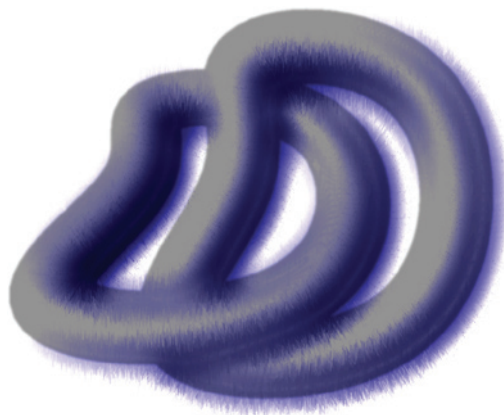


INDUSTRIAL TECHNOLOGY - GRAPHICS INDUSTRIES

2007 HSC MAJOR PROJECT



MANAGEMENT FOLIO



ANDREW HARVEY



This work by [Andrew Harvey](#) is licensed under a [Creative Commons Attribution 2.5 Australia License](#).

<http://creativecommons.org/licenses/by/2.5/au/>

Copyright © 2007 by Andrew Harvey (BOS Student: 17807706)

I can be contacted at andrew.harvey4@gmail.com.

See References section and caption for a list of figures which are owned and/or copyrighted and/or produced by another individual or organisation.

This document sometimes refers to BOS Student: 17807706, this refers to Andrew Harvey. The use of this number is because I could not use my real name when I submitted this document as my Major Project for Industrial Technology.

The following are trademarks or registered trademarks of the associated company:

Visual Basic, Visual C++, Windows, Windows XP: Microsoft Corporation

Pentium: Intel Corporation

Adobe, InDesign, Photoshop: Adobe Systems Incorporated

Solidworks: Solidworks Corporation

Rhinoceros: Robert McNeel & Associates

Autodesk, 3ds Max, 3ds max, AutoCAD, Inventor, Maya, AutoLISP: Autodesk, Inc.

Alias, SketchBook: Alias Systems Corp.

mental ray: mental images GmbH

PTC, Pro/DESKTOP, Pro/ENGINEER: Parametric Technology Corporation

Mathematica, WolframTones: Wolfram Research, Inc.

QuarkXPress: Quark Inc.

IMSI, TurboCAD: IMSI

ACECAD, ACECAT: ACE CAD Enterprise Co., Ltd.

Wacom: Wacom Co., Ltd.

Google: Google

Also many of the products and companies mentioned in the Research section may be trademarks or registered trademarks.

This document was produced for the NSW 2007 HSC, for the Industrial Technology Graphics Industries Major Project, set and marked by the Board of Studies NSW. This course was completed at Blakehurst High School. It was produced in conjunction with a production component. The project was commenced in late October 2006 and completed in June 2007 (due August 2007).

In several production items the word APOLLO is shown as a registered trademark with the symbol ®. The APOLLO word is NOT a registered trademark by me. The registered trademark symbol was added to increase the realism of the production component. No harm or misunderstanding was meant by the use of this symbol in that context.

Published Versions:

August 2007 - Web Edition

June 2007 - Submitted Edition

Printed in Australia.



contents

CONTENTS

- i. CONTENTS -

CONTENTS	III
INTRODUCTION	VIII
STATEMENT OF INTENT	10
RESEARCH	11
SOURCES OF DESIGN RESEARCH	11
Jewellery Shops	11
Magazines	11
Internet	11
Catalogues	12
WATCH DESIGN RESEARCH	13
Internet	13
Magazines/Catalogues/Brochures/Advertisements	17
PRODUCTION/FOLIO RESEARCH	19
Project Structure	19
Rapid Prototyping	19
Presentation Layouts	20
Workflow	21
Printing and Paper	21
DEVELOPMENT OF IDEAS	22
LOGO DESIGN	22
DESIGN DEVELOPMENT	23
DESIGN DEVELOPMENT - IDEAS SKETCHES	23
DEVELOPMENT OF DESIGN/3D MODEL	24
DESIGN PROBLEMS & DESIGN EVALUATION	34
CLIP	34
BAND	36
ERGONOMICS	37
MATERIALS	37
OVERALL SIZE	44
GENERAL DESIGN EVALUATION	44
RELATIONSHIP BETWEEN DESIGN AND PRODUCTION	49
SELECTION AND JUSTIFICATION	50
BRIEF SUMMARY	50
MATERIALS	50
SKETCHING AND DESIGNING	50
STAEDTLER 3H, HB, 3B & 4B Pencil	50
PAPERMATE Pacer (0.5mm leads)	50
STAEDTLER Eraser	51
KENT 30-60-90 Set Square	51
French Curves	51
REFLEX White 80GSM A4 Paper	51
PRODUCTION OF PROJECT	51
REFLEX A4 Inkjet Gloss 150gsm	51
REFLEX White 80GSM A4/A3 Paper	51
Samsung Black and Colour Toner Ink	51
MANAGEMENT FOLIO	51
REFLEX White 80GSM A4 Paper	51



