ALERT ATR Project: Ground Truth Labeling

Author: Franco Rupcich

Email: [franco.rupcich@gmail.com](mailto:franco.rupcich@gmail.com)

Phone: 414-559-3338

Version 4

Contents

1 Introduction 2

2 Definitions 2

3 Background 2

3.1 Ground Truth Labeling 2

3.2 Selecting a Ground Truth Labeling Tool 3

3.3 MeVisLab 3

4 Ground Truth Labeling using MeVisLab 4

4.1 Overview 4

4.2 Instructions for Use 7

5 Lessons Learned about applying MVL 9

5.1 Online Support 9

5.2 Crashing during interpolation 9

5.3 Thin Sheets 9

5.4 CT Artifacts 9

6 References 9

7 Revision History 10

# Introduction

This document describes how ground truth labels were generated for the TO4 project. In general, these labels were created using MeVisLab, a public domain image processing program.

# Definitions

See the document entitled “ALERT ATR Project: Top-Level Specifications” for the list of terms and acronyms for this project, as well as for details regarding targets versus. pseudo-targets, detection, false alarms, misses, probability of detection, probability of false alarm, recall, and precision.

# Background

## Ground Truth Labeling

The purpose of the ALERT ATR Project (also known as Task Order 4, or TO4) is to address improvements in CT-based explosive detection equipment by developing improved ATR algorithms. An ATR algorithm takes as input a CT image containing objects, some of which are targets, and outputs a label image, which contains labels of objects for which the ATR has considered as targets. In order to assess the performance of an ATR algorithm, the label image it produces must be scored against a corresponding label image containing all targets in the corresponding CT image. This type of label image is referred to as the “ground truth label image.”

Ground truth labeling requires that for each target in a 2D CT slice all pixels in that slice for that target must be identified and given a unique label. All the labels for all the slices that contain the target are combined into a single image, called the *ground truth label image*, which is the same size (x,y, and z) as the original CT image. This process is denoted segmentation. In general, the segmentation can be performed using one of the three following methods:

1. Automatic segmentation
2. Manual segmentation
3. Semi-automated segmentation

The first method is not a viable option, since no known automated segmentation method is able to provide sufficient performance in segmenting any given object type from airline luggage. This is due to a number of reasons, including splitting of objects as a result of image artifacts (beam-hardening, metal streak and shading artifacts) and merging of physically touching objects with similar CT numbers. Indeed, improving automated segmentation techniques is a major goal of TO4.

The second method requires manual segmentation of each target in each slice of an image, making this approach prohibitively time-consuming or prohibitively expensive.

The third method is the middle ground between the first two. Semi-automated techniques allow the human user to segment the objects known to be targets. The computer-assisted aspect of the tool can speed up the segmentation process by guiding the contours being drawn by the user and by interpolating contours for an object across slices.

## Selecting a Ground Truth Labeling Tool

Several free visualization and processing software packages [1][2][3] were investigated by Seemeen Karimi during Task Order 1: Segmentation Initiative (TO1), which also required that ground truth labeling be performed [4]. Karimi reported that while these tools are simple to learn, they do not allow interpolation between contours, a significant time-saving process. In addition, Karimi concluded that some of these programs offer other methods for semi-automatic segmentation, but the results cannot be manually edited.

Karimi chose to use the MeVisLab image processing software tools for ground truth labeling during TO1, as they allowed for guided contouring of images, as well as for interpolation of contours across image slices.

For this project, we chose to use MeVisLab based on Karimi’s recommendations. In addition, we were able to modify the existing code that Karimi had created for TO1 to suit the purposes of TO4.

## MeVisLab

The MeVisLab image processing software provides an image viewer and image processing tools that enable computer-assisted segmentation. MeVisLab is being developed and used by [MeVis Medical Solutions AG](http://www.mevis.de/mms" \t "_blank) and [Fraunhofer MEVIS](http://www.mevis.de/mre" \t "_blank) (formerly MeVis Research GmbH) in Bremen, Germany.

An image processing task is achieved by creating a graphical program in MeVisLab. The graphical program is also called a network. The network is created of interconnected modules. Each module performs a specific operation. A module may receive input data from a module and may create output data that it passes to the next module. A module may provide control or book-keeping functions.

The MeVisLab software can be downloaded from the following website.

<http://www.mevislab.de/download/>

The software runs on Windows, OS X, and Linux operating systems. The Mac OS X version was used in this work.

