVT_CDRH_case_capacity
MVT_CDRH_annotation_capacity
MVT_null_calculation_warning

MVT_UI_case_selector_advanced *Deferred*

HOST_query_all_experimental_variables
HOST_query_exclude_series

Includes (optional)

- Filter Cases to Analyze {deferred}
- Manually Select Cases To Exclude

MVT_exclude_individual_cases ~~CHEAP~~

- Designate Nominal Ground Truth Annotations

MVT_ground_truth_reader

- Calculate Measurements

MVT_error_difference_measures

- Calculate Summary Statistics

MVT_statistics_selector_panel *DEFERRED*

- Specify Outliers

MVT_outliers_SD
MVT_outliers_IQR
MVT_highlight_outliers CHEAP

- Visually Compare Segmentations

VIEW_double_oblique_MPR_viewing
VIEW_volume_segmentation_display_as_contours_on_slices
VIEW_multiple_read_only_markups
MVT_MPR_markup_comparisons

- Select Independent Variables
- t-Test

MVT_t_test DEFERRED

- N-way ANOVA Analysis

MVT_one_way_ANOVA_methods
MVT_factorial_ANOVA_methods
MVT_variable_selection

- Multiple Regression Analysis

MVT_multiple_regression

- Mixed Effects Analysis

MVT_mixed_effects DEFERRED

- Levene Test

MVT_Levene_test DEFERRED

- Statistical Plotting

MVT_histogram_charts

MVT_box_and_whisker_charts DEFERRED

- Generate Statistics Reports

MVT_statistic_analysis_report DEFERRED

MVT_export_plots

MVT_export_documents DEFERRED

MVT_export_data DEFERRED

- Exit MVT

Has special case

- SUC: Sources of Variation Analysis
- SUC: Reader Variability Analysis
-

The following requirements keys apply to the entire sub-use-case above.

DATA_load_and_display_AIM_annotations
MVT_list_original_measurements ~~CHEAP~~
MVT_SoV_case_study
MVT_existing_summary_statistics

5.11 SUC: Sources of Variation Analysis

This is a special case of Measurement Variability Analysis, in which all experimental measurements are compared to Nominal Ground Truth. All of the statistical analysis methods provided in the parent use case are applicable to each special case.

Includes (Optionally)

- Select Independent Variables
- Multiple Regression Analysis
- ANOVA Analysis

MVT_partitioning_by_values *DEFERRED*

5.12 SUC: Reader Variability Analysis

This is a special case of Measurement Variability Analysis in which Nominal Ground Truth, if any, is ignored, and measurements are simply compared to one another. Inter-reader variation compares measurements by each reader to measurements by each other reader. Intra-reader variation compares multiple readings of the same measurement by the same reader to each other. All of the statistical analysis methods provided in the parent use case are applicable to each special case.

Includes

- Reader Variation
- Visualize Reader Variation
- Box Plot of Reader Variation

MVT_box_and_whisker_charts *DEFERRED*

MVT_intra_reader_variation
MVT_inter_reader_variation

5.13 Installation

See the Installation instructions in the AVT Wiki.

6 Licensing

TODO: say a few words about the caBIG licensing model.

7 Unsettled points / Open issues

none

Table of Requirement Keys

Copyright © Siemens AG 2007, 2010. All rights reserved. For internal use only.
Alle Rechte vorbehalten. Nur für internen Gebrauch.