COMPONENTS	51
SKETCHING AND DESIGNING	51
Freehand Sketches	51
PRODUCTION OF PROJECT	51
3D Model	52
Isometric Drawings	52
Oblique Drawings	52
Perspective Drawings	53
Planometric Drawings	53
Rendered Drawings	53
Orthogonal Drawings	53
Physical Model	53
Exploded Drawings	53
Drawing Template	54
Dimetric Drawings	54
Trimetric Drawings	54
Assembly Drawings	54
Sectioned Drawings	54
Stress Analysis Drawings & Data	54
Stereo Images	54
360 Degree Turntable Animation	54
MANAGEMENT FOLIO	54
PROCESSES	55
SKETCHING AND DESIGNING	55
One Aspect at a Time	55
Annotations	55
PRODUCTION OF PROJECT	55
Workflow	55
Templates	55
Layouts	56
Layers in Rhinoceros®	56
Appropriate DPI	58
Production Item Database	58
In-House Developed Software	60
Xrefs	60
Rapid Prototyping	60
MANAGEMENT FOLIO	61
Styles	61
Master Pages (aka. Template pages)	61
Formulae in Spreadsheets	61
RESOURCES	62
SKETCHING AND DESIGNING	62
Grids	62
Graphics Tablet: ACECAD® Acecat® Flair (5" x 3.75")	62
Alias® SketchBook™ Pro 2.0	63
PRODUCTION OF PROJECT	63
Optima WorkPro D9S PC (Microsoft® Windows® XP)	63
Autodesk® AutoCAD® 2007	64
Rhinoceros® 4.0 BETA	64
Autodesk® 3ds Max® 9	65
Adobe® Photoshop® CS2	66
Samsung® CLP-550N Colour Laser Printer	66
Scanner	66
Digital Camera	66
Microsoft® Visual Basic® 2005 Express Edition	67
Autodesk® Impression	68
MANAGEMENT FOLIO	68
Adobe® InDesign® CS2	68
Microsoft® Excel 2003/2007	69
Microsoft® Word 2003/2007	69
Adobe® Photoshop® CS2	70
TIMELINE PLAN	72
GENERAL EVALUATION	75
EVALUATION OF COMPONENTS	75
Statement of Intent	75
Research	75



Development of Ideas	76
Selection and Justification	76
Timeline Plan	76
Finance Plan	76
Compiling the Folio	76
Evaluation	76
Logo	76
OH&S Issues	77
Problems in Production	77
Developed Software	77
Freehand Sketches	77
3D Modelling	77
Orthogonal/Isometric/Oblique/Perspective/Exploded Drawings	77
Rendered Drawings	78
Planometric Drawings/Dimetric/Trimetric Drawings	78
Physical Model	78
Drawing Template	78
Stress Analysis Drawings	78
Quality Control	78

FINANCE PLAN **80**

GENERAL EVALUATION	81
--------------------	----

USE OF APPROPRIATE INDUSTRIAL PROCESSES AND EQUIPMENT **82**

EVIDENCE OF SAFE WORKING PRACTICES AND OH&S ISSUES **84**

COMPUTER RELATED ISSUES	84
Eyesight/Monitor Use	84
RSI/Keyboard Use	84
Seating Posture	85
Power Supply	85
GENERAL ISSUES	86
Printer Ink/Toner	86
Lifting Heavy Objects	86
Fire Extinguisher	86
General First Aid	86

