



Universitat
Autònoma
de Barcelona



**DEVELOPMENT OF A TOOL FOR THE CONSTRUCTION OF “GROUND TRUTH” FOR
COMPLEX COLOR IMAGES WITH TEXT**

Memòria del Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per

David Torne Berga

i dirigit per

Dimosthenis Karatzas

Bellaterra, 17 de Septiembre de
2009

Index

1. Introduction.....	3
The Problem – The Need.....	3
Objectives.....	4
Structure of the thesis.....	5
2. Planning and Viability Study.....	6
3. State of the Art.....	8
Color Text Extraction.....	8
Ground Truthing Methods.....	9
4. Ground Truth Specification.....	10
5. A Ground Truthing Tool for Color Text Segmentation.....	13
Requirements – Functionality.....	13
Implementation Strategy.....	14
Overview of the Application Developed.....	15
6. Results.....	27
7. Conclusions.....	29
Objectives Revisited.....	29
Lessons Learn.....	30
Future Work.....	30
8. References.....	31
9. Appendix.....	33
A Xml Glyph Definition.....	33
B Ground Truthed Images.....	35

Introduction

The Problem – The Need

In the present era of Internet, text information embedded in color images is more frequent than ever before. We can find search engines that explore a great amount of web pages every day in search for text information that in many cases cannot be retrieved as it does not exist in the textual part of the Web page, but in photographs or labels. An important problem yet to be solved and being explored by some researchers is how to efficiently extract text from complex color images [1].

For problems like text extraction, the non trivial task of performing statistical comparisons with pixel-level segmentation Ground Truth must be done [2] to help evaluate their work. Nevertheless, researchers often find that they do not have a Ground Truth to evaluate the performance of their algorithms.

Currently, there is neither any Ground Truth available to evaluate the algorithms of text extraction for complex color images nor any software to create efficiently the required ground truth information.

The problem addressed by this project is the current lack of Ground Truth information that would allow researchers to evaluate the performance of algorithms that extract text from images and the lack of a tool to create or modify that Ground Truth.

Our project will aim to solve both because we consider that having a Ground Truth (GT) is essential for the progress of the text extraction field when facing complex color situations. Having ground-truthed data sets representative of the complex color combinations that may occur in the real world is crucial to evaluate with high accuracy the performance of the mentioned algorithms and knowing their drawbacks is a way to improve them in the future.


Objectives

The goal of this project is to construct a software tool that allows the creation of complete pixel-level ground truth information of complex colour images. For that the software should support both the easy labeling of the pixels in any image that form a readable text as well as managing metadata and the ground truth structure overall.


To construct the software we chose to work with Web images because of their variability, but the software should be generic and easily applicable to other domains.

A parallel objective of the project is also to define a Ground Truth specification and corresponding file format to store the GT information.

We mark thus the following objectives to be accomplished to complete the project.

 To define a Ground Truth Specification for the Segmentation of Complex Color Text Images

- I. Review the State of the Art on ground truthing methods
- II. Review the state of the art on Color Text Extraction and establish their evaluation needs
- III. Review a representative set of Web images and define the information necessary to be stored in the GT file
- IV. Define a structure for the GT file

 To implement a Software for building GT data that

- I. Allows the loading and visualization of images and the corresponding GT data
- II. Comprises tools for labelling images at the pixel level
- III. Comprises tools for editing GT metadata

Structure of the thesis

At the chapter 2 the work plan of the project will be detailed along with the methodology used to do the development of the software. An analysis of the resources and the reasons to follow this schema will be justified.

The chapter 3 will explore the state of the art surrounding Ground Truthing techniques and we will emphasize the current needs in color text extraction for Ground Truthed datasets.

The chosen data structure of the Ground Truth for the color text extraction problem will be explained in chapter 4 as well as a review of basic functionality of its constituting files.

We explain with more detail all the requirements needed to develop the application with all the main decisions taken to meet them.

In this chapter 5 an overview of the architecture is shown explaining the main libraries and structure of the software.

To finish this chapter a full example of labeling one image will be portrayed highlighting the main benefits of using this software to quickly create Ground Truth sets for the loaded images.

In chapter 6 to explain the results we will show the ability that the software has to synthesize the most important information and the effectiveness in which we can develop data sets.

In chapter 6 we show selected examples of ground thruthed images and perform a study on the differencing of ground truthing images of varying difficulty.

Conclusions will be extracted in chapter 7 showing the objectives accomplished as well possible extensions for the project yet to be made.

Planning and Viability Study

The period time to complete this project was a year, starting in October of 2008 and with a completion deadline of July 2009 with a maximum delay of September 2009.

This project followed an iterative model of software development. Each of the iterations was planned to be completed in a batch of one or two weeks.

The first two weeks an analysis of the requirements was performed, followed by two months of dedication to the design and the definition of the ground truth specification, four more for the code writing, and the two last to run the tests.

Frequent weekly meetings were planned to inline to the characteristics of the iterative model to allow us to keep track of progress made and solve problems easily while keeping the global design unchanged [3].

The team comprised two stake-holders. One, the student, had an active role in all the phases while the director provided guidance on any aspect of the project in every phase.

The resources needed were a computer loaded with the 2005 edition of Visual Studio C++ and the help documentation during all the coding phases as well as Microsoft Office Suite.

The project was organized to be completed during the equivalent to 15 credits which translated in hours equals to 300 hours for the whole project.

The resources in men-hours have been organized to one student that will spend 3 credits worth of work from October to February and 12 credits from February to July, giving thus 8 hours per week on the first semester in labor days and 30 hours per week in the second.

The resources needed were a computer loaded with the 2005 edition of Visual Studio C++ and the help documentation during all the coding phases as well as Microsoft Office Suite.

Although all the software will be downloaded for free from the official University repositories, and the computer is the student's own laptop, the official cost to develop all the project accounting the hours of work paid by means of university credits would be 2050 € (assuming a Sony Vaio laptop 900€, the Student Microsoft Office Suite 2007 100€, the Microsoft Visual Studio 2005 Edition 200€, the Microsoft Professional Project 2007 400€ and 15 credits university credits of student work 450€).

Id	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	Data analysis	16 días?	mié 26/11/08	mié 17/12/08	
2	Sample obtain	6 días?	mié 26/11/08	mié 03/12/08	
3	Plataform analysis	11 días?	mié 03/12/08	mié 17/12/08	
4	Explore algorithm and state of art	6 días?	mié 03/12/08	mié 10/12/08	
5	Document Presentation	6 días?	mié 10/12/08	mié 17/12/08	
6	Project Design	32 días?	mié 17/12/08	jue 29/01/09	
7	Data structure design	6 días?	mié 17/12/08	mié 24/12/08	4
8	Platform design	6 días?	mié 24/12/08	mié 31/12/08	4
9	Algorithms design	6 días?	mié 07/01/09	mié 14/01/09	7
10	Extension Algorithms design	5 días?	jue 15/01/09	mié 21/01/09	9
11	Residign all project	6 días?	jue 22/01/09	jue 29/01/09	10
12	Code Implementation	130 días?	vie 30/01/09	jue 30/07/09	
13	Xml reading and visualization	6 días?	vie 30/01/09	vie 06/02/09	11
14	Test	7 días?	lun 09/02/09	mar 17/02/09	13
15	Xml edition functionality	6 días?	mié 18/02/09	mié 25/02/09	14
16	Test	7 días?	jue 26/02/09	vie 06/03/09	15
17	Xml save and files management	6 días?	lun 09/03/09	lun 16/03/09	16
18	Test	7 días?	mar 17/03/09	mié 25/03/09	17
19	Image and xml pixels visualiztion	6 días?	jue 26/03/09	jue 02/04/09	18
20	Test	7 días?	vie 03/04/09	lun 13/04/09	19
21	Zooming options and functionality	6 días?	mar 14/04/09	mar 21/04/09	20
22	Test	7 días?	mié 22/04/09	jue 30/04/09	21
23	Pixel editing functionality	6 días?	vie 01/05/09	vie 08/05/09	22
24	Test	7 días?	lun 11/05/09	mar 19/05/09	23
25	Region editing tool (Flood Fill)	6 días?	mié 20/05/09	mié 27/05/09	24
26	Test	7 días?	jue 28/05/09	vie 05/06/09	25
27	Toolbars visualization and behaviour	6 días?	lun 08/06/09	lun 15/06/09	26
28	Test	7 días?	mar 16/06/09	mié 24/06/09	27
29	Extension of region algorithms	6 días?	jue 25/06/09	jue 02/07/09	28
30	Test	7 días?	vie 03/07/09	lun 13/07/09	29
31	Project Functional test and debug	6 días?	mar 14/07/09	mar 21/07/09	30
32	Efficiency and Beautification	7 días?	mié 22/07/09	jue 30/07/09	31

Figure 1: Project plan made with Microsoft Project 2007.

State of the Art

Color Text Extraction

In computer science a typical way to evaluate the performance of an algorithm is to compare the output produced by the algorithm on a given problem with the output already known to be correct for that problem.

The output or solution known to be correct of any problem is called Ground Truth of the problem.

In the field of document analysis, Ground Truthed data sets are very important resources to verify the correct behavior of the applications. Quite often it is necessary to manually define the Ground Truth data for a given problem to be able to evaluate the algorithms [4].

When using a Ground Truth, the evaluation of the results is made by making defined comparisons between the results of the algorithms and the correct Ground Truthed data sets. The error of these comparisons is measured by weighting the error of the different properties considering the influence they have to the correctness of the output.

When creating a Ground Truth a large and representative number of data sets are usually necessary to represent all the scope of the problem to be solved. In these cases a semiautomatic way to generate data sets is usually employed to avoid an unnecessary waste of time and human resources.

In the case of color text extraction we usually find occasions where no more than ten or twenty Ground Truthed images are used to evaluate the results [5].

As researchers do not have a well defined Ground Truth to evaluate the algorithms of color text extraction they frequently use alternative methods, for example making a visual check of a small amount of images as this example of text extraction from posters of civil war [6].

Ground Truthing Methods

Different Ground Truthing frameworks exist to create Ground Truthed data-sets in problems similar to the color text segmentation.

Most of the Ground Truthing applications and benchmarking frameworks share some common properties and needs that this project will include to perform the best evaluations.

When creating a Ground Truthing tool there is a trade of between fastness and accuracy, as both, a grand quantity of sets and high quality are needed.

A great quantity of data sets is necessary to assure that the Ground Truth covers most of cases. A fast Ground Truthing software is thus a must to achieve that need.

On the other hand the efficiency must be accompanied by the capability to permit the user to define information with the accuracy what he considers appropriate. The GT data is considered the absolute goal for automated processes to achieve. So making a bad Ground Truth heavily misleads the code and wastes enormous efforts.

These issues have been revised in problems like symbol recognition performance evaluation [7] and Ground Truthing for image segmentation [8], but not specializing in our concrete case of color text extraction.

Other ground truthing creation software and studies for layout analysis [9] have already obtained great results highlighting the importance this project also has to give to the design a simple descriptive Ground Truth format to reflect most of complexities this problem can find in the real world.

Ground Truth Specification

The first problem we faced was choosing the way to describe the text regions of images. The first doubt was if to store the pixels as connected component manner [10], proved useful at the binary labeling task, or to store all the pixels individually. This meant decide if to store the pixels of a glyph using a more compact structure or another more explicit but easier to manage for any software.

The decisions taken was chose the individual pixels description to give more simplicity and portability to the ground truth.

Having all the pixels of a glyph allows more precise, easy and compatible Ground Truth use for any algorithm than any other method considered. Our effort at this point was to store as less quantity as possible but specify every x y point and its properties in our definition. This chose is easier for the project to adapt the algorithms that have been already foreseen.

By analyzing the more difficult color web text images, the conclusion arrived was it was needed a global description that best described them all.



Figure 2: Normal text image



Figure 3: Binary Image by algorithm

In figure 2 and figure 3 it can be seen for example how the overlap of letter can make difficult for an algorithm to determine the difference. Hence in our specification we should support the same pixel (or group of pixels) to belong to multiple characters in order to deal with overlapping cases.



Figure 4: "O" as symbol

As seen in figure 4, in some occasions letters must be inferred from the rest of the word. In these cases it is usually difficult even for the human user to decide whether a pixel belongs to the text or the background. This is a peculiar case, as there is no pixel-level ground truth available, and the only way to label such pixels would be to assign

both labels (text and background). We do not consider such cases for the current specification.

When designing the specification and the corresponding file structure for the ground truth we considered it important to maintain compatibility to existing ground truth structures. Through collaboration with the Pattern Recognition and Image Analysis group at the University of Salford, we reviewed their work within the EU project IMPACT [11], and devised a XML structure that is an extension to the specification developed within the project.

When designing the details, we realized it was needed a distinction between the different parts of a glyph. We have included hence different regions: the core, the outline and the shade part of any glyph.

To have a distinction between these parts helps to perform specific evaluations that need to know in advance to know where the exact pixels form the outline, the shade or the core of a text are, e.g. for aliasing evaluations.

Thus a careful study for the xml design guided by the main requirements of not to include too much information but to be descriptive enough to perform a good analysis of the algorithms of text extraction has lead to the following final specification.

In any Ground Truth instance *three* different xml files will be used to produce a full description of the text of any image (this was necessary to maintain compatibility to the IMPACT structures). All three XML files along with an image will define the document instance for our application.

gt-[name].xml

“gt” stands for *Ground Truth* and will be used as the main file to open an existing document and to link all three files and document path. The structure of this file was already defined within IMPACT.

pc-[name].xml

“pc” stands for *page content* and will describe the content of the document hierarchy ordered in text regions, text lines, words and glyphs and describing features like colors, fonts type, positions or content among others. The structure was already defined within IMPACT.

gl-[name].xml

“gl” stands for *glyph* and it is our contribution. This file expands the information for the glyphs given by the “pc” xml in order to include all the labeled pixel coordinates of the image that form the glyph. The pixel editing part will mainly work with this file updating its content.

As explained above, the “gt” and “pc” files which define the global structure of our data was defined by the University of Salford for the IMPACT project The main

contribution of this project has been focused on the definition of a more detailed, pixel-level description of a character or glyph.

The text of an image is specified by the set of x,y points that form every glyph, each glyph having eight different labeling combinations depending on whether the pixel is or not in the core of the glyph, on the outline and the shade region.

The complete XML definition of the ground truth as well as samples of the three files can be found in appendix A.

In figure 5 a sample of a gl file resulting from the labeling of a text illustrates the main structure of the glyphs description and the one in which most of the Ground Truthing software will describe the text location of the images.

```
<?xml version="1.0" encoding="utf-8" ?>
- <pcGts xmlns="http://schema.primaresearch.org/gt/gts/glyph" xmlns:xsi:
  xsi:schemaLocation="http://schema.primaresearch.org/gt/gts/glyph ht
  20090126T000000000-0000">
- <pcMetadata>
  <pcCreator>David Torne</pcCreator>
  <pcCreated>2009-9-17T00:58:42</pcCreated>
  <pcLastChange>2009-9-17T201:204:23</pcLastChange>
  <pcComments>By Torne v1.0</pcComments>
</pcMetadata>
- <glyph id="id4" wordRefId="id3">
- <coordGroup id="id8" no_coords="5" core="-1" outline="0" shade="0">
  <p x="82" y="30" />
  <p x="83" y="30" />
  <p x="83" y="31" />
  <p x="84" y="29" />
  <p x="84" y="32" />
</coordGroup>
- <coordGroup id="id9" no_coords="4" core="0" outline="-1" shade="0">
  <p x="83" y="28" />
  <p x="83" y="29" />
  <p x="84" y="28" />
  <p x="84" y="30" />
</coordGroup>
- <coordGroup id="id10" no_coords="1" core="-1" outline="-1" shade="0">
  <p x="84" y="31" />
</coordGroup>
- <coordGroup id="id11" no_coords="4" core="0" outline="0" shade="-1">
  <p x="86" y="28" />
  <p x="86" y="29" />
  <p x="86" y="30" />
  <p x="86" y="31" />
</coordGroup>
- <coordGroup id="id13" no_coords="4" core="0" outline="-1" shade="-1">
  <p x="85" y="28" />
  <p x="85" y="29" />
  <p x="85" y="30" />
  <p x="85" y="31" />
</coordGroup>
</glyph>
</pcGts>
```

Figure 5: Sample gl file from the image HelpWhales.jpg

A Ground Truthing Tool for Color Text Segmentation

Requirements – Functionality

The phase of the requirements analysis concluded this features as the most important to guide the development of the project.

