

Project Final Report for CPSC 503 Project

“3-dimensional visualization of Shape differences”

Ronan SICRE

Department of Computer Science
University of Calgary, Canada
2500 University Dr. NW
Calgary AB T2N 1N4

7th December 2006

1 Abstract

The Bone Imaging Laboratory where the project takes place owns a micro-CT scanner: medical high definition scanner. It scans small slices of human bones (arm or foot). It also can scan animal bones (mouse, dog...). Using those 3D data given by the scanner, a lot of processing can be done. Previous projects aim was to find ways to show or detect osteoporosis (skeletal disease).

In order to see the evolution of this disease, the aim of this project is to compare bone images and show the differences in a really user friendly way. To realize that, we will use an eye wear to see 3-dimensional representations from images displayed. Then an other part of the project will be to find ways of showing differences by using vectors drawing.

2 Introduction

Augmented reality systems are on the verge of being used everyday in medical training, preoperative planning, preoperative and intraoperative data visualization, and intra operative tool guidance.

Micro-CT (Microtomograph) imaging techniques is used with augmented reality. This imaging techniques can produce multiple planar images that are combined spatially to create a volumetric data set. With the volumetric data set, it is possible to reconstruct three-dimensional models of the anatomy being studied. In orthopedic surgery, for example, a fractured hip seen in a volumetric CT scan can be reconstructed into a three-dimensional model to allow the surgeon to visualize the entire hip volume from any direction. By using a wide variety of visual imagery and three-dimensional reconstructions, augmented reality

can be applied to numerous different applications including surgical guidance, localization of anatomic structures, and surgical training [Bla98].

The purpose of the project series is to develop a new method that can be used for high-throughput phenotypic (physical characteristics) analysis based on 3D image data obtained from micro-CT scanning of bones for musculoskeletal research.

A previous master project has been done this year [Gir06]. Its purpose was to use some methods that will superimpose and compares 3-dimensional mice bones representation.

Now this project's aim is to use some previous methods and new ones in order to show differences between two 3D bones representation. Each 3D bone is represented by using many images: voxels or surfaces data (marching cubes techniques). Different 3D representations will be used: the differences can be shown by vectors drawn all around the bone that link the two images compared. These differences must be clear and user friendly. Then it will be possible to use crystal eyes; there are glasses that gives 3-dimensional images from 2-dimensional ones displayed on a standard monitor.

3 Previous Work

Before the previous project started, the study of the Morphometrics (study of the shape) had a big importance. Morphometrics assess the differences in shape, its measurement is a real aspect of phenotype studies. In this case, the work realized by Dr. Hallgrímsson (Faculty of Medicine of the University of Calgary) who is heading ongoing research in the field of morphology was really important to define the goals of the different projects.

The concept of the previous project method is to obtain 3D micro-CT image data for each mouse in an experimental group. These images are superimposed in a common reference frame using a semi-automatic method. The images are then summed into a single 3D "Generalized Shape Image", containing mean density values and shape variations. Shape variations at bone surface may then be quantified (mean and standard deviation) by analysis of the GSI gradient. Another method, based on the position of the isosurface of mouse bone, can be used to quantify shape variation using a surface-to-surface distance measurement. This method provides statistical data which can be used to classify the different experimental groups of mice.

After understanding the goals of this project it has been found that it must be some better ways of representation. In fact the quality of the representation is really important in order to diagnose a bone disease or observe the bone micro structure. We know that 3-dimensional representation generated by stereoscopic tools improve the interaction for some applications. According to [JA02], an experiment was carried out regarding the importance of stereoscopic display and hand-image collocation for dextrous interaction. Their results were surprising. Stereo vision was very important to both accuracy and speed of task

completion. If the findings are borne out it has important implications for the volumetric presentation of medical data.