EVALUATION **88**

GENERAL PROJECT EVALUATION	88
Project Evaluation In Relation to Statement of Intent	88
Drawing Production	88
3D CAD/CAM Design Software	88

IDEAS THAT NEVER MADE REALITY	89
Hologram	89
Less Technical User Reference Drawings	89
Real-Time 3D Viewing of 3D Model	89
Dynamic Simulation of Band	90
Rendering in Less Common Projections	90

EVALUATION OF PRODUCTION COMPONENTS	91
Drawing No. 54	92
Drawing No. 75	93
Production Item 76	93
Drawing No. 40	95
Drawing No. 22	95
Drawing No. 55	95
Drawing No. 57, 58, 59...	95
Drawing No. 60	96
Drawing No. 61	96
Drawing No. 56	96
Drawing No. 21	97
Drawing No. 77	97
Drawing No. 73	99
Drawing No. 78	100



Drawing No. 82	100
Drawing No. 79	101
Drawing No. 100	102
Drawing No. 92	104
Drawing No. 95	104
Drawing No. 96	105
Drawing No. 97	106
Drawing No. 98	107
Drawing No. 101	108
Production Item No. 102 (Physical Model)	109
Flaws in Production Components/Product Design	109
APPROPRIATENESS OF DESIGN AND/OR DESIGN MODIFICATION	109
COMMUNICATION TECHNIQUES	110
Report Style Text	110
Desktop Publishing/Non-Fiction Publication	110
Technical Drawings	110
Flowcharts	110
Tables	110
Spreadsheets	110
Gantt Charts	110
Photographs	110
Freehand Sketches	110
Multimedia Video	110
MISCELLANEOUS RENDERINGS	111
EVALUATION OF MANAGEMENT FOLIO	115
ONGOING EVALUATION	115
EVALUATION FROM INDUSTRY	117
EVIDENCE OF INDUSTRIAL PROCESSES/ TECHNOLOGIES	118
SOLUTIONS TO PROBLEMS IN PRODUCTION	130
COMPUTER/SOFTWARE PROBLEMS	130
COMPUTER CRASHES	130
SOFTWARE CRASHES	130
COMPUTER VIRUSES	130
MANAGEMENT OF DATA	131
CREATING MESHES	131
GRAPHICS PROBLEMS	131
OBLIQUE CIRCLES	131
ISOMETRIC PROJECTION VS ISOMETRIC DRAWING	132
REPRESENTATION OF FILLETED EDGES	133
SCALES & AS1100	133
COLOUR SPACE	134
COMPUTER RENDERING PROBLEMS	134
TOO SLOW	134
PATH NOT FOUND ERROR	136
APPENDIX A - DEVELOPED SOFTWARE	138
OBLIQUE PROJECTION TOOL	138
VISUAL BASIC® 6 USER INTERFACE	138
VISUAL BASIC® 6 CODE	138
DERIVATION OF OBLIQUE TRANSFORMATION EQUATION ¹	139
EVALUATION	140
VIEWIT	141
VISUAL BASIC® 2005 USER INTERFACE	141



BASIC PROGRAM STRUCTURE	143
LIST OF SUBS AND CUSTOM FUNCTIONS	143
process()	143
getVP()	143
dxfread()	143
circletolines(...)	143
arctolines(...)	144
testQ(...) (Function)	144
csvread()	144
rawread()	144
plyread()	144
dxfwrite()	144
csvwrite()	144
coordsys()	144
frontface()	144
rotateazel()	145
proj()	145
perspproj()	145
btnDefaultAxonScales_Click()	145
BackgroundWorker1_DoWork()	145
BackgroundWorker1_RunWorkerCompleted()	145
PICTORIAL DEVELOPMENT OF VIEWIt	145
USER REFERENCE	145
VISUAL BASIC® 2005 MAIN FORM CODE	163
EVALUATION	194
PROBLEMS AND SOLUTIONS	195
Multithreading	195
Drawing Exchange File (DXF)	195
DXF Basic Format Overview ¹	195
Multipurpose Projection Matrix	196
Derivation of Projection Matrix	196
One Point Perspective Algorithm	198
Importing DXF Circles & Arcs Problems	201
Drawing of Arcs	202
Calculation of Axonometric Scales	208
POSSIBLE IMPROVEMENTS	208
Hidden Line Removal	208
Two Point Perspective Algorithm	209
Raster Image Support	214
Avoid Sampling Circles When Unnecessary	214
DRAWINGS CREATED USING VIEWIt	214
REFERENCES	220