An easy application to create a Ground Truth for algorithms that extract text in color images must be develop as well as some samples to provide researches with the first Ground Truth set of data that show them the results.

The application must be able to allow researchers to easily load images and select the pixels of the image that pertain to the text region and save or load this information in a predefined xml containing the coordinates of the text as well as other description of the document properties.

The software must be user friendly to allow at all moment the effectiveness of the job of facilitating a fast manual selection and modification of the pixels that pertain to text region for any given image.

The code must be scalable to permit future extensions or modifications.

The xml definition must be simple but descriptive enough to be useful for the text extraction algorithm analysis.

Implementation Strategy

Based on the project requirements and a developer choice of using C++ the project will make use of MFC library and will be developed using Visual Studio 2005 platform.

The application will be composed of an xml framework and an image framework working both under the common MFC framework to avoid complexity of a high correlation.

The designing has been divided in three phases.

The first is the design of an adequate MFC framework to be able to include both, the image framework and xml framework.

The image framework will used to visualize the image, visualize and edit the select the pixels coming from the xml framework and save back the changes to xml framework. This framework will include different tools related with the edition and visualization of the pixels.

The Image part must be able to show the working image, selected and deselect, perform zoom operations, invert pixels and automatically select regions the image.

The framework will able to load different images formats (jpg, bmp, png, gif and tiff), sizes, and color bits per pixel (1, 4, 8, 16, 24 or 32) performing fast pixel operations.

The Xml part will be able to read and save the xml files, visualize and easy change its content.

All the parts will use them own selected libraries and will perform differentiated actions.

As an example when designing the xml part different xml libraries have been considered. After balancing the need of efficiency, the documentation available and the portability, it has been concluded that the MSXML 6.0 library of Microsoft will be the main file processing library.

The main functionality of the XML library will be to show and process the previous three files in a tree form and to be able to automatically process, add, delete or modify the selected items and to be able to store the changes back into the files.

The main nodes to be shown in the application will be the text regions, text lines, words and glyphs parsed from the files.

Overview of the Application Developed

As a working sample, the process followed for labeling one image will now be showed with its main steps.

When we start, the first option is to choose an image to load as showed in figure 6.

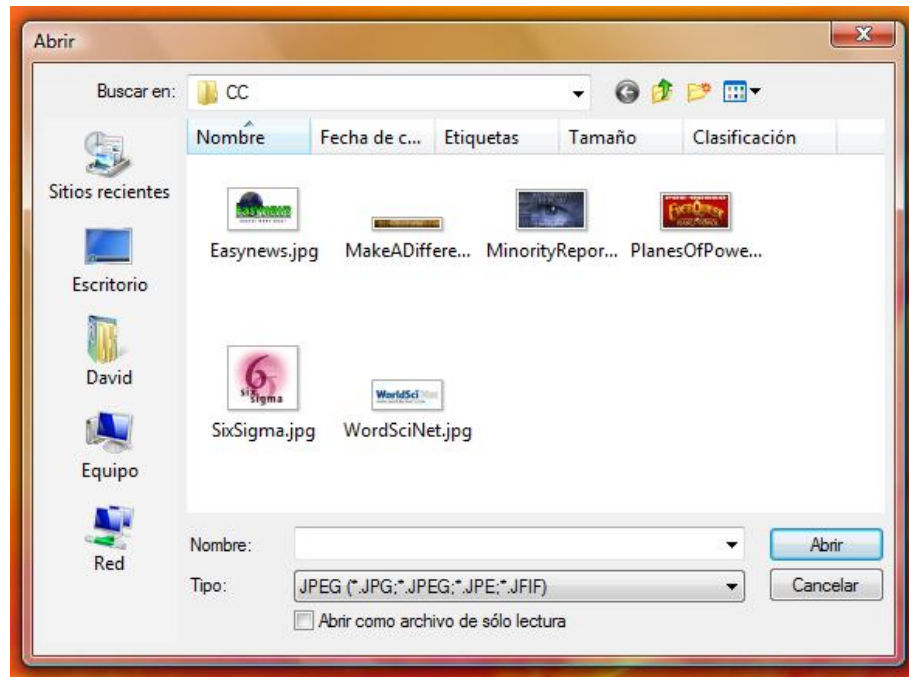


Figure 6: Starting window for a new image.

Once loaded we will start by starting to form the xml structure, so a right click to the xml tree will start adding the first text region definition at the right part of figure 7.

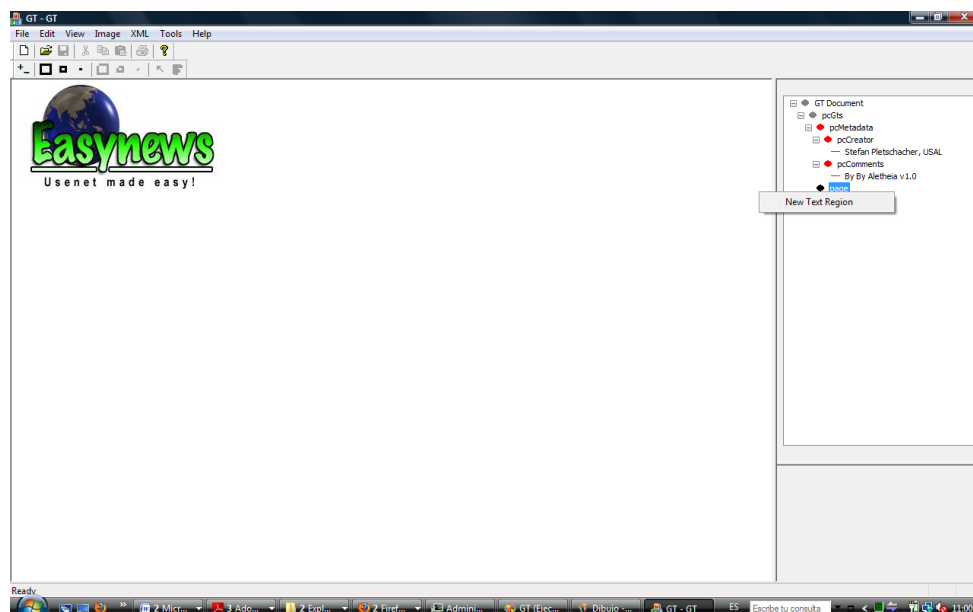


Figure 7: Creating a text region node.

Once created the first node, we will continue right clicking on the child nodes to create the number of text lines, and words that better define the text of the image.

When created and labeled all the words, we will take advantage of the option of automatically create its glyphs from a labeled word as shown in figure 8.

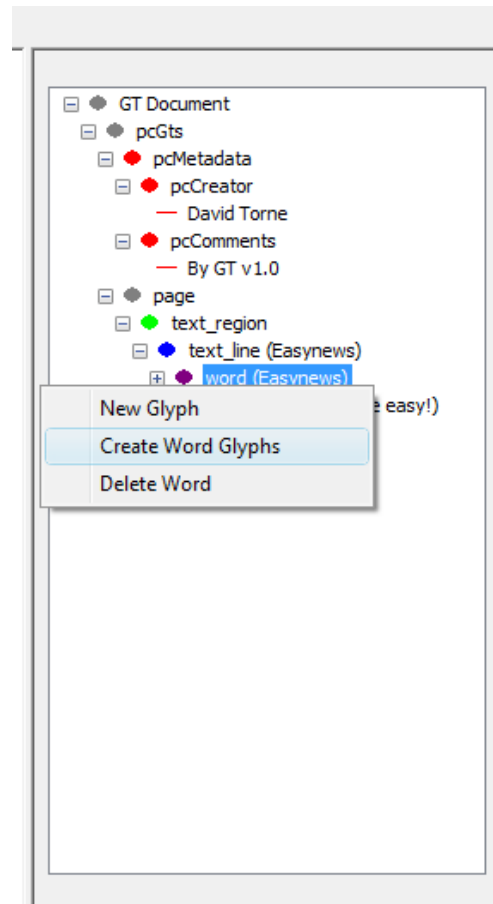


Figure 8: Automatically creates all the glyphs based on the word label

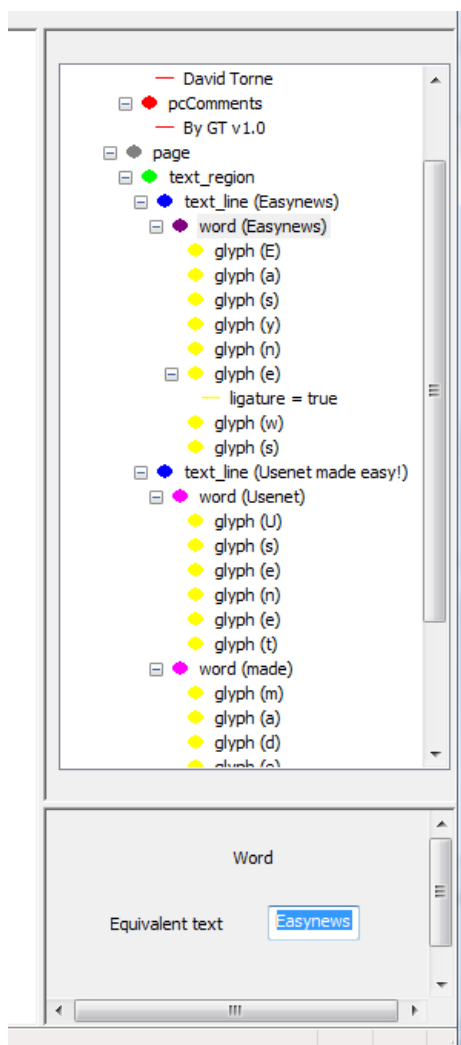
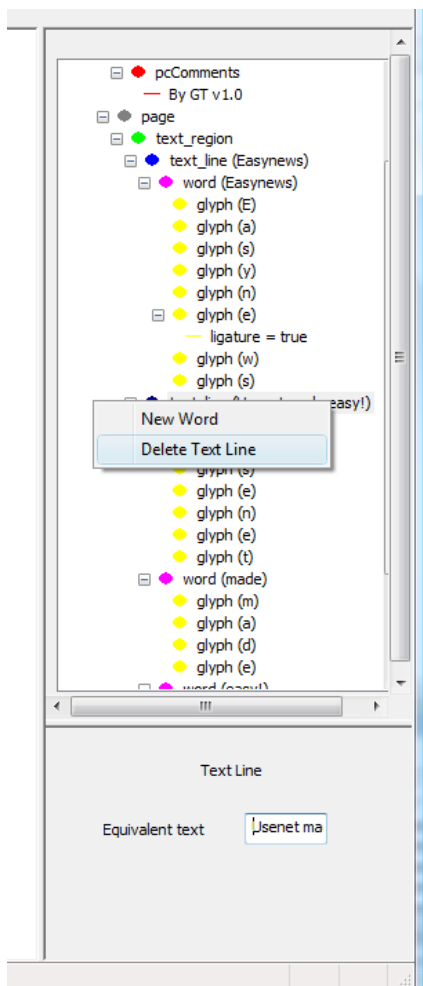


Figure 9: Complete edited xml Data.

While developing the property form and for efficiency reasons a dynamic update of the node text in the view occurs any time information changes in the property forms to allow a fast visualization of the content.

If the user considers that two characters are a glyph the creation of an only a glyph will be used, understanding for a glyph a pictograph or hieroglyph used to represent one or more characters having the latter case the ligature property.

With all the mentioned functionality we can edit easily the GT xml tree the figure 9.



In cases of user errors xml tools like the deletion of nodes have been done as shown in figure 9 which allows the instant deletion of the whole branch.

In case whole branches were accidentally deleted as in figure 10 we can select undo the xml state, as well as a redo later.

Figure 9: Delete option in the context menu.

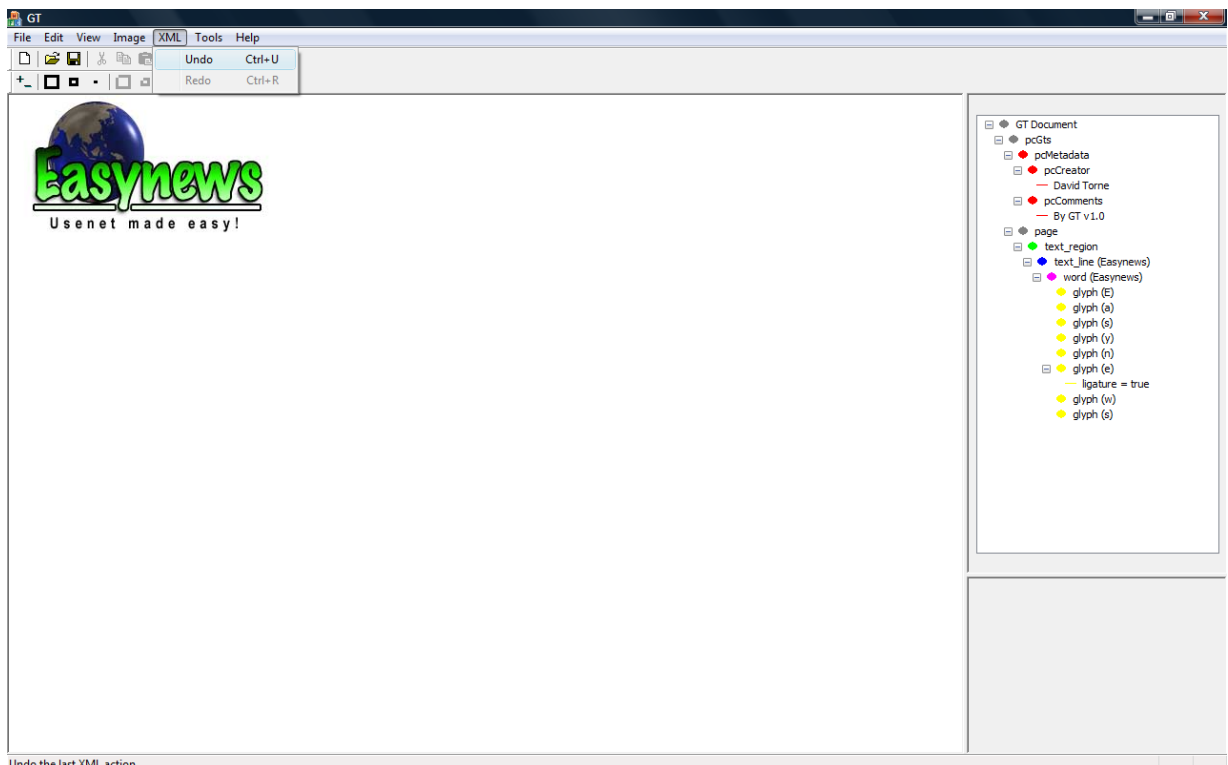


Figure 10: Option to undo a recently deleted xml branch, including the pixels it may contain.

Now that the xml structure is well defined we will start selecting the pixels from the image that form each glyph.