On an other hand, [IT96] deals with the basic characteristics of pointing to three-dimensional images generated by stereoscopic images. The experimental examination found that it was difficult to point to the correct position of the image and that the depth of the image was underestimated so that the object perceived by the subject was distorted. Also, subjects who were familiar with viewing 3D images made less error in position and size perception than did subjects having little experience in viewing 3D images. These results imply that the performance of 3D pointing in stereoscopic 3D images will improve with experience although it is difficult to perform a 3D pointing precisely.

4 Project

4.1 Tools

This part describes the different tools used for the project: hardware and software.

4.1.1 Micro CT scanner

Scans were performed using a Skyscan 1072 100-kV micro-CT. The protocol entails 0.9 rotation steps, 1.9 ms exposure time with 0.005 mm spatial resolution. Raw data obtained with the Skyscan are 2D slice stacks in “.bmp” format. The program stacks the 2D slices (AIM files) on top of one another to create a 3D image file. For accurate conversion, the element size must be given. This value is contained within the "Dicom" file and is 0.00625 mm for all specimens used in this project. The large data size creates some difficulties when attempting to open these files with a 3D visualization program. Therefore, the images are re-sampled by a factor of 2 in each direction. More information on [http].

4.1.2 Software sources

Visualization Toolkit (VTK) The Visualization Toolkit (VTK) [Kit03] is an open source package maintained by Kitware, Inc. It consists of a set of C++ classes to visualize a wide variety of data [SML]. The package is platform-independent and uses existing visualization libraries like OpenGL or Mesa. It incorporates object-oriented principles and event-observer mechanisms. The VTK package is used for the application of functions implemented in the project program. These functions allow filtering, modification and visualization of the data and its attached attributes [http]. Here Version 5.0 is used.

CMake CMake is a software provide by the Kitware group (VTK) to simplify the construction of project software . It allows for all links to VTK and ITK libraries, which CMake automatically configures, as well as also linking the interface creator software. The VTK and ITK groups provide software to build

some GUI but you also can create your GUI with software like Qt [http]. Version 2.4 is used.

4.1.3 Crystal eyes

CrystalEyes is a lightweight, wireless set of liquid crystal shutter eye wear for Stereo 3D imaging in engineering and scientific applications. The product delivers high-definition, stereoscopic 3D images on all major UNIX, Windows 2000/NT/XP and Power Mac G5 workstations displays. CrystalEyes is activated by an infrared emitter that connects to the user's workstation [httpd].

CrystalEyes is supported by most professional software applications used in product visualization and simulation, molecular modeling, GIS/mapping and medical imaging [httpc].

4.2 Visualization

4.2.1 Data Format

Let us first describe the data. This is representative of the most commonly used techniques in image processing. This project uses ".PLP" data built from ".AIM" data: a 3-D image data type developed specifically for Micro-CT Scanner (XTremeCT Scanco Medical). The surface data (polydata) are normally constructed from voxel data (slices) using algorithms such as the Marching Cubes algorithm (in our case). It is possible to attach information to each cell (that represent a surface), that will be necessary to find the distance to the moving 3-D image.

First of all, the program visualizes polydata. The polydata used was a 3-D image of a bone: ".PLD" file. A brief description of the graphic pipeline used follows.

4.2.2 Pipeline

PolyDataReader -> PolyDataMapper -> 3DActor -> Render

The *vtkPolyDataReader* filter reads the polydata of the file. Point and cell data of the surface are read as well as the scalar data. A transition filter must be applied to visualize this data.

The *vtkPolyDataMapper* creates the surfaces from the marching cubes polydata. This filter controls the parameters of the scalar attribute data, for example: visibility, color, lookup table.

The *vtk3DActor* creates the actor displayed in the window. This filter allows the control of the visibility of the model, the opacity of the surfaces and the color of the model.

The *vtkRender* is the last filter applied. When this filter is used all steps of the graphic pipeline are created and the data is ready to be displayed in the window.

To build the project program, Cmake was used. An example was also available to understand the different visualization methods and pipeline. This example was a program that visualize a “.AIM” data with all the options possible. Notice that the pipeline is different that the one used for the polydata.