INTRODUCTION

I inserted an introduction here because I needed to address how I have structured parts of this *management folio*. I have not included all the things set out in the *assessment criteria* (see figure ii.1) under separate chapters. This is because I incorporated certain things together. I have followed the *design and management* section as specified, however this is not the case for the *workplace communication* section.

'Documentation of the major project from conception to completion' and even the 'evidence of ongoing evaluation' could be considered the "diary" that many of the sample major projects have. I have not included a "diary" of such as I found that everything that was in a diary can be found in other parts of this folio. This "diary" or 'evidence of ongoing evaluation' can be found in the evaluation section where I evaluate some of my production components from start to finish, including how I went about completing that component. In this evaluation section I also document general project evaluation, project evaluation in relation to my statement of intent, drawing production, 3D CAD/CAM design software, ideas that never made reality, flaws in the production components and product design, communication techniques, evaluation of management folio, and evaluation of my production components from industry personnel.

'Appropriateness of design and/or design modification' can be found in the *development of ideas* section. 'Evidence of a range of communication techniques' is documented in the evaluation chapter, but is evident in this folio and production components. 'Evidence of a range of computer applications, eg word processing, spreadsheets, CAD, multimedia' can be found by looking at this folio and the production components.

Assessment criteria

Design and management

- statement of intent
- research
- development of ideas
- selection and justification of materials, components, processes and other resources
- timeline plan – projected order of production and estimate of time allocation
- finance plan – projected cost of materials and services (if applicable)
- use of appropriate industrial processes and equipment
- evidence of safe working practices and OH&S issues

Workplace communication

Documentation of the major project from conception to completion including:

- evidence of ongoing evaluation
- appropriateness of design and/or design modification
- student's evaluation of the major project and its relationship to the statement of intent
- evidence of a range of communication techniques
- evidence of a range of computer applications, eg word processing, spreadsheets, CAD, multimedia

Figure ii.1 - Design, Management and Communication Assessment Criteria
(An extract from a NSW Board of Studies Publication)

To fulfil the assessment criteria of the production component (see figure ii.2) a range of things needed to be done. Firstly high *quality of the product* was necessary which I stated in my statement of intent and I made sure of by employing the best software solutions and processes that were available. To further insure my product (production items incorporating a *set of related drawings*



around a product) was of high quality I implemented quality control measures and strategies, which were carried out on my product throughout various stages of production.

Evidence of a range of skills was also necessary, so to ensure that I had evidence of high quality in the application of a wide range of skills and techniques in the planning and production of the major project I produced a wide range of production items with the use of a wide range of processes and techniques in management folio, design development and production components. The extent of the skills applied can be seen in the 3D stereo images, 3D model, physical prototype, rendered images, composite images incorporating the elements of design, parts list, template design, watch design, freehand sketches, programming of software, scripting inside existing software, collaboration of this folio, perspective drawings, isometric drawings, trimetric drawings, dimetric drawings, oblique drawings, orthogonal drawings, planometric drawings, sectioned drawings, exploded drawings, assembled drawings, and a combination of these, multimedia presentations and documentation techniques plus many many more.

Degree of difficulty was covered by choosing to produce production items based on a watch, which has many curves and much detail, especially in the clip.

Evidence of industrial processes and use of industrial technologies can be seen in my major project and are mainly documented in the Evidence of Industrial Processes/Technologies section but can also be seen in the Selection and Justification of Processes and Resources section.

Links between planning and production are shown by the matching statement of intent with the actual completed project and with the comparison of the production component and the development of ideas section.

Evidence of solutions to problems in production can be seen in the solutions to problems in production section.