By selecting the zoom icon we will fix a size for the image as in figure 11 and 12 and we will be able to zoom in or zoom out to any point using mouse clicks or the wheel move

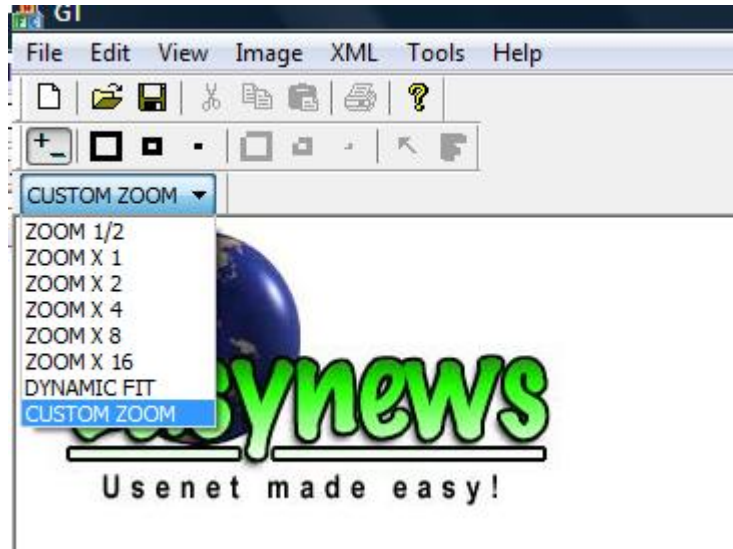


Figure 11: Different zooming options.

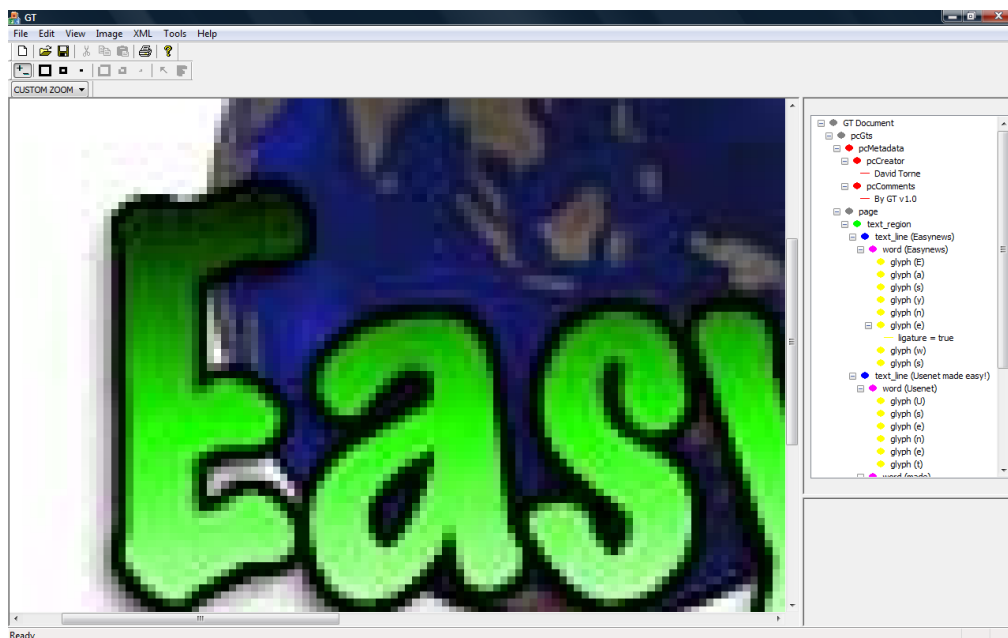


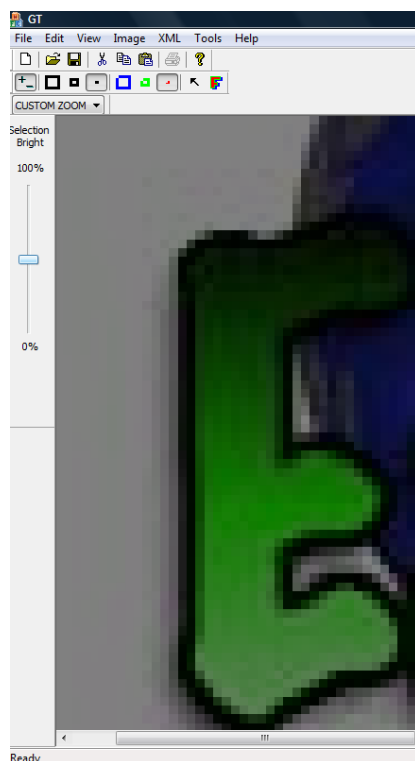
Figure 12: Zoom In to the E letter using the mouse.

We now will start editing the pixels of every glyph, so we can begin selecting the first letter from the word “Easynews”. A click on the E letter in the xml tree will allow the use of editing options (see toolbar in Fig 13) by clicking the editing icon. We can start

with the core part of the glyph as in figure 13.



Figure 13: The colored icons activate the edition mode of the clicked region.



The selection of the editing mode automatically darkens the image to increase the contrast with the selected pixels and to avoid confusion between the editing color and the image one.

At the same time a scroll bar appears at the left side (see in figure 14) that lets us change the contrast between the image and the selection pixels.

Figure 14: Bright Bar allows changing the contrast.

Now we have at our disposal tools to start editing either at pixel level



or at



region level.

Since we try to edit fast, we will start to see if we can select the whole letter core using the region tool. The region tool is based on a flood-filling algorithm [12].

If we click the region icon, a new bar at the left will appear the allow us to set the sensitivity of the selection as in figure 15. We can see that while activated, all the bars can be hidden or moved at our will for convenience as shown in figure 16.

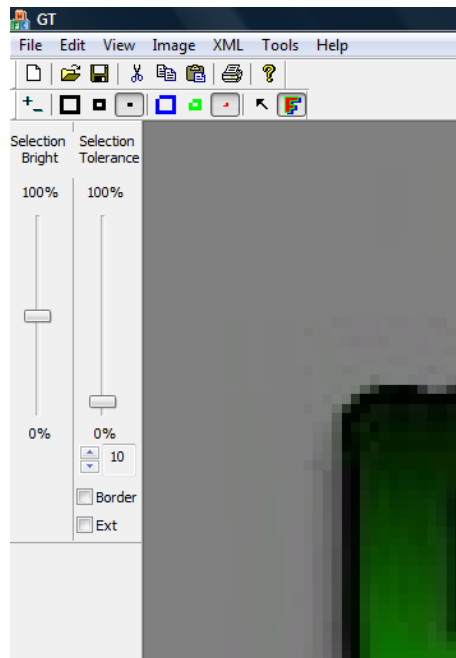


Figure 15: Region Sensitivity bar.

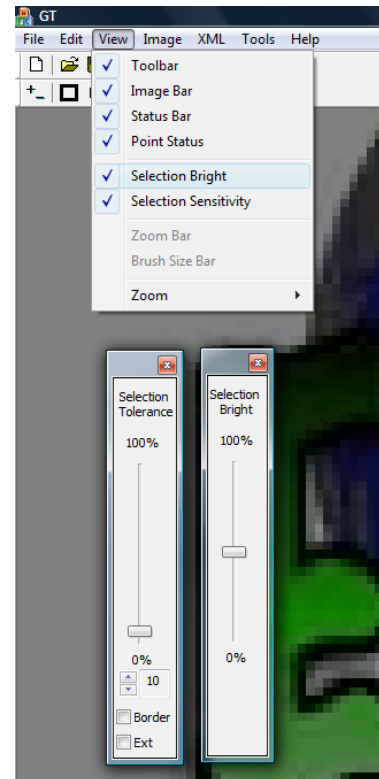
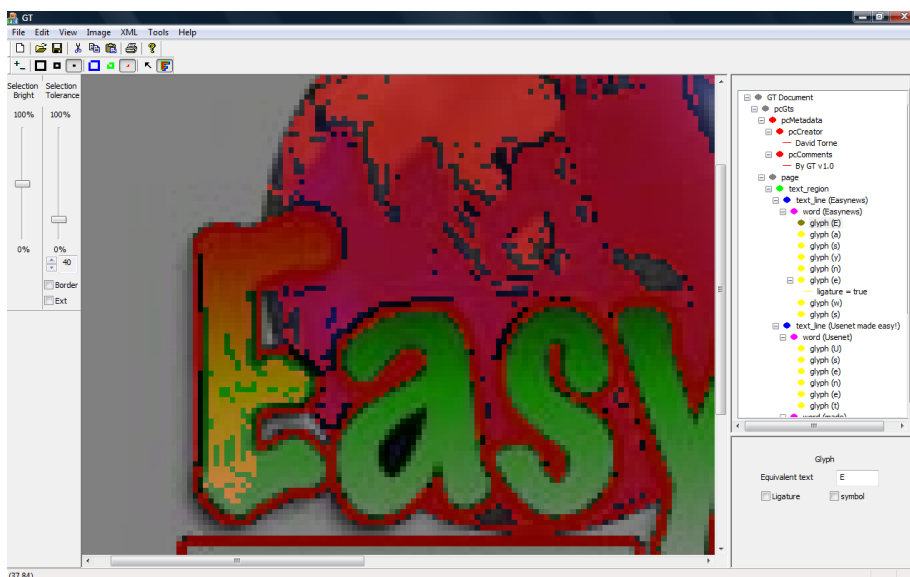


Figure 16: Custom bars view options.

We can start now clicking on the E letter with a colour distance thresholds of , say 40. Since the algorithm used is based on the color distance, we have to be careful not to make it too permissive or too restrictive because if so the underlining flood filling



algorithms can surpass the glyph outline as in figure17 or fill just a small part.

Figure 17:
Selected region surpassing the letter E with the pixel position clicked at bottom left.

If the selection surpassed the limits we can press the undo button to delete the last action. We can choose between pressing menu -> Edit->Undo or Ctrl + Z.

As we can see by changing the transparency level when diminishing the contrast, the E letter it is not well delimited from the upper part, meaning that the threshold must be changed in order to select just the letter.

Here different editing techniques can be done. We can select first the upper part and then the lower with a different sensitivity, or we can choose a low tolerance and press the Ctrl key while clicking the mouse to select repeatedly. Both will end up filling the letter as it is shown in figure 18.

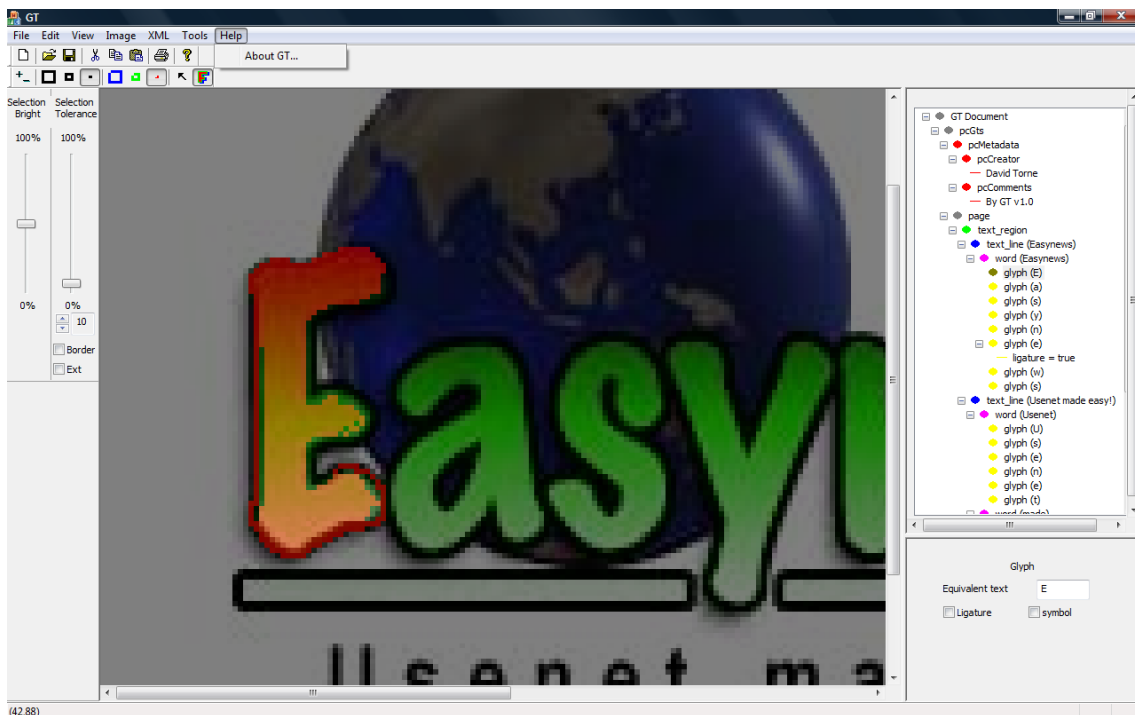


Figure 18: Selection of core of letter E.

In the above sample an option marked as EXT has been used. In the left bar we can see this box option, which performs the selection using a different algorithm that works better in avoiding surpass the limits of the regions. It works in a more local approach to measure the color distance versus a more global one of the default flood fill. The drawback is that when there is a well defined region it selects with less density than with the global option, not EXT. We can see the difference in figure 19 when the same pixel is clicked at the same distance using the option of Ext (Extended option) and without.

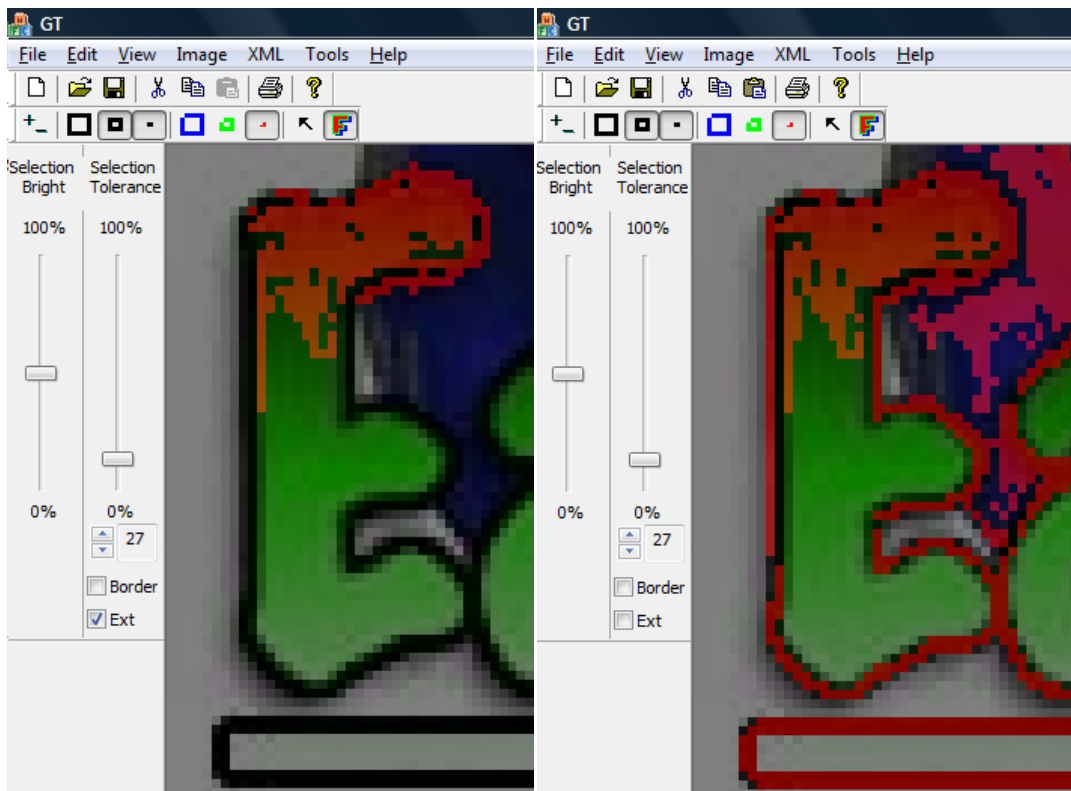


Figure 19: Results of Region Selection at the point (34, 68) with Ext Option or without.

At this stage and when finished with the letter selection we may want to perfect it by editing manually some pixels.

For this job we can select the pixel tool represented by an arrow. Automatically a new menu box will appear to select the size we want for the selection tool shown figure 19.