The MeVisLab forums provide additional support from the developers and can be found at

<http://forum.mevis.fraunhofer.de/index.php>

# Ground Truth Labeling using MeVisLab

## Overview

We have created a computer-assisted manual segmentation network, called *TO4 Ground Truth Segment and Labeling Network.mlab*, which is a modified version of Karimi’s network created during TO1. The following sections outline the basic steps taken to create a ground truth image. Detailed step-by-step instructions are given in Section 4.2. Figure 1 is an image of the network in MeVisLab.

### Load CT Image

The user must first load a DICOM version of a CT image containing the objects that will be labeled. A label image is created using the same dimensions as the CT image. To start, all pixel values of the label image are zero. Individual labels will subsequently be inserted into the label image to create the final GT label image containing all labels for all contoured objects from the given CT image.

### Segmentation (Contouring)

The network assists the operator in drawing contours for an object using underlying semi-automated segmentation techniques. A contour in MeVisLab is called a Contour Segmentation Object (CSO). Lists of CSOs for a given image can be saved and loaded. Once the user has created contours for a given object on several slices, the contours can be interpolated across those slices. The interpolation of contours across slices is the single largest time-saving step because it allows a 3D volume to be segmented by drawing contours on only a fraction of the slices as opposed to each slice. MeVisLab’s ability to assist in drawing contours and interpolate contours on any of the three standard multi-planar reconstruction planes (transversal, sagittal, and coronal) is one of the primary reasons it was chosen as the software tool to perform ground truth labeling.

### Labeling

All points within the 3D contoured volume can be filled in with a single label (mask) value. The label values used were the target IDs specified in the object/packing database. In the case of multiple objects of interest, the individual objects can be separately labeled and combined together into the ground truth label image.

### Region Growing

Region growing may be used to make the label more accurate. For example, if an object contains a hole in the center, it may be easier to contour and label the entire object (including the hole) and then use region growing with appropriate threshold values such that the hole is not included in the label.

### Add Individual Labels to Label Image

Once a label has been created for a single object, it can be inserted to the label image. The user then continues to label additional objects and add them to the label image.

### Save GT Label Image

Once all desired objects have been labeled and added to the label image, the user saves the label image in .raw format (16-bit unsigned short). Images can then be converted from .raw to .fits format using the *raw2fits* conversion tool distributed as part of the TO4 Software Tools (see the document entitled “TO4 Software Tools Spec”), and gzipped using *gzip* to produce a fits.gz format.