Assessment criteria

- quality of the product
- evidence of a range of skills
- degree of difficulty
- links between planning and production
- evidence of industrial processes
- use of appropriate materials
- use of industrial technologies
- evidence of solutions to problems in production

Figure ii.2 - Production Assessment Criteria
(An extract from a NSW Board of Studies Publication)

Appendix A - Developed Software is an appendix for a reason. It was placed there to show the technical details of exactly how the software I developed worked and what it did. It is also there to show the range of skills I have. However Appendix A deviates quite reasonably from the syllabus. Although it is strongly related to graphics, it is more related to the area of computer graphics and mathematics rather than technical drawing. As it is less related to the syllabus, it is an appendix not a chapter like the rest.



STATEMENT OF INTENT

I intend to design, document and produce high quality drawings and other production items of a consumer product, a wristwatch.

I intend to design my watch based on existing styles and products. But I also intend to implement some new and innovative features. When designing the wristwatch I intend to consider and investigate by using research and design development sketches the following aspects:

- **Function** – I must first investigate what the wristwatch will be used for, what purpose will it serve, and how it will be used. Knowing the function will help me to make a design that will work. That said I will design a watch that has functionality first and aesthetics second.
- **Aesthetics** – I will attempt to make my design modern and appealing to my target audience of young to middle-aged men. I will consider shape, texture, colour, form and materials in my design to make it as aesthetically pleasing as possible. I intend to design a watch that looks elegant, expensive and of premium quality.
- **Ergonomics** – I will attempt to design a product that will fit on all people, given that everyone has different shaped and sized wrists.
- **Safety** – I will attempt to produce a design with minimal risk of harm to person using the watch.
- **Durability** – I will attempt to make a design that will last a long time and not break.

I also intend to produce in-depth documentation of all aspects of the project. I will do this mainly through the Management Folio. This documentation will allow me to evaluate and say why and how I did things related to the production and management of the project and of the folio. This documentation also allows me to plan things such as finances and time management and also to keep a record of the project.

In addition to designing and documenting, I also intend to produce drawings and other production items of the wristwatch that I designed. I intend to produce clear, accurate and neat drawings, which will convey enough information about the product so that it can be manufactured. This will be done mainly through orthogonal, isometric and assembly drawings. I also intend to show information in my drawings that will allow it to be marketed to the public. This will be done mainly through rendered drawings, presentation layouts and physical models. In addition I intend to produce drawings which will show how the watch operates, this will be done mainly through rendered animations.

My main reasons for choosing to design and draw a wristwatch are because I wanted to produce something that was difficult, and once finished would look professional and of a high standard. Also a wristwatch can show my full potential and range of skills, knowledge and problem solving ability.



RESEARCH

I used research for two parts of my project. Being the design of the wristwatch and for which materials, components, processes and resources to use for the production of the drawings. The types or forms of research that I used included internet, visiting shops, talking to others to gather their ideas, opinions and knowledge and watch advertisements in magazines. Research was also used to find partial or full solutions to problems that I encountered during production. Research was also used in the production of *ViewIt*, see Appendix A → *ViewIt*, page 141. Not all my research is documented here, some small notes and footnotes can be found in other parts of this folio which mention the source of my research for that problem or item.

SOURCES OF DESIGN RESEARCH

Here I used research to get ideas of the design of my wristwatch from existing models of watches and what other people had to say about watches. These ideas were related to look, shape, colour and functionality, and the ideas gathered through research helped me with the design process. I mainly researched into the type of watch outlined in my statement of intent, a men's watch of premium quality.

I talked to many of my friends and people I knew from a range of backgrounds, asking them things such as what features they thought a watch should have, what should it be made of, how should the fastening mechanism work, what shape it should be etc. From this oral research I immediately made freehand sketches of ideas that they recommended. These can be found in the Design Development Sketches folio. Initially I had planned to additionally use surveys conducted by *CHOICE* magazine, however when I went to their web site I was unable to find anything on watches.

JEWELLERY SHOPS

I visited shops in order to see aspects of the watch that I couldn't see from a standard image. The shops I visited include:

- Goldmark Jewellers, Westfield Hurstville
- Prouds Jewellers, Rockdale Plaza
- Angus & Coote, Westfield Hurstville

In particular I looked at the different fastening devices already on the market, which was a lot easier