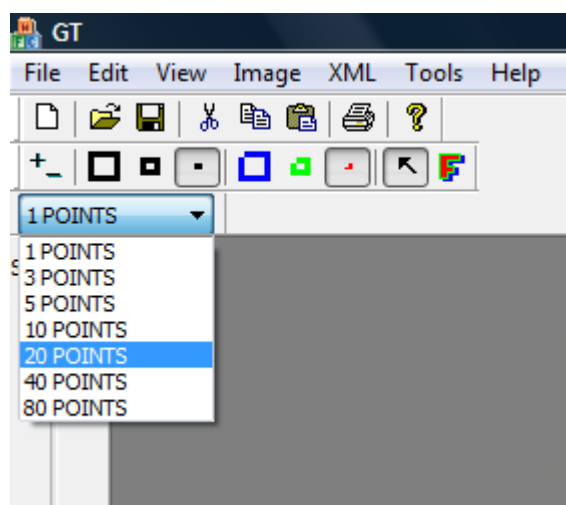


Figure 19: Size Box of Pen Tool.

By pressing the left mouse button over the picture now we are able to select manually over a point like figure 20 or deselect it like in figure 21.



Figure 20: Selecting size 1.



Figure 21 Deselecting size 3.

When using the editing mode we may want to preview before clicking the mouse which pixels will be selected or deselected. Pressing the Control we will see the pointer and size of the pen in white as illustrated in figure 22.

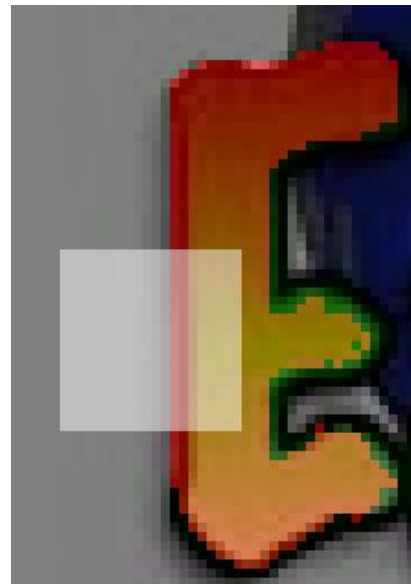





Figure 22: Visualizing the pointer without editing.

All the changes are automatically stored in the xml definition so we do not need any explicit action to save the changes besides saving all the work before exiting.

The next job is to edit the outline part of the glyph. We will start selecting the outline editing mode . Note that any editing mode  will only be available if we select first a glyph from the xml tree. In our case we have selected the E glyph. At any time we have the option of see or hide the view of the core, the outline or shade by using the menu or by marking any of the icons.  The view of coincident regions will result in viewing the combination of the colors though while editing only one region changes.

In our task of editing the outline we are using the same editing tools used for the core but now we can take advantage profit of an option of region tool labeled “Border”, marked in figure 23, created to help editing the outline. This option will select or delete only the border pixels of a predefined region.

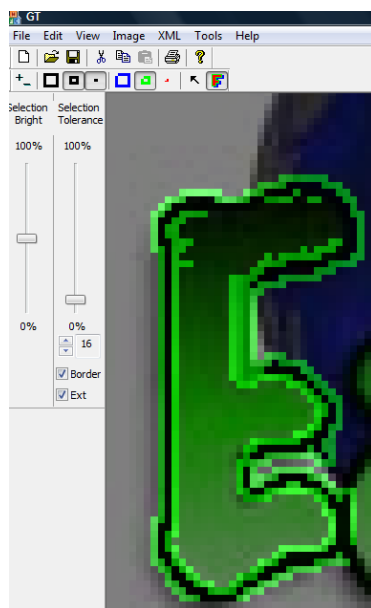


Figure 23: Use of the border option.

Fast techniques that can be used to label the outline in difficult color cases is selecting the border option with a small tolerance and cleaning the unwanted marked pixels and to use the previous tools to select or deselect regions.

In some occasions it may be useful to use the inverse operation “Invert” located in the menu under the Tools submenu option. This will invert the pixels of region being selected without affecting any other region.

After editing the three parts and viewing the result we obtain the completion of the glyph shown in figure 24.

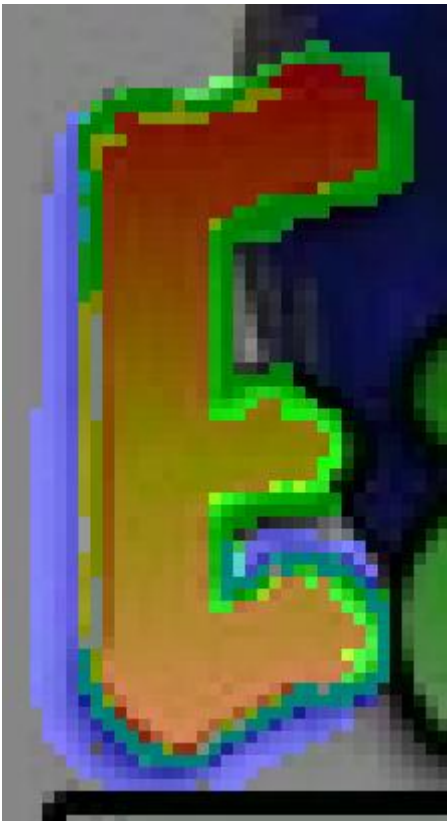


Figure 24: E glyph edited and viewing the three regions

The rest of the glyphs in this sample are easier to edit due to the fact that they are better delimited and tools like the region flood fill will select all the pixels in only one click.

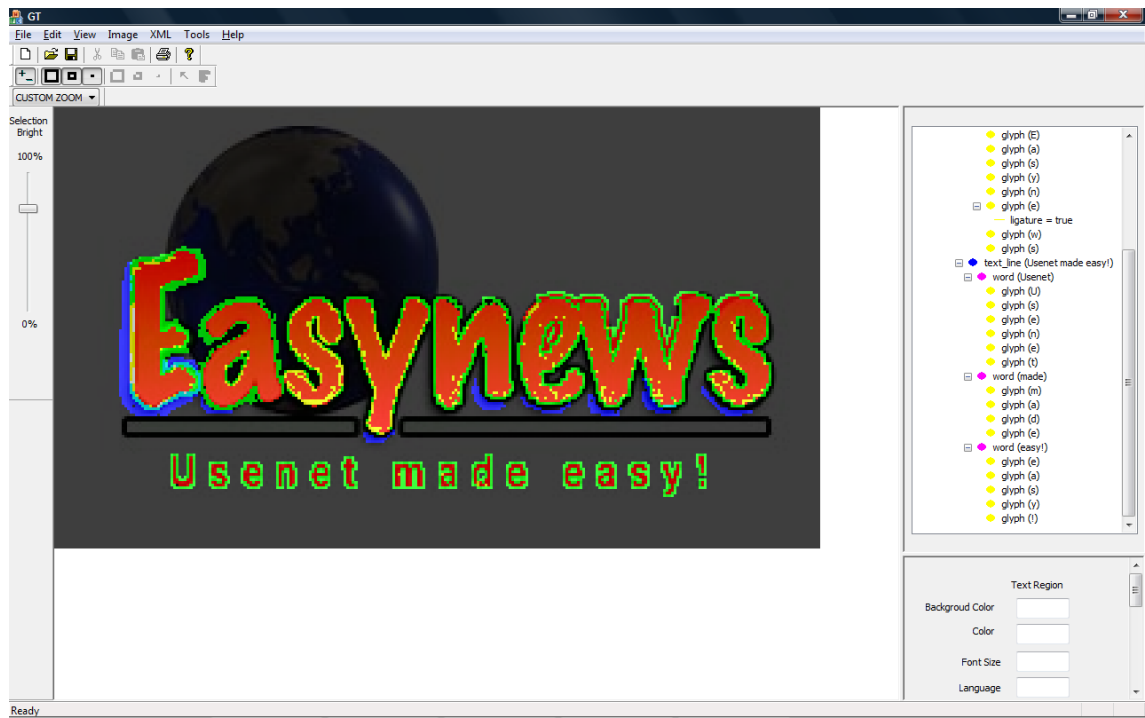


Figure 25: Complete edition of the image easynews.jpg

Options like Cut, Copy or Paste of whole selections into other glyphs or between glyph regions may be useful also for an easy trespass of content or a fast erase of a whole selection (Cut). The temporally content will disappear if a node that it is not a glyph is selected.

As shown in figure 26 if we select any node from the xml tree we will see the content of all its children (see Figure 26).

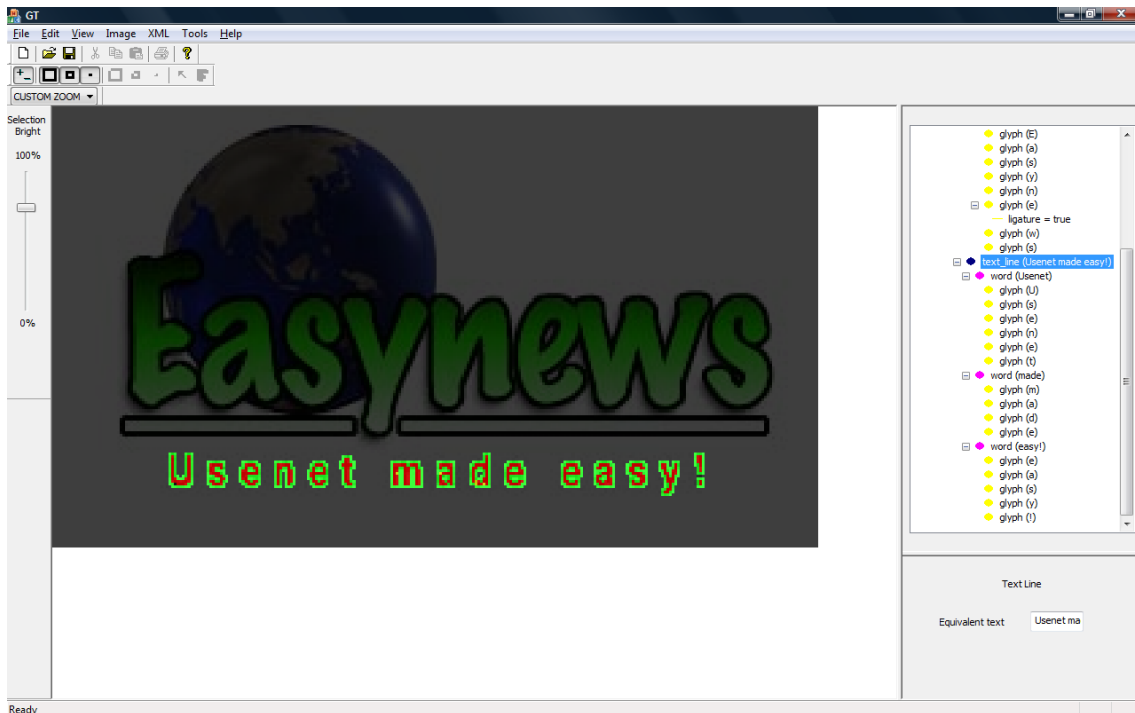


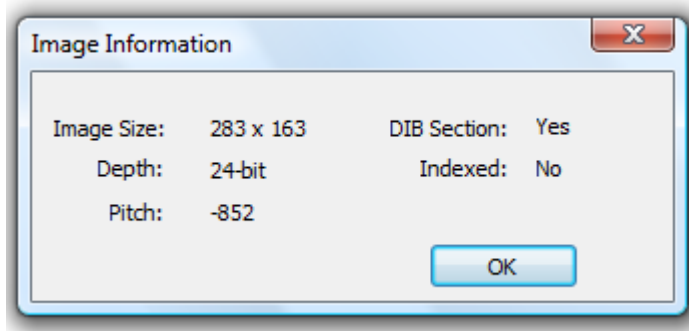
Figure 26: Automatic view of the content of a text line

Finally we will save all the content with any name and three files will be stored with that name and their prefixes.

Common operations like asking before closing, Save, Save As distinctions or recent docs shortcuts are allowed as well as actions like printing the selection or preview the print result.

Additional information related to the image loaded can also be viewed by selecting menu->Image->Info

Figure 27: Image Information



This document can be opened and modified at any time by opening the main gt- file.

Results

We have created a first sample data-set that contains 25 images to test the efficiency of working with the developed software.

For all 25 images we have produced ground truth information.

In the experiments, the learning of new techniques as repeated flood fill runs (flood fill + Ctrl) with a low tolerance for difficult glyphs has proved that once acquired an expertise level the time of editing is reduced to a factor a half to two thirds.

As not all images present the same difficulty, there are variations on the time it takes to ground truth an image. We have identified the following three main problems that can occur: (1) the presence of very small (1-pixel wide) anti-aliased glyphs, (2) small color differences between the text and the background and (3) glyphs that comprise a range of colors. Examples of the above cases are shown in figures 28 to 30.



Figure 28: Flood fill working on too *small glyphs* ("n").



Figure 29: Flood fill working on a glyph with the *same color range* than *background*.



Figure 30: Flood fill on a glyph containing *variable color shades*

Based on our experience at the time of labeling images, we have noticed a learning curve, beginning with the first time labeling taking 10' for a mid difficult image (see figure 32), 14' for a difficult image (see figure 33) and 6' for an easy (see figure 31).

To define the difficulty of the image we have weight the previous three difficulty properties to every glyph named: *small glyphs*, *small color difference as background*, *variable colors in the glyph*.



Figure 31: An image with easy characters

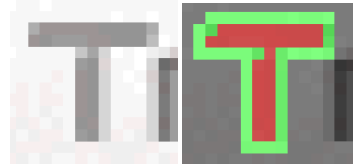


Figure 32: The letter “T” took 25 sec to be labeled



Figure 33: An image with mid difficult characters



Figure 34: The letter “W” took 40 sec to be labeled



Figure 35: An image with difficult characters

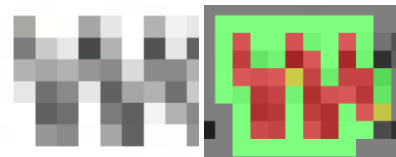


Figure 36: The small letter “w” took 1' 7" to be labeled

For the full list of Ground Truth images see appendix B.

Conclusions

Objectives Revisited

All the objectives of the project set at beginning have been successfully accomplished. This is the revisited list marked with its accomplishment.

✚ To define a Ground Truth Specification for the Segmentation of Complex Color Text Images

- ✓ Review the State of the Art on ground truthing methods
- ✓ Review the state of the art on Color Text Extraction and establish their evaluation needs
- ✓ Review a representative set of Web images and define the information necessary to be stored in the GT file
- ✓ Define a structure for the GT file

✚ To implement a Software for building GT data that

- ✓ Allows the loading and visualization of images and the corresponding GT data
- ✓ Comprises tools for labeling images at the pixel level
- ✓ Comprises tools for editing GT metadata

We can conclude that the project has been a succeeding, though there has been a delay of two months.

Lessons Learnt

The main lessons learnt from the project after finishing and to consider if it would be redone is that when problems turn out to be very time consuming or difficult to get into the desired result, they can generally be fast tracked by asking for a hint to the director or solving them in other moments of the development.

Another issue that I want to point out is that when designing a project the time of execution matted. It means that though an only function may solve the problem and may not take very long if you do it with one structure or another, when executing all the program, the slowness of the functionality is the addition of all the bad programming practices and hence difficult to speed up with small tweaks. The next time I would try to make more simply and fast taking into account the scalability.

Regarding to the code

, to review better library alternatives maybe would have saved me a lot of worries inherent to its own complexity. The MSXML library has been a direct choice for being the official Microsoft XML library but nevertheless the unnecessary complexity of this library requires a lot of efforts of debugging and study. Though I looked to some alternatives, now I would consider searching better.

When making a program to solve a problem it is very important to understand exactly what you are solving, sadly I think I just understood most of its complexity and details at the end. Definitely understanding very well what you are solving defines in great measure the succeeding of the project, at least regarding to the Ground Truthing creation.

Future Work

Future work devised for the project is complementing the ground truthing tool with an evaluation framework to perform detailed comparisons between ground truth and algorithms outputs.

Another perform is adding more software options to assess more directly the progress of the automatic color text segmentation algorithms.

In addition, we would like to develop a full set of Ground Truthed images large enough, and representative of the real world, to create a public color text segmentation Ground Truth database.