4.3 Crystal eyes

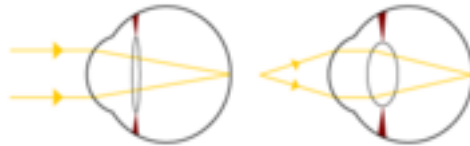
So, next on the project will be to use the "crystal eyes" system. Let us first describe the human vision system.

4.3.1 Description of human perception of depth

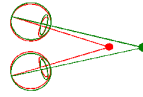
According to [per99], there are different ways of perceiving depth from ocular or binocular information.

stereo vision or stereopsis, it is the process of perceiving the relative distance to object based on their lateral displacement in the two retinal images. It is possible because our eyes are laterally separated and their visual fields overlap in the central region of vision. The two images of most objects in the overlapping portion are slightly different. So, the same point in the environment projects to locations on the left and right retinae that are displaced up a way that depends on how much closer or farther the point is from the fixation point.

Accommodation It is the process through which the ciliary muscles in the eye control the optical focus of the lens by temporarily changing its shape. This is a monocular depth cue (available from a single eye). If the visual system has information about the tension of the muscles that control the lens's shape, then it has information about the distance to the focused object.



Convergence It correspond to the extent to which the two eyes are turned inward to fixate an object. When both eyes fixed the same point, the same light coming from it falls on the center of both fovea (each fovea has one center). Here the fact that convergence provides information about depth is that the angle formed by the two lines of sight varies systematically with the distance between the observer and the fixated point.



4.3.2 Description of Crystal eyes

Crystal eyes offers the possibility by wearing special glasses to see 3-dimensional images from 2-dimensional ones displayed on a standard monitor. It is possible whenever a means is provided for creating and presenting different views to the left and right eyes. Each view is taken from an appropriately shifted viewing location.



This eye wear build an illusion of depth. In fact the user stereovision system and its convergence sensors, will “believe” that it is real depth images but the convergence sensors will says that the image is flat. Finally it really gives the feeling that the bone (image displayed) comes out of the screen.

4.3.3 Precisions

The system includes an infrared emitter to send out a pulse identifying a field as left or right, and stereoscopic eye wear that receives and decodes the pulse and creates the drive signals necessary to activate the shutter.

There are already some function in VTK that draw stereo-images. In fact "crystal eyes" mode uses frame-sequential capabilities available in OpenGL to drive LCD shutter glasses and stereo projectors.

To make it work, the crystal eyes hardware, the video card and the program code have to work together. This part require a lot of understanding about the possible configurations of each part. In this case, the video card is the main part that is the link between the eye wear and the program. Also problems are difficult to find because there is no efficient way to define if the error comes from the program or the video card configuration (or maybe the eye wear).

4.4 Vectors drawing

The next aim of this project is to give an idea of the differences between two bone images by drawing vectors on the first image (fixed). These vectors will represent the position of the closest point on the second bone (moving). The method used is called HedgeHog.

The next step is to superimpose the two compared representation. In fact the superimposition requires a registration algorithm. Image registration transforms and interpolates image data into a common orientation for effective analysis. Registration is the process of finding the transformation that best maps the

points of one image, called the moving image, to another image, called the fixed image. There are three main kinds of registration: point-based, surface-based, and volume-based. Volume-based registration is the most accurate. Registration is called rigid when the transformation involves only rotation and translation. In many cases it is useful to change the scale factor; the registration is therefore called similarity. This method has already been implemented and is used from a previous program facilitated by stereoscopic.

Then a method can be used to get for each surface of the fixed bone 3D image the distance of the closest surface in the moving image. The surface to surface measurement that will be used allows calculating the distance and making some scalar data matching based on the closest point method. For a surface to surface measurement always two scans of the bone are needed. In the case of the site-specific change, these scans are normally related to different points in time. In general site-specific means a measurement that is performed between two bones and returns the change in all points of the respective surfaces. In our case, the algorithm was already implemented.