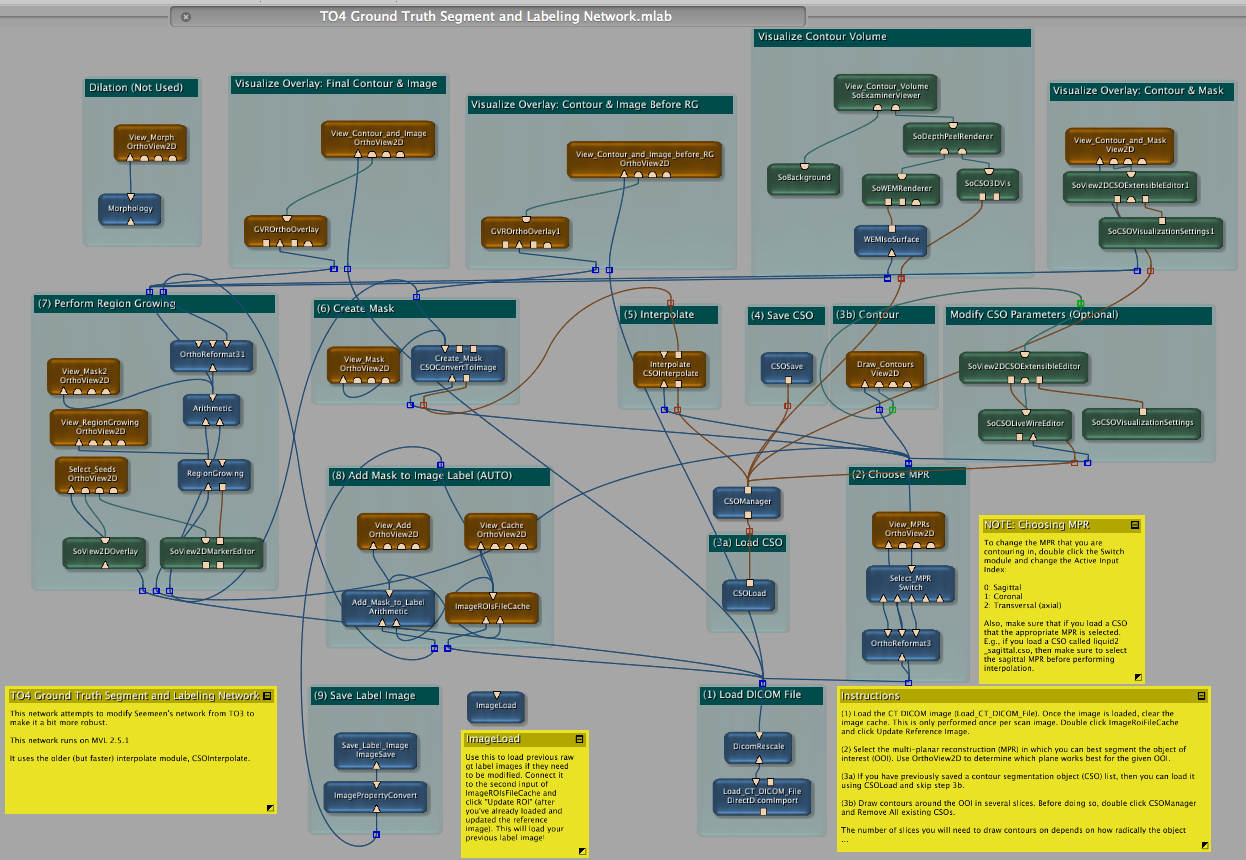


Figure : TO4 Ground Truth Segment and Labeling Network

## Instructions for Use

NOTE: The step numbers of these instructions coincide with the numbers of the groups with named headings in the network (see Figure 1).

1. Load a CT DICOM image using the “Load\_CT\_DICOM\_File” module. To load the CT image, browse to the location of the file and select OK. Then click the “Clear Log + Import” button.

**WARNING**: If trying to load an image comprising multiple single-slice DICOM files, MeVisLab seems to load the image in reverse order, i.e. the first slice is the last slice. To avoid this, load the image as a single DICOM file containing all slices of the image. To convert multiple single-slice DICOM files to a single multi-slice DICOM file, use the Tudor DICOM package [found here: <http://santec.tudor.lu/project/dicom>] for ImageJ.

Double click the “ImageROIsFileCache” and click “Update Reference Image.” This creates a zeroed out label image with the same dimensions as the input CT image. Labels will be added to this image, and the final aggregated label image will eventually be saved as the GT label image.

**NOTE**: Updating the Reference Image is only done the first time a CT image is loaded. Clicking “Update Reference Image” after labels have been added will zero out the image.

1. Double click the “Select\_MPR” module to select the MPR that will provide the most straightforward contouring of the OOI (0=Sagittal, 1=Coronal, 2=Transversal). To view the image in different MPRs, double click the “View\_MPRs” module, which will bring up an image-viewing pane, and then use the drop-down box in the upper left corner of the window to select a different MPR.

**NOTE:** You can change the window/level of the image by right-clicking on the “View\_MPRs” module and selecting “Show Window->Settings,” or you may use the right mouse-button on the image itself.

1. If you have previously saved a contour segmentation object (CSO) list, then you can load it using the “CSOLoad” module and skip to step 4.

Otherwise, double click the “Draw\_Contours” module to bring up the image. Draw contours around the OOI in several slices. Each contour is saved as a CSO and can be viewed and managed by double clicking the “CSOManager” module.

**NOTE:** Before beginning to draw contours for a new OOI, double click “CSOManager” and click “Remove All” to remove any CSOs from a previous OOI.

**NOTE:** The number of slices you will need to draw contours on depends on how radically the object changes from slice to slice. Generally you can contour every 5-20 slices.

**NOTE:** You can change the window/level of the image by right-clicking on the “Draw\_Contours” module and selecting “Show Window->Automatic Panel,” or you may use the right mouse-button on the image itself.

1. Save the CSO list for this object using the “CSOSave” module. This is done so that the CSO list may be loaded later without having to re-contour the image (i.e., in case MeVisLab crashes, or in case the image needs to be re-labeled).
2. Interpolate the contours by double clicking the “Interpolate” module and clicking “Apply.”