References

[1] D. Karatzas and A. Antonacopoulos, "Text extraction from web images based on split-merge segmentation method using color perception", *International Conference on Pattern Recognition*. (2004), pp. 634–637

[2] Leon Todoran, Marcel Worring, and Arnold W.M. Smeulders, "Data GroundTruth, Complexity, and Evaluation", *Measures for Color Document Analysis, Proceedings of the 5th International Workshop on Document Analysis Systems V, Lecture Notes In Computer Science* (2002) Vol. 2423, pp. 519 - 531

[3] Johansson, P., Degerstedt, L., Jönsson.: "Iterative Development of an Information-Providing Dialogue System.", In: *Proceedings of 7th ERCIM Workshop*. (2002)

[4] Thomas Strecker, Joost van Beusekom, Sahin Albayrak, Thomas M. Breuel. "Automated Ground Truth Data Generation for Newspaper Document Images.", *10th International Conference on Document Analysis and Recognition July 2009*

[5] K. Sobottka, H. Kronenberg, T. Perroud, H. Bunke, "Text extraction from colored book and journal covers", *IJDAR* (2000) 2: 163-176

[6] Antonio Clavelli, Dimosthenis Karatzas, "Text Segmentation in Colour Posters from the Spanish Civil War Era", *10th International Conference on Document Analysis and Recognition July 2009*

[7] M. Delalandre, E. Valveny, J. Lladós, "Performance evaluation of symbol recognition and spotting systems: An overview", in: *Workshop on Document Analysis Systems (DAS)*, 2008, pp. 497–505.

[8] F. Ge, S. Wang and T. Liu, "New benchmark for image segmentation evaluation", *Journal of Electronic Imaging* 16 (3) (2007), p. 033011.

[9] A. Antonacopoulos, D. Karatzas, and D. Bridson. "Ground truth for layout analysis performance evaluation.", In H. Bunke and A. Spitz, editors, *Document Analysis Systems*

VII, Proceedings of the International Association for Pattern Recognition (IAPR) Workshop on Document Analysis Systems (DAS2006), pages 302–311, Nelson, New Zealand, Feb. 2006. Springer Lecture Notes in Computer Science,

LNCS 3872.

[10] Jung_Me Park, Carl G. Looney, Hui_Chuan Chen, "Fast Connected Component Labeling Algorithm Using A Divide and Conquer Technique." Technical report, 2000.

(<http://cs.ua.edu/TechnicalReports/TR-2000-04.pdf>).

[11] Hildelies Balk, "IMPACT Half Year Report", October 2004, available from <http://www.impact-project.eu>

[12] "Flood Fill", Wikipedia, http://en.wikipedia.org/wiki/Flood_fill

Appendix

A Xml Glyph Definition

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://schema.primaresearch.org/gt/gts/pagecontent/2008-11-01"
  elementFormDefault="qualified" xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:pc="http://schema.primaresearch.org/gl/gts/glyphcontent/2008-11-01">

  <element name="pcGts" type="gl:glGtsType"></element>
  <complexType name="glGtsType">
    <sequence>
      <element name="pcMetadata"
type="gl:pcMetadataType"></element>
    </sequence>
    <element name="p" type="gl:glyphType"
      minOccurs="0" maxOccurs="unbounded">
    </element>
    <attribute name="glGtsId" type="ID"></attribute>
  </complexType>

  <complexType name="glMetadataType">
    <sequence>
      <element name="pcCreator" type="string"></element>
      <element name="pcCreated" type="dateTime"></element>
      <element name="pcLastChange" type="dateTime"></element>
      <element name="pcComments" type="string" minOccurs="0"
        maxOccurs="1"></element>
    </sequence>
  </complexType>
  <complexType name="glyphType">

    <element name="coordGroup" type="gl:coordGroupType"
      minOccurs="0" maxOccurs="unbounded">
    </element>

    <attribute name="WordRefid" type="ID" use="required"></attribute>
    <attribute name="id" type="ID" use="required"></attribute>
  </complexType>

  <complexType name="coordGroupType">

    <element name="p" type="gl:coordsType"
      minOccurs="0" maxOccurs="unbounded">
    </element>
```

```
<attribute name="shade" type="int" use="required"></attribute>
<attribute name="outline" type="int" use="required"></attribute>
<attribute name="core" type="int" use="required"></attribute>

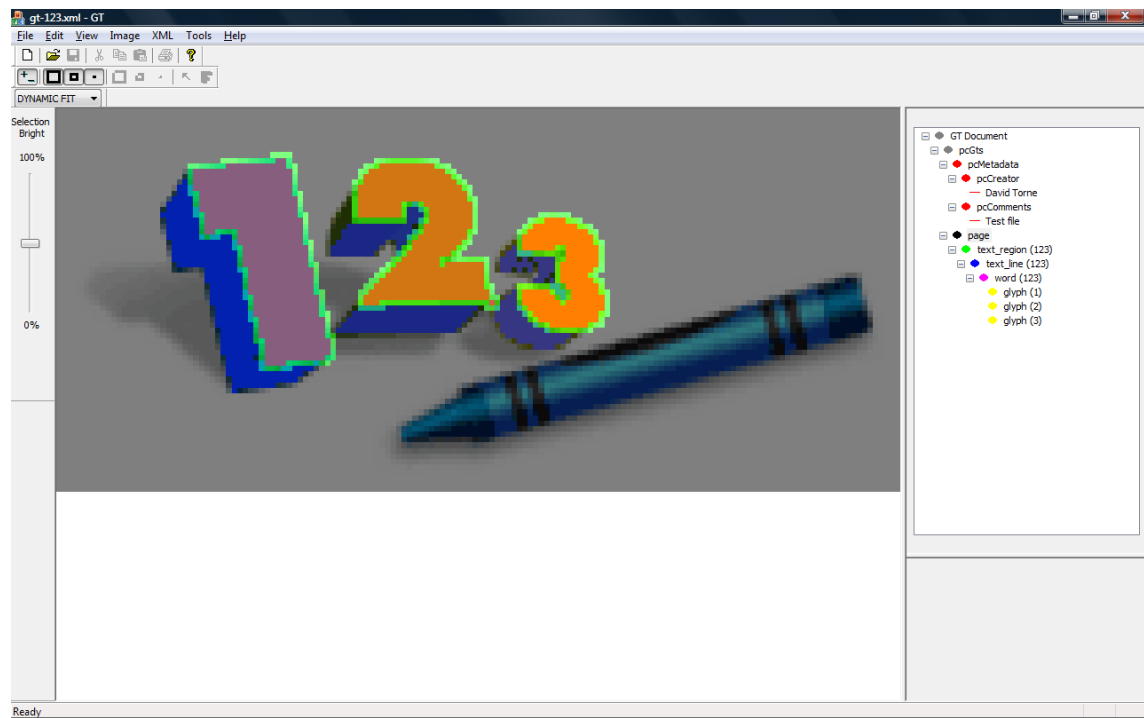
<attribute name="no_coords" type="int" use="required"></attribute>
<attribute name="id" type="ID" use="required"></attribute>
</complexType>

<complexType name="coordsType">
  <attribute name="x" type="int" use="required"></attribute>
  <attribute name="y" type="int" use="required"></attribute>
</complexType>
</schema>
```

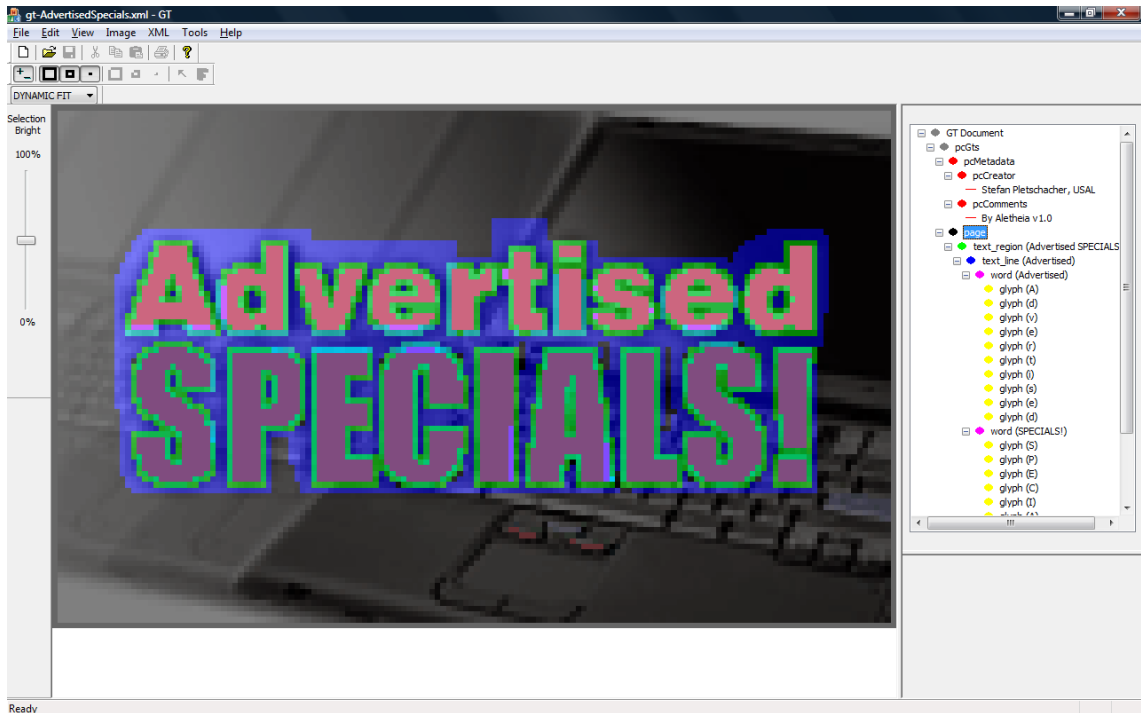
B Ground Truthed Images



24bit.jpg



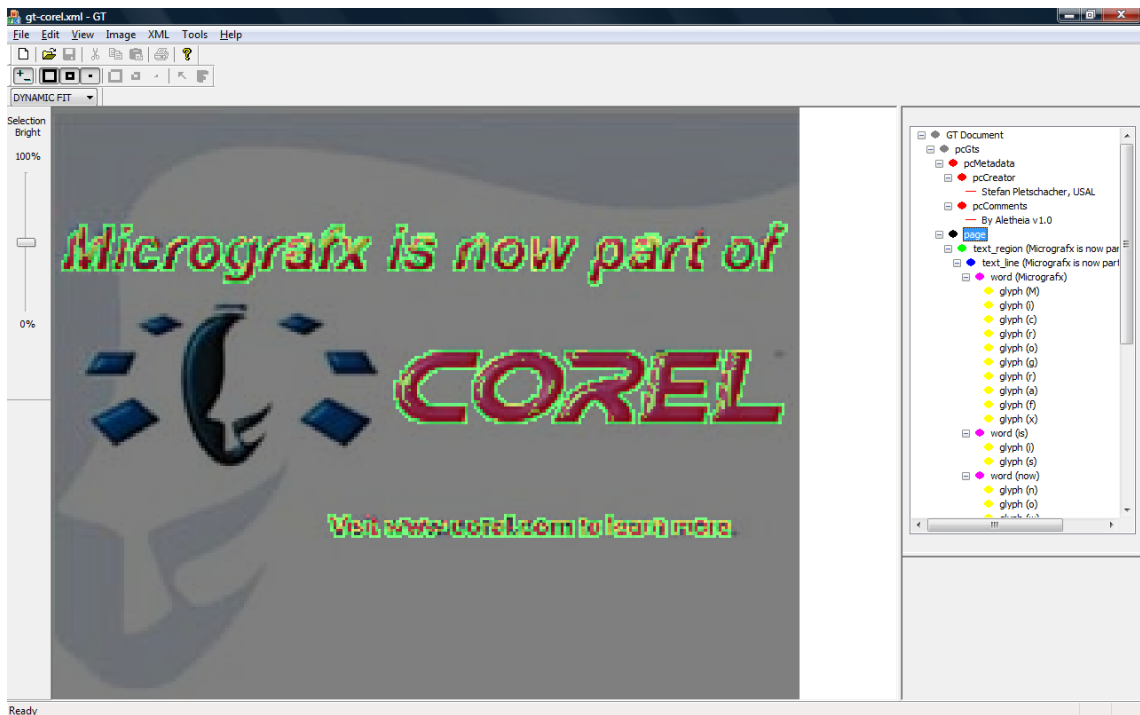
123.bmp



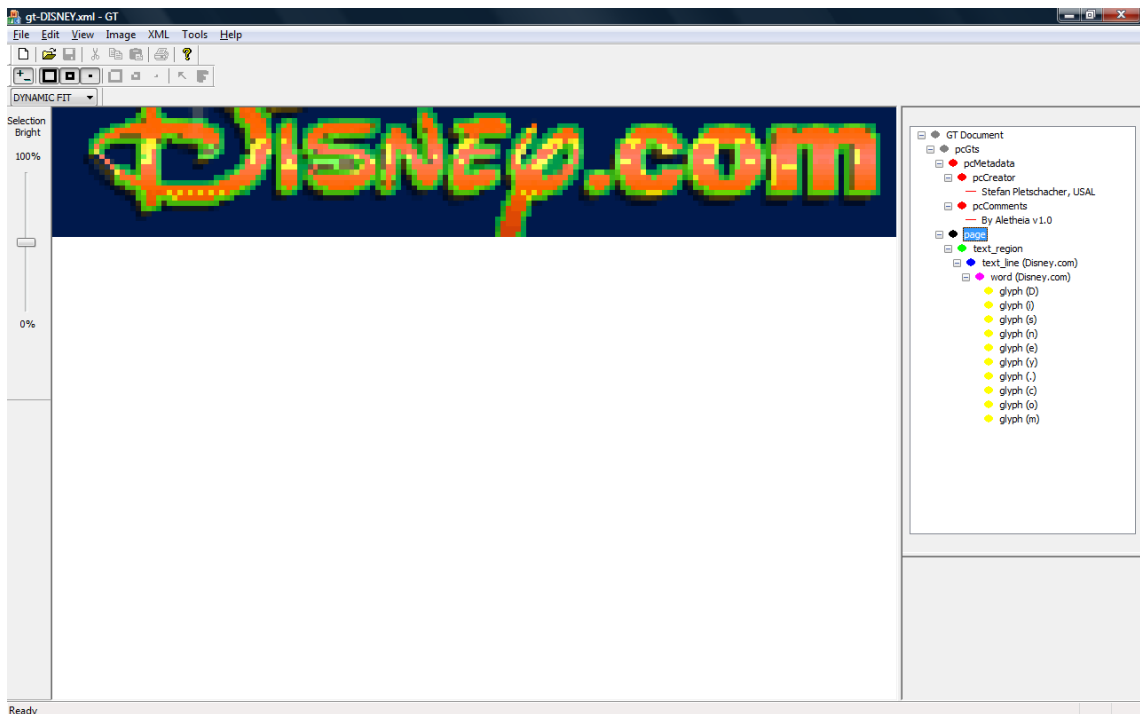
AdverstisedSpecials.bmp



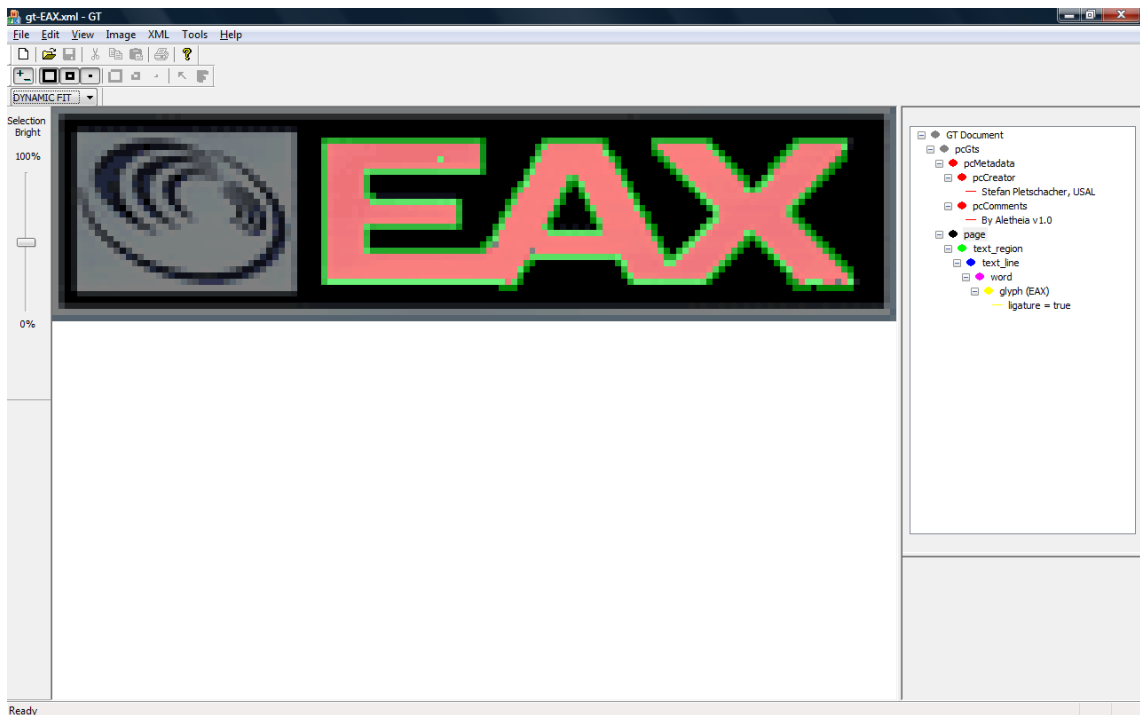
BarnesNoble.bmp



Micrograf.jpg



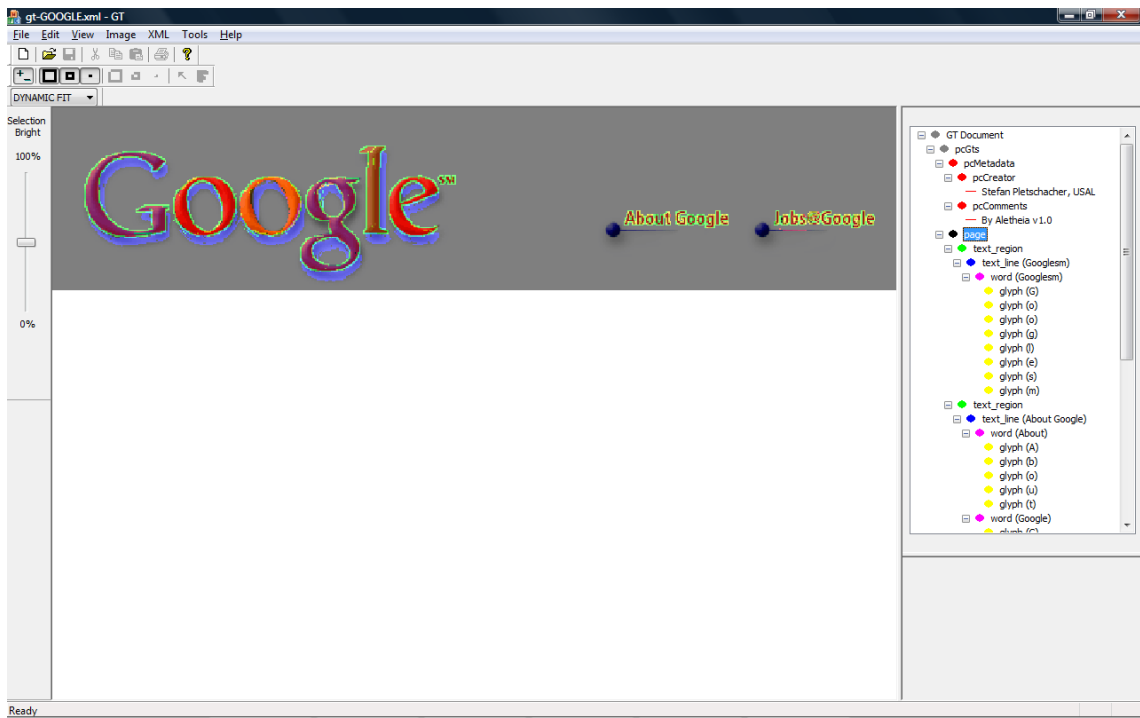
DisneyLogo.bmp



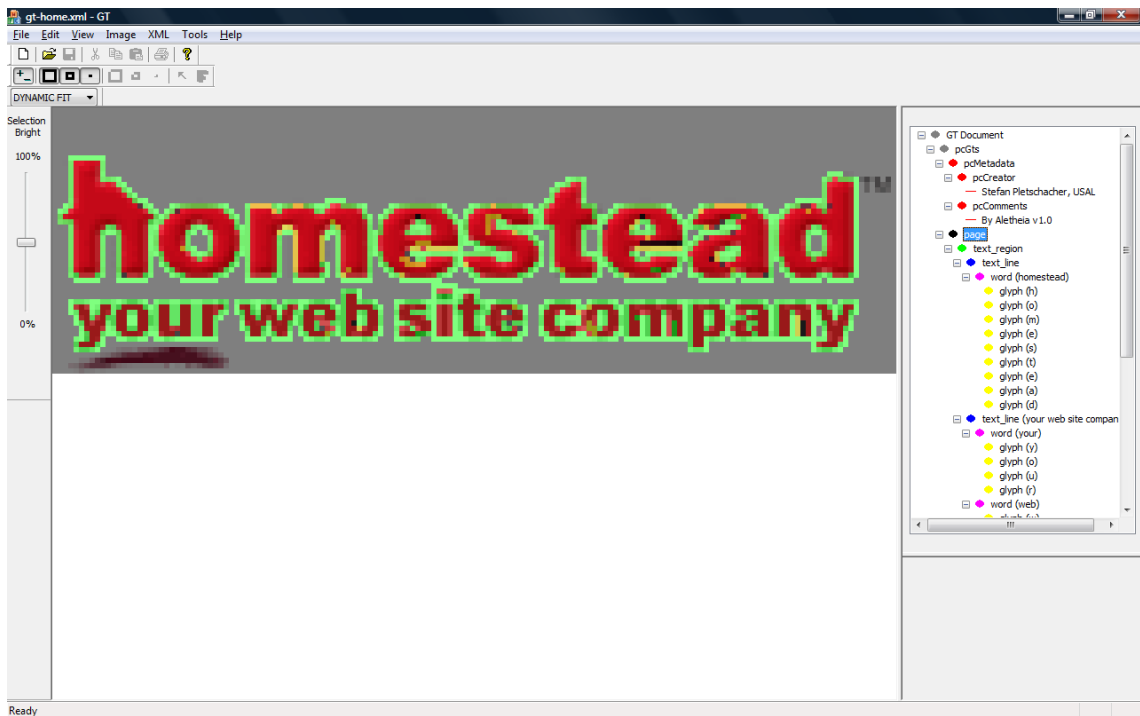
Eax.bmp



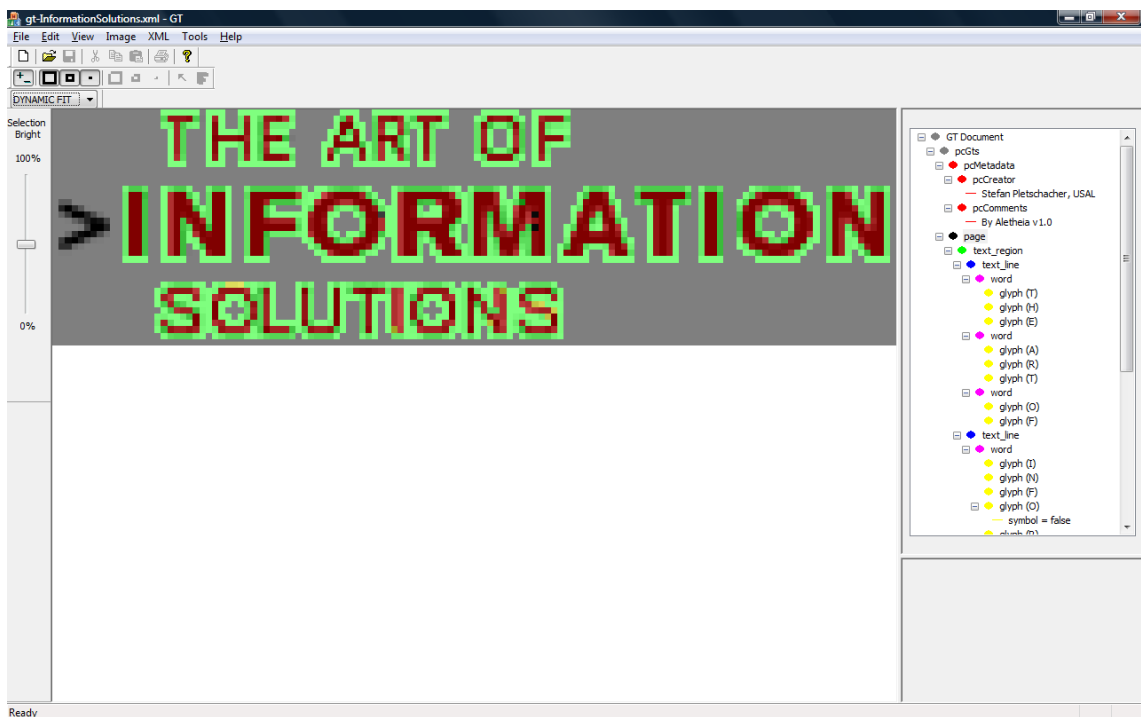
ebayLogo_Large.bmp



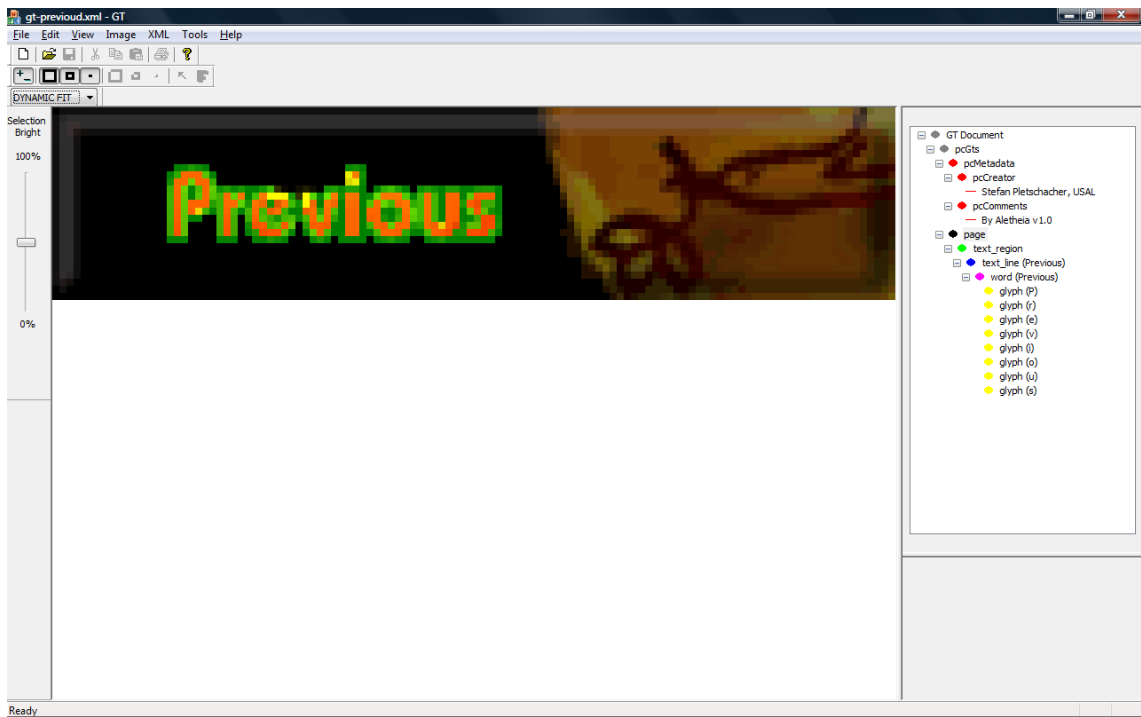
Google.bmp



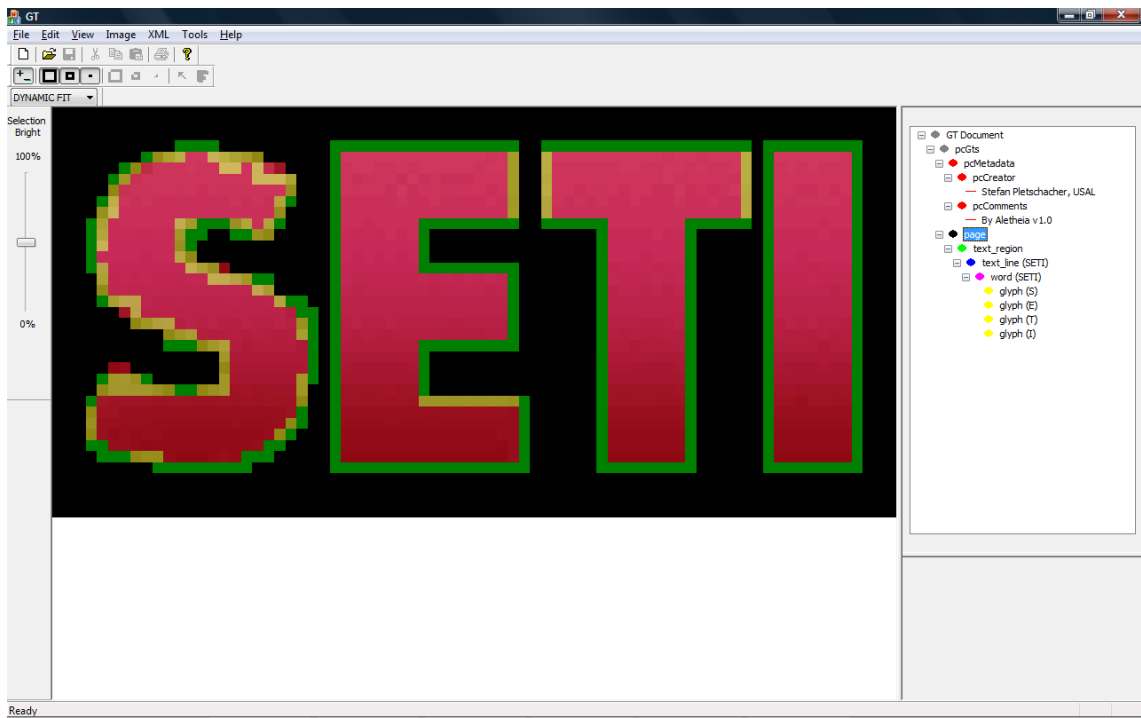
Homestead.bmp



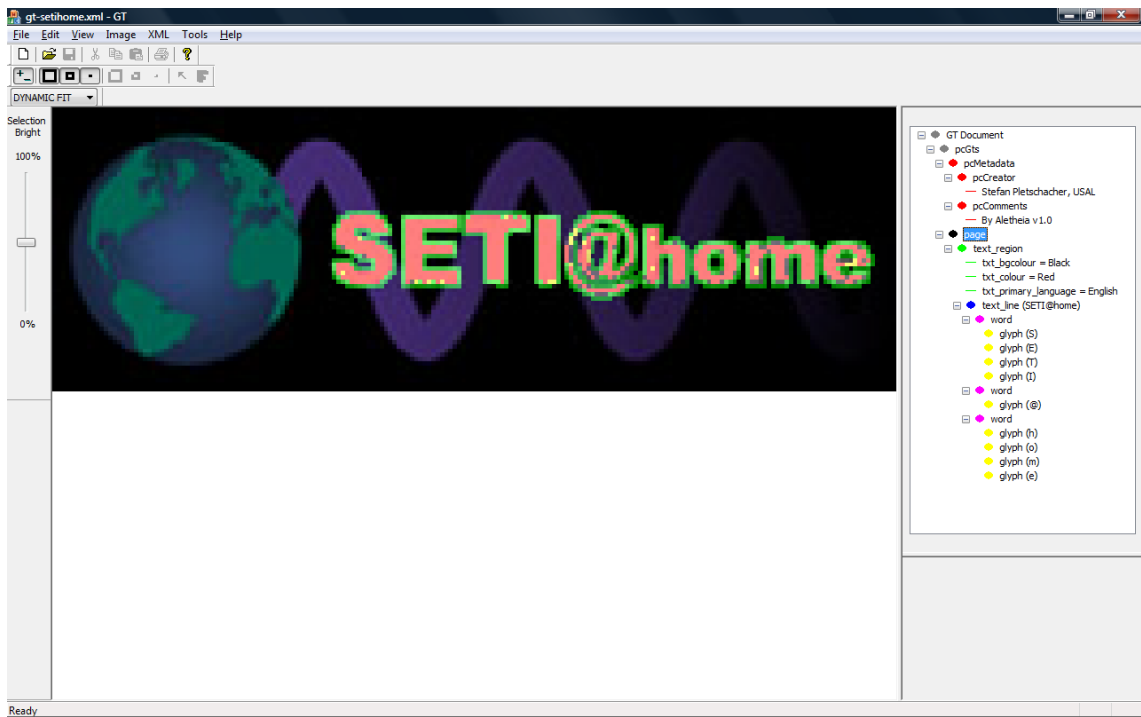
InformationSolutions.bmp



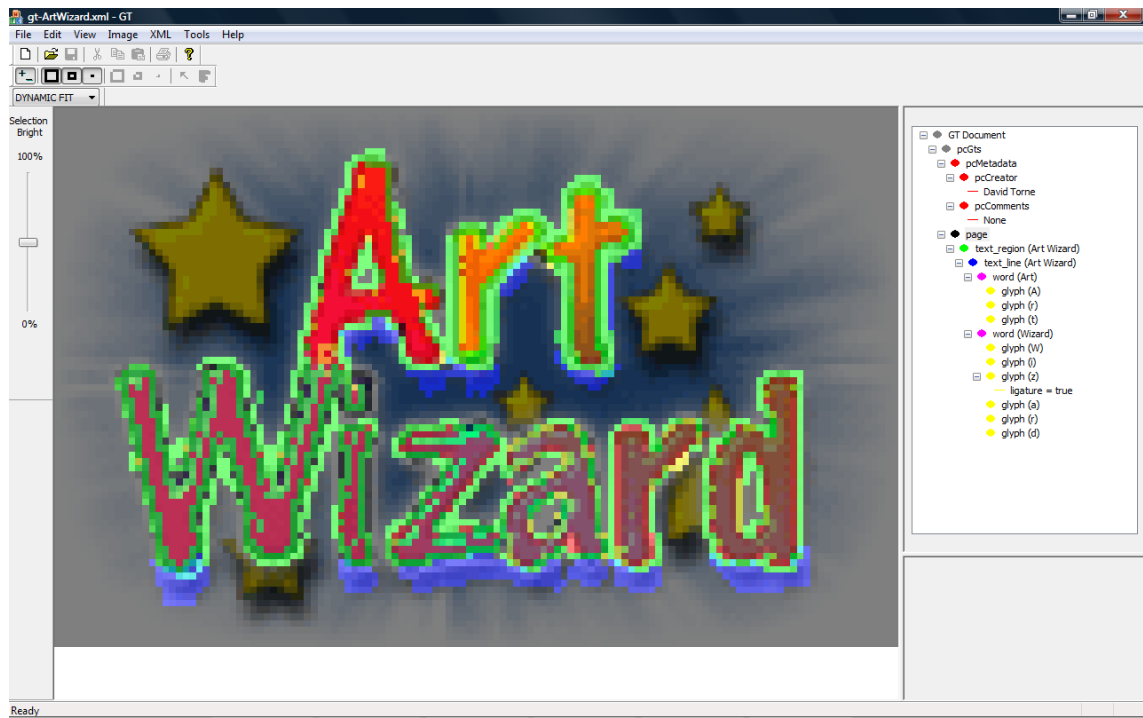
PreviousConnect.bmp



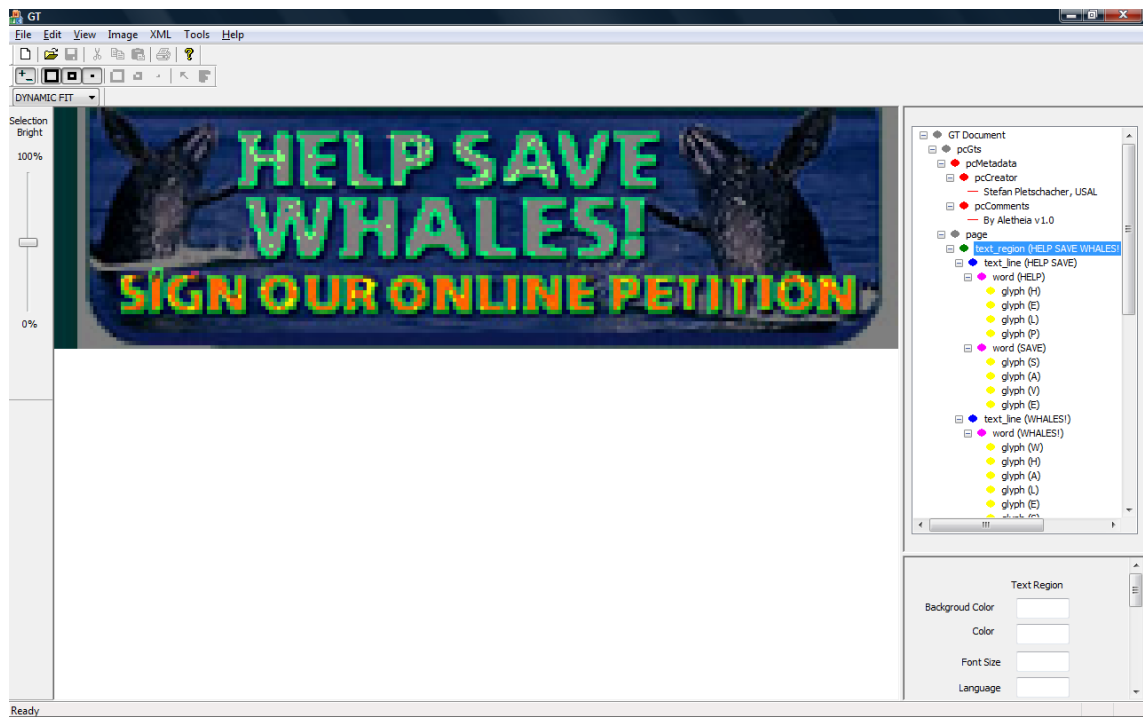
SmallSetiLogo.bmp



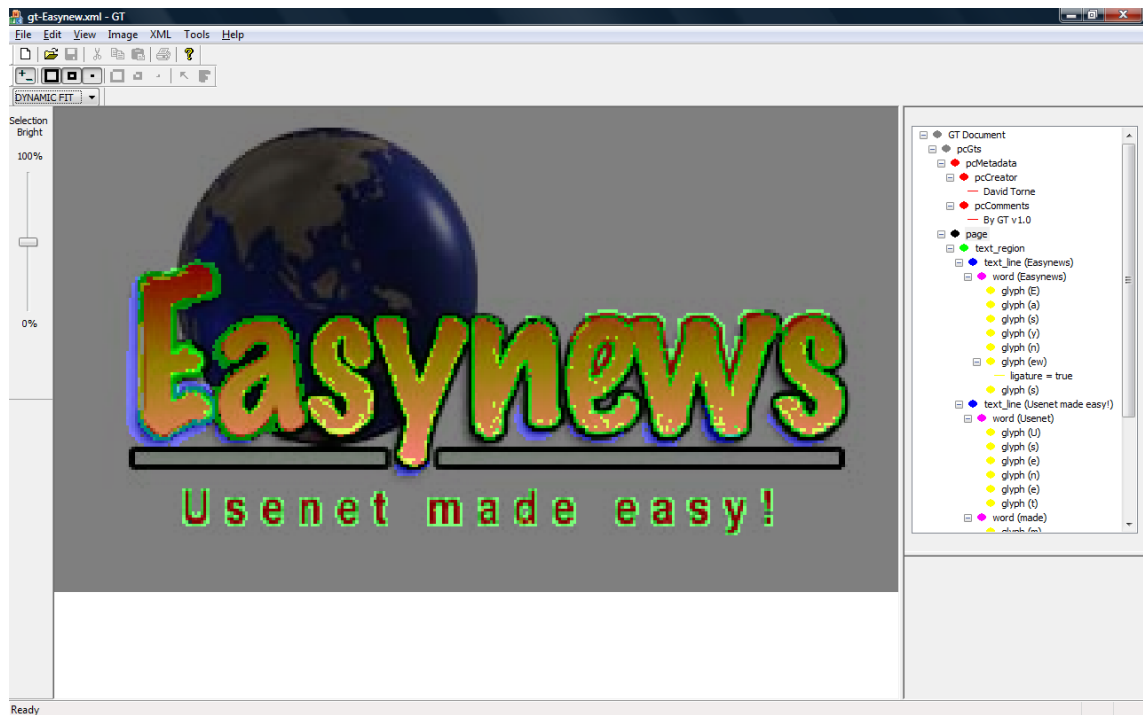
SetoLogo.jpg



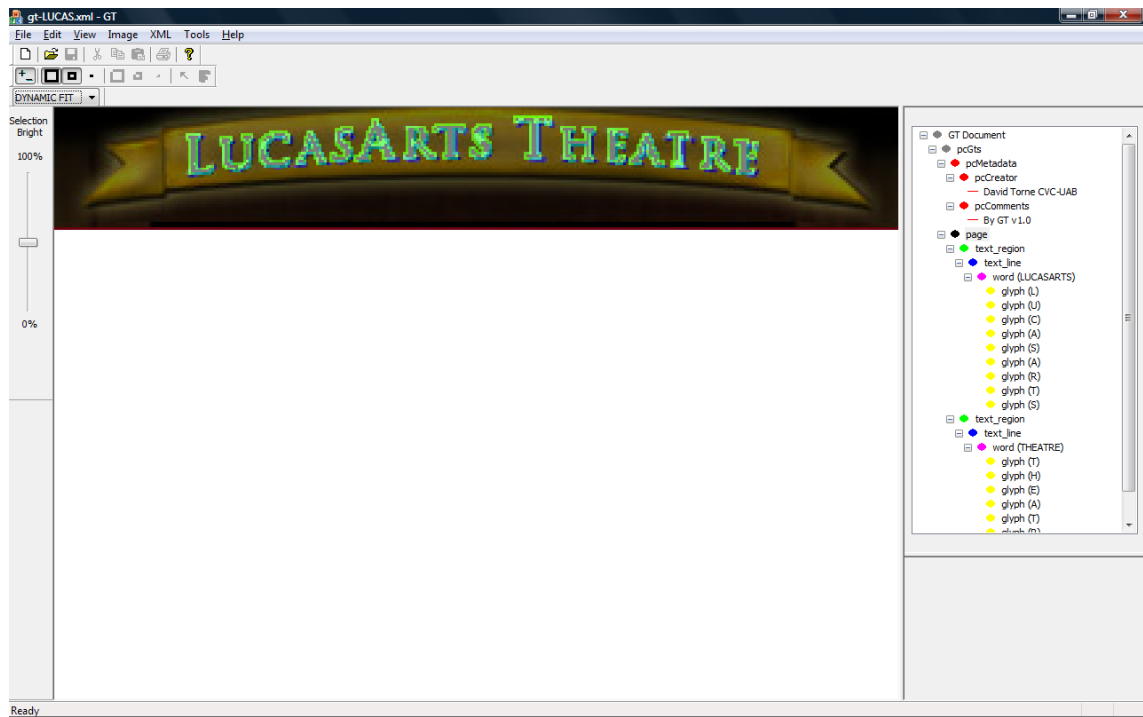
ArtWizare.bmp



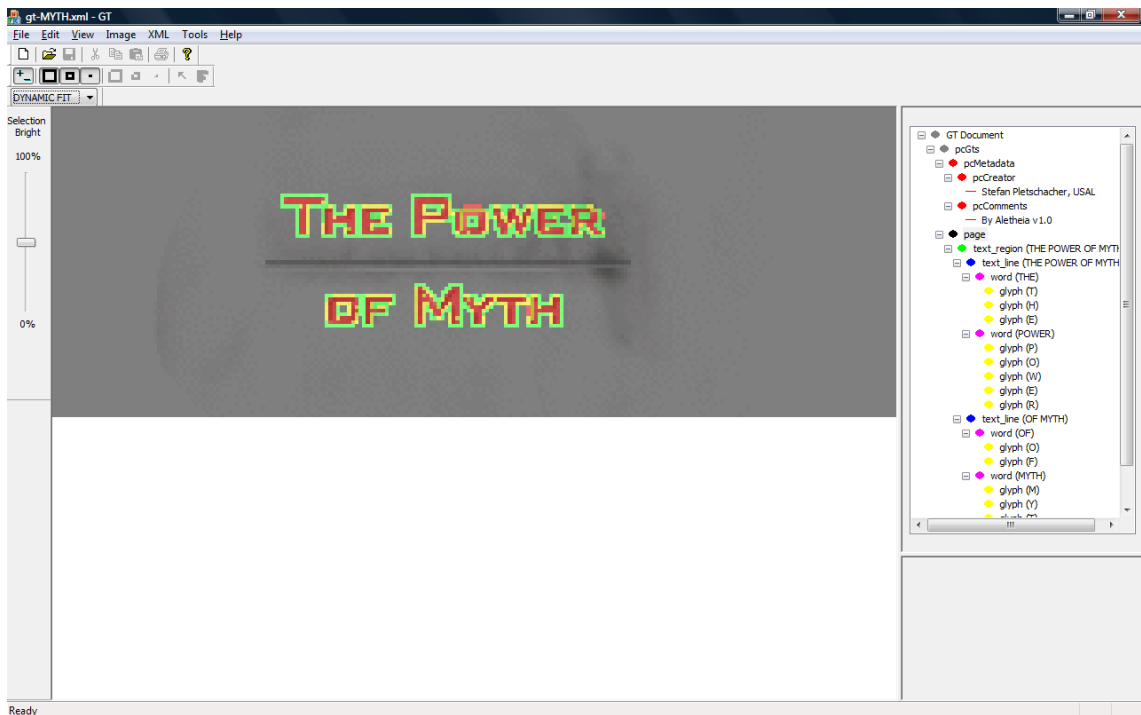
HelpWhales.jpg



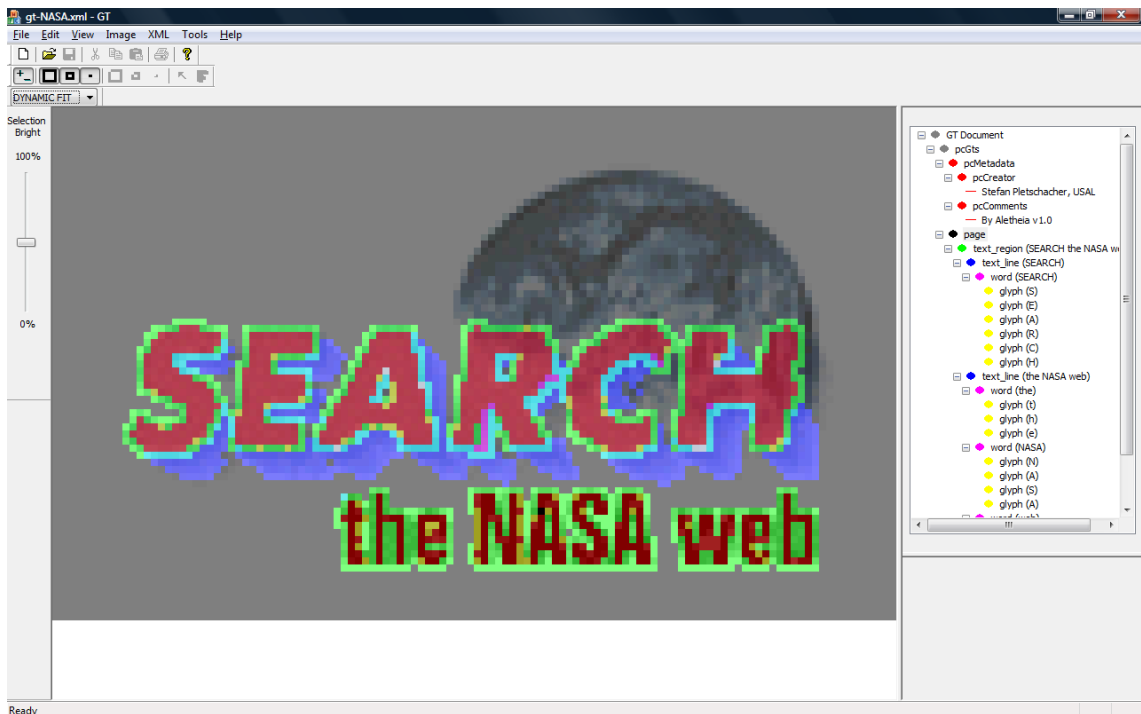
Easynews.jpg



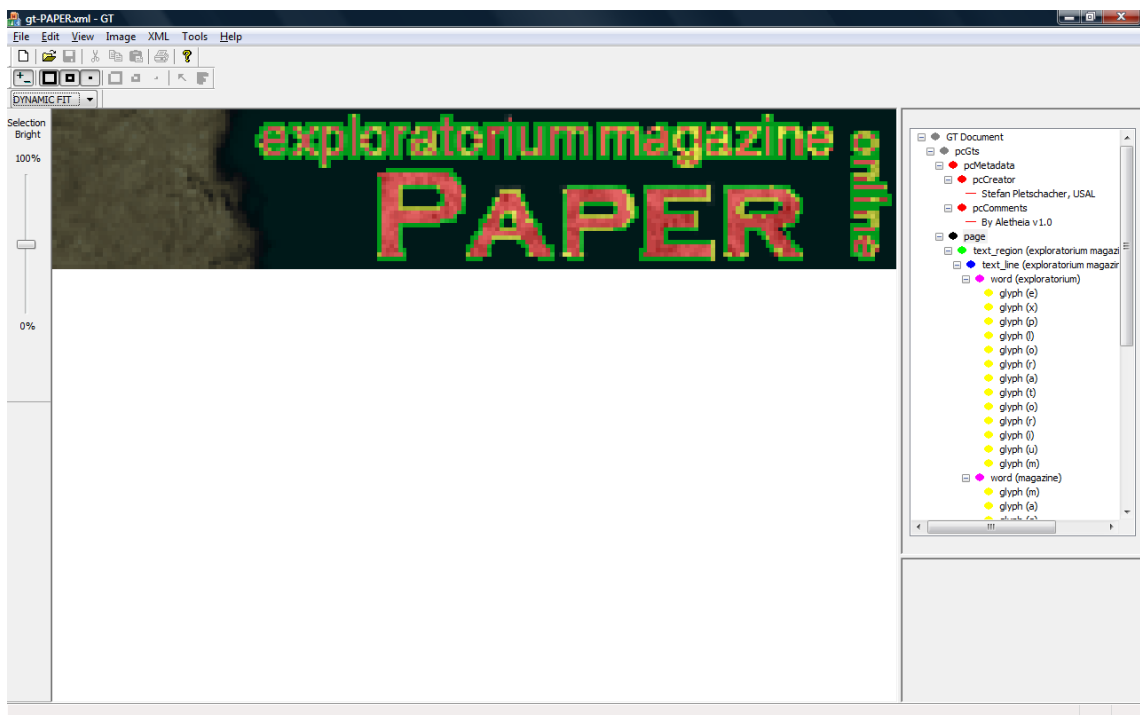
LucasArtsTheater.bmp



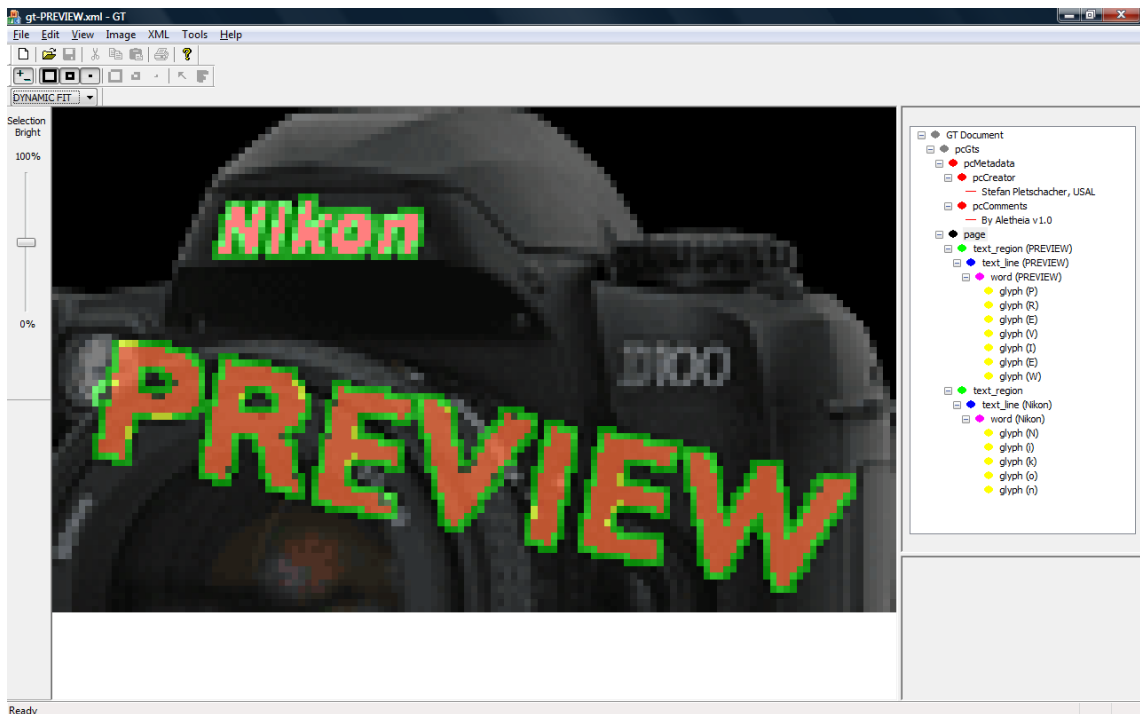
ThePowerOfMith.bmp



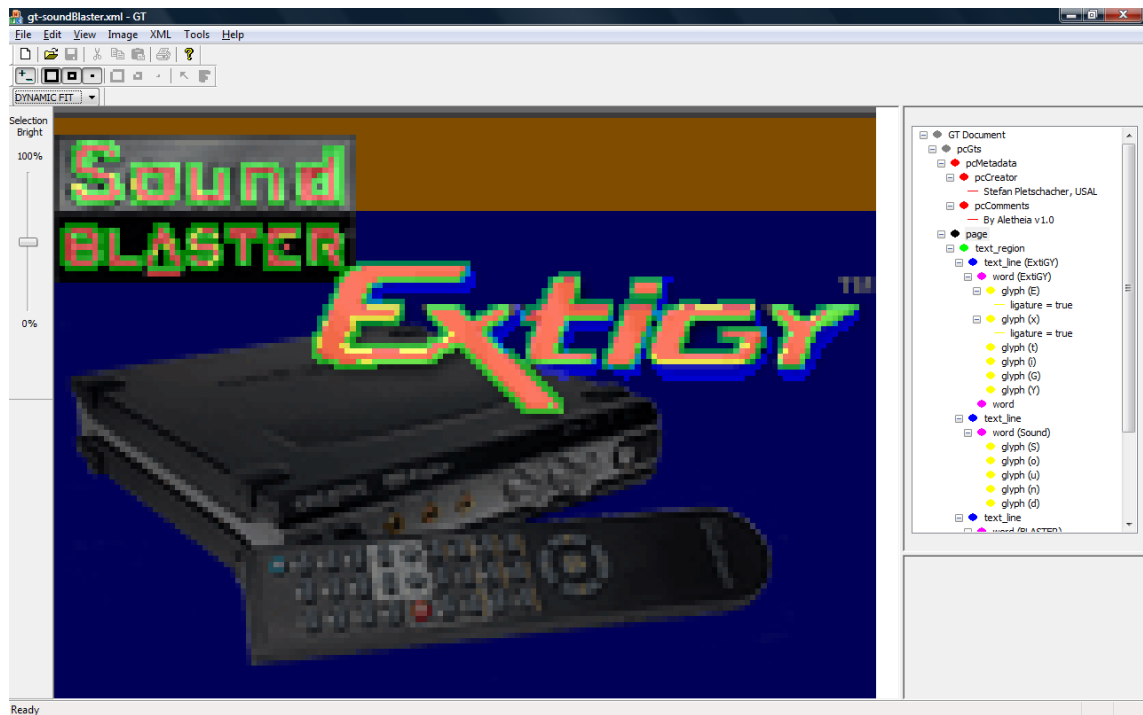
SearchNASA.bmp



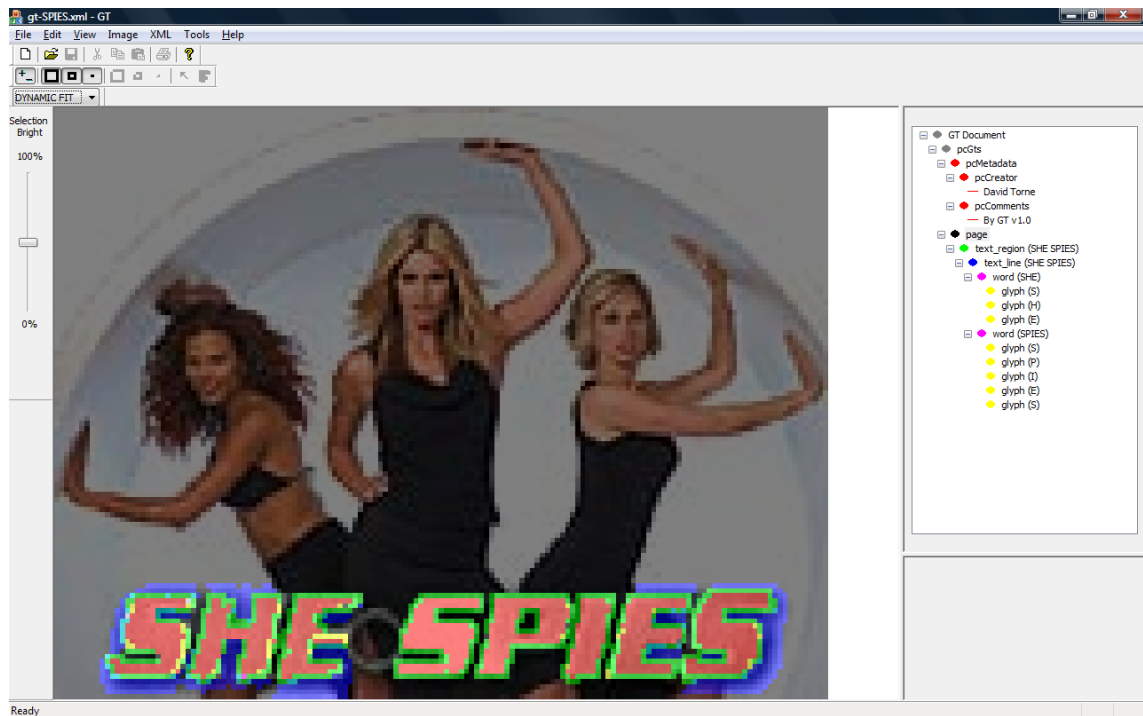
PaperBanner.bmp



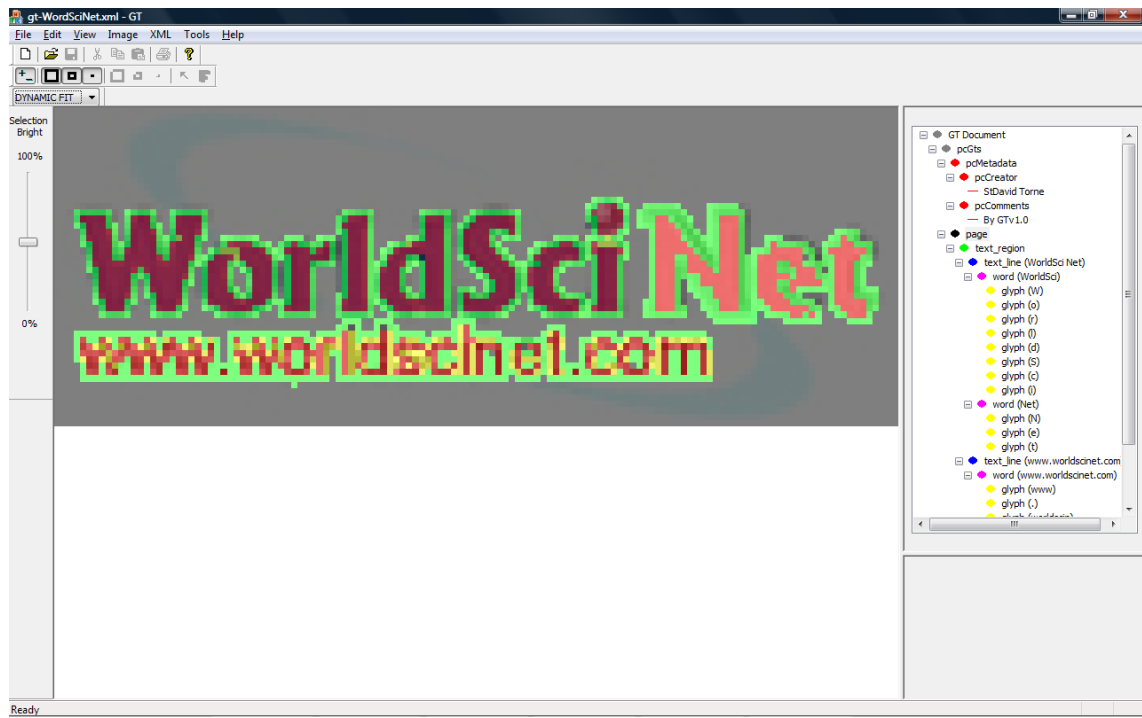
NikonPreview.bmp



Extigy.bmp



SheSpies.bmp



WordSciNet.jpg

Summary of the thesis (English)

This thesis summaries the work of the Computer Engineering of degree project.

It will explain the main reasons to do the project as well examples that illustrated the resulting application that will try to solve the need, in this case for the creation of Ground Truth data sets for algorithms of complex color text segmentation.

All the processes of the creation will explained in different chapters from the definition of problem , the work plan, the requirements and the design, to a complete illustration of the resulting software and corresponding data sets.

Resum de la memòria (Català)

Aquesta memòria resumeix el treball de final de carrera d' Enginyeria Superior d'Informàtica.

Explicarà les principals raons que han motivat el projecte així com exemples que il.lustren l'aplicació resultant. En aquesta cas el software intentarà resoldre la actual necessitat que hi ha de tenir dades de Ground Truth per als algoritmes de segmentació de text per imatges de color complexes.

Tots els procesos serán explicats en els diferents capítols partint la definició del problema, la planificació, els requeriments i el disseny fins a completa il.lustració dels resultats del programa i les dades de Ground Truth resultants.

Resumen de la memoria (Español)

Esta memoria resume el trabajo de final de carrera de la carrera de Ingeniería Superior de Informática.

Explicará las principales razones que han motivado la realización del proyecto así como ejemplos que ilustran la consecuente aplicación. En este caso se intentará resolver la actual necesidad que hay en tener datos Ground Truth para los algoritmos de segmentación de texto para imágenes de color complejas.

Todos los procesos serán explicados en los diferentes capítulos partiendo de la definición del problema, la planificación, los requerimientos y el diseño hasta una completa ilustración de los resultados del programa y de los datos de Ground Truth resultantes.