From here, another method could be used to determine the normal of the point that will define the direction of the vector. The problem here is that it is not possible to have a normal vector to a point. So the normal has to be attached to a surface. But for each point there are several surfaces linked. This method: "vtkPolyDataNormals" works by determining normals for each polygon and then averaging them at shared points.

At the end, the basic bone is displayed with all the vectors (now each magnitude and orientation are known). The drawing will be complete by using the 3D visualization Tools.

5 Results

It is important to notice that the 3D representation of bones are really new and did appear in the last 2-3 years. It has already improved and helps the medical staff especially for diagnosing disease and observing bones microstructure. Using stereoscopic tools is becoming more and more popular as the development of augmented reality in the medical field

5.1 Previous program

The crystal eyes options has been set up on the program of the previous project. This program does registrations on images to match them. The stereoscopic display give a really better render, it is then easier to pick the points (first step of registration).

Although, [IT96] says that it is difficult to pick point using stereoscopic, but in this case the users who do the calibration are already use to do it with the

same program on a screen, and feel really better with the stereoscopic 3D. Notice that [IT96] has been written in 1996 and stereoscopic drawing has improved since then.

In our case, The result concerns more the precision of the point picking than the speed. Those results are based on two users feeling.

5.2 Main project

[JA02] describe the gain of this visualization tools in the medical field compare with hand-image collocation. The results are important for the medical staff. It was confirmed by the feeling of each user of the program.

By using this eyes wear, the differences between the two bones are really easy to see. Especially because the depth given by the visualization is directly linked to the eyes wear system. Then it gives some really nice and user friendly representations. For those who do not catch easily the 3D on the screen (or are not use to it), the stereoscopic system is advantageous.

On the main program, for the HedgeHog vector drawing option, if the image is static it is not easy to catch the direction of the vectors. It is improved by the fact that it is possible to act on the picture and move it in every direction (and zoom on every area). But here is the where Crystal eyes improves visualization. it is possible to see the vectors in 3D and that was deffinitly an improvement to the system.

One problem that plagues stereoscopic displays as used in traditional virtual reality setups is eye strain, and in extreme cases, nausea. This occurs because the eyes must focus on the display image which is located physically on the surface of the screen. The brain, however, is fooled by the stereoscopic effect into thinking that the three-dimensional image is located at some other location closer to or farther from the screen. This discrepancy causes fatigue. Eye strain or discomfort can be reduced greatly in an augmented reality system by carefully placing the plane of zero parallax (the plane on which points viewed by the left eye and right eye appear at the same location) within the center of focus of the environment being viewed. In nonstereoscopic systems, eye strain also can be reduced by using corrective optics to place the image plane in the middle of the environment being viewed [Bla98]. This has been confirmed by users who did keep the goggles for a long time.

Also, scientists are beginning to recognize that stereoscopic visualization of scientific data greatly facilitates their efforts to gain support for research and to help them get needed funding. Presenting their work in 3-D stereo in a way that clearly conveys the meaning of the data not only helps explain the researcher's idea, but also casts a glow of both credibility and capability.

6 Conclusion

To conclude, most of the time here was spent on learning how to use the different programs, finding the interesting methods and functions. Understand how to use and the benefits of the different 3D display tools and the "crystal eyes" system in order to link the different parts. Now crystal eyes is available for about each program done in the Laboratory and will definitely find some new applications for the next projects.

7 Appendix

7.1 Video card configuration

video card configuration:

Here the aim is to modify the configurations of the NVIDIA X driver to allow the good stereo:

Add the following to the "Device" section of the XF86Config file
(/etc/X11/XF86Config in Red Hat EL/WS 3):
Option "Stereo" "3"

There is an easier way to do it by using some commands directly in the shell in root mode :

`nvidia-xconfig -stereo=STEREO/-no-stereo`

Enable/Disable the stereo mode. Valid values for STEREO are: 1 (DCC glasses), 2 (Blueline glasses), 3 (Onboard stereo), 4 (TwinView clone mode stereo), 5 (SeeReal digital flat panel), 6 (Sharp3D digital flat panel).