**WARNING:** Every so often, this step will cause MeVisLab to crash. From my experience, it will crash if

1. not enough contours have been drawn before interpolation, especially when the object changes radically (for example, at the beginning and end of an object, especially sheets)
2. contours are drawn poorly before interpolation (for example, a single contour that overlaps itself)

It is best to draw a few contours and then click “Interpolate.” If MeVisLab does not crash, you can save the CSO list using the “CSOSave” module and continue drawing contours. If at some later point during contouring of the OOI MeVisLab does crash, you now have a saved CSO list that you can load in step 3 instead of having to start over with the contouring.

**TIP**: For a single object with disparate components within a single slice, contour and interpolate the two components separately, but use the same mask value. This is to prevent the interpolator from trying to interpolate multiple components across slices.

1. Double click the “Create\_Mask” module to create the image mask (label). The "Foreground Value" field dictates the value of the label. Click “Update” to create the label.

**NOTE:** You can check how well the pre-region-growing label matches up with the OOI in the CT image using View\_Contour\_and\_Image\_before\_RG. In addition, View\_Contour\_and\_Mask allows you to determine how well the label lines up with the drawn contours, and View\_Contour\_Volume allows you to view the interpolated contours in 3D volumetric form.

1. Use the "Select\_Seeds" module to select one or more seed points in the OOI for region growing. Manually select lower and upper thresholds for the region growing algorithm using the “RegionGrowing” module.

**NOTE:** You can check how well the label matches up with the OOI in the CT image using View\_Contour\_and\_Image. If you’re not satisfied with the label, you can select different region growing thresholds, or you can start at step 3 and try drawing additional contours.

1. Add the current label to the aggregated label image by double clicking the “ImageROIsFileCache” module and then clicking "Update Roi." Once the image has been added, double click the “SoView2DMarkerEditor” module and click “Delete All.” This will remove all seed points in preparation for the next OOI to be contoured.

**NOTE:** You can view the aggregated label image by double clicking the “View\_Cache” module.

**REPEAT STEPS 2-8 FOR EACH OOI IN THE CT IMAGE.**

1. Save the aggregated label image using the “Save\_Label\_Image” module.

# Lessons Learned about applying MVL

## Online Support

MeVisLab does not have very extensive documentation, and so its learning curve is quite steep, especially to those with little computer programming and/or image processing experience. However, the community of users and developers that participate in the MeVisLab developer’s forum (<http://forum.mevis.fraunhofer.de/index.php>) has proven to be very knowledgeable and quick to help with any issue.

## Crashing during interpolation

MeVisLab is not a very stable program, and it may crash from time to time. Through extensive use of the program, it was empirically determined that the primary causes of the program crashing are

1. Not enough contours have been drawn before interpolation, especially when the object changes radically (for example, at the beginning and end of an object, especially sheets)
2. Contours are drawn poorly before interpolation (for example, a single contour that overlaps itself)

The **WARNING** in step 5 in the instructions reiterates this point and indicates that the user should save the CSO list often so as not to lose their work if the program does crash.

## Thin Sheets

Thin sheets (approximately <5 pixels, or 5mm, thick) are very difficult to segment in MeVisLab. This is due in part to the active contouring algorithm sometimes jumping over the sheet to the next nearest object. The presence of CT artifacts (streaking and shading) further complicates the process of segmenting thin sheets. Further, MeVisLab will sometimes crash during segmentation of these thinner objects (see Section 5.2). The best approach, though time-consuming, is to segment many slices (every 5-10) when segmenting thin sheets.

## CT Artifacts

The presence of metal in many of the bags causes severe streaking/shading artifacts. These artifacts can cause splitting of an object, and even cause portions object to become invisible. In these cases, region growing is of little use (since the object will have very high and very low CT numbers from the artifacts). Thus, the user should segment the object in as many slices as possible, especially those containing artifacts and take care to segment as close to the object as possible.

# References

[1] NIH, “itk-SNAP,” <http://www.itksnap.org/pmwiki/pmwiki.php>

[2] NIH, “Image J Image Processing and Analysis in Java,” <http://rsbweb.nih.gov/ij/>

[3] Philippe PUECH and Loic BOUSSEL, “DICOMWorks,”<http://dicom.online.fr/>

[4] Karimi, Seemeen. “Computer-Assissted Manual Segmentation Method for Grand Challenge,” May 18, 2010.

# Revision History

|  |  |
| --- | --- |
| **Version** | **Changes** |
| 1 | Initial revision |
| 2 | Modified based on feedback from Carl Crawford. Added more background. Added Lessons Learned section. |
| 3 | Added additional background and lessons learned. |
| 4 | Minor revision based on feedback from Carl Crawford. |