`nvidia-xconfig -c, -xconfig=XCONFIG`

Use XCONFIG as the input X config file; if this option is not specified, then the same search path used by the X server will be used to find the X configuration file.

`nvidia-xconfig -o, -output-xconfig=OUTPUT-XCONFIG`

Use OUTPUT-XCONFIG as the output X configuration file; if this option is not specified, then the input X configuration filename will also be used as the output filename.

`nvidia-xconfig -h, -help`

Print usage information for the common commandline options and exit.

`nvidia-xconfig -A, -advanced-help`

Print usage information for the common commandline options as well as the advanced options, and then exit.

In our case, to enable the stereo mode for crystaleyes, we use the commands that follows:

```
nvidia-xconfig -stereo=3
nvidia-xconfig -o -stereo=3
nvidia-xconfig -c -stereo=3
```

Notice: Follow the error messages after the first command. if everything works, there is no need for the second and third commands. Otherwise it will be probably necessary to write

7.2 Program Code

```
/*=====
Read a polydata and visualize it.
Use crystaleyes
Visualize HedgeHog vectors
Ronan SICRE
advisors: Dr. S. K. Boyd and Dr Jeff Boyd
=====

include "vtkSurfaceToSurfaceFilter.h"
include "vtkScalarsStatistics.h"
include "vtkLookupTableInf.h"
include "vtkConvertAttachedVectorsToPolyData.h"
include "vtkUtility.h"
include "vtkMarchingCubes.h"
include "vtkProperty.h"
include "vtkScalarBarActor.h"
include "vtkActor.h"
include "vtkPolyDataMapper.h"
include "vtkRenderer.h"
include "vtkRenderWindow.h"
//
include "vtkXOpenGLRenderWindow.h"
include "vtkHedgeHog.h"
//
include "vtkRenderWindowInteractor.h"
include "vtkImageGaussianSmooth.h"
include "vtkPolyDataWriter.h"
include "vtkTextProperty.h"
include "vtkCubeAxesActor2D.h"
include "vtkCommand.h"
include "vtkSphereSource.h"
include "vtkPointData.h"
include "vtkHistogram.h"
include "vtkXYPlotActor.h"
include "vtkStripper.h"
include "vtkImageConstantPad.h"
```

```

include "vtkBoxWidget.h"
include "vtkBox.h"
include "vtkTransform.h"
include "vtkClipPolyData.h"
include "vtkAIMReader.h"
include "vtkDoubleArray.h"
include "vtkArrowSource.h"
include "vtkGlyph3D.h"
include "vtkTextActor.h"
include "vtkScalarBarWidget.h"
include "vtkPolyDataReader.h"
include "vtkDecimatePro.h"
include "vtkPolyDataNormals.h"
include "vtkCellData.h"
include "vtkPointDataToCellData.h"
include "vtkImageAccumulate.h"
int main( int argc, char **argv )
char * filename1, *filename2;
int maskRatio = 50;
vtkPolyDataReader *reader = vtkPolyDataReader::New();
filename1 = argv[1]; //first argument: pld file read
reader->SetFileName(filename1); //reader for the pld file
printf(filename1);
printf("");
vtkPolyDataReader *reader2 = vtkPolyDataReader::New();
filename2 = argv[2]; //first argument: pld file read
reader2->SetFileName(filename2); //reader for the pld file
//staff
vtkPolyDataNormals *normals = vtkPolyDataNormals::New();
normals -> SetInput(reader->GetOutput());
normals -> FlipNormalsOff();
normals->SplittingOff();
normals -> Update();
// We create a simple pipeline to display the data.
vtkHedgeHog *hedgehog = vtkHedgeHog::New();
hedgehog->SetInput(reader->GetOutput());
hedgehog->SetScaleFactor(1.0);
hedgehog->Update();
vtkPolyDataMapper *HHMapper = vtkPolyDataMapper::New();
HHMapper->SetInput(hedgehog->GetOutput());
HHMapper->SetScalarRange((hedgehog->GetOutput())->GetScalarRange());
HHMapper->ImmediateModeRenderingOn();
///end of staff
vtkPolyDataMapper *boneMapper = vtkPolyDataMapper::New();
boneMapper->SetInput(reader->GetOutput());
vtkPolyDataMapper *boneMapper2 = vtkPolyDataMapper::New();

```

```

boneMapper2->SetInput(reader2->GetOutput());
vtkActor *HHactor = vtkActor::New();//actor
vtkProperty *prop = vtkProperty::New();
prop->SetColor(1.0, 1.0, 1.0);
HHactor->SetProperty(prop);
HHactor->SetMapper(HHMapper);
vtkActor *bone = vtkActor::New();//actor
vtkProperty *prop2 = vtkProperty::New();
prop2->SetOpacity(1.0);//0.3
bone->SetProperty(prop2);
bone->SetMapper(boneMapper);
vtkActor *bone2 = vtkActor::New();//actor
vtkProperty *prop3 = vtkProperty::New();
prop3->SetOpacity(0.2);//0.3
bone2->SetProperty(prop3);
bone2->SetMapper(boneMapper2);
vtkRenderer *aRenderer = vtkRenderer::New();//renderer
vtkRenderWindow *renWin = vtkRenderWindow::New();//render window:
crystal eyes staff
renWin->AddRenderer(aRenderer);
renWin->StereoCapableWindowOn();
renWin->SetStereoTypeToCrystalEyes();
renWin->StereoRenderOn();
renWin->SetSize(300,300);
renWin->SetWindowName(filename1);
printf(renWin->GetStereoTypeAsString());
printf("");
vtkRenderWindowInteractor *iren = vtkRenderWindowInteractor::New();
//KEY
char keyPressed = iren->GetKeyCode();
if (keyPressed == 'o')
printf("Keypressed : 'o'");
if(prop2->GetOpacity() < 1)
prop2->SetOpacity(prop2->GetOpacity()+0.1);
if (keyPressed == 'p')
printf("Keypressed : 'p'");
if(prop2->GetOpacity() >0)
prop2->SetOpacity(prop2->GetOpacity()-0.1);
bone->SetProperty(prop2);
bone2->SetProperty(prop3);
iren->SetRenderWindow(renWin);
aRenderer->AddActor(bone);
aRenderer->AddActor(bone2);
aRenderer->AddActor(HHactor);
aRenderer->GetActiveCamera()->Azimuth(30);
aRenderer->GetActiveCamera()->Elevation(30);

```

```
aRenderer->ResetCamera();  
renWin->StereoCapableWindowOn();  
renWin->Render();  
iren->Start();  
return 0;
```

References

- [Bla98] Fritz MS; DiGioia Anthony M. MD Blackwell, Mike MS; Morgan. Augmented reality and its future in orthopaedics [symposium: Computer assisted orthopaedic surgery: Medical robotics and image guided surgery]. Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA, 15213, 1998.
- [Gir06] J. Gire. A high-throughput method to quantify shape differences for micro-ct. Master's thesis, Universite Jean Monnet, 2006.
- [htta] <http://www.cmake.org>.
- [httpb] <http://www.kitware.com>.
- [httpc] <http://www.microphotonics.com/xraymenu.html>.
- [httd] <http://www.reald.com/scientific/>.
- [IT96] Noro K Inoue T, Kawai T. Performance of 3-d digitizing in stereoscopic images. Japan, Nov 1996. Kanagawa Institute of Technology, Department of Information and Computer Sciences.
- [JA02] Waterworth JA. Dextrous and shared interaction with medical data: stereoscopic vision is more important than hand-image collocation. Creativity Studio Tvistevagen 47, 907 19 Umea, Sweden, 2002. Interactive Institute.
- [Kit03] Kitware Inc. *The VTK User's Guide - Updated for Version 4.2*, 2003.
- [per99] *Vision Science, Photons to Phenomenology Chap 5*. 1999.
- [SML] Will Schroeder, Ken Martin, and Bill Lorensen. *The Visualization Toolkit - Updated for Version 4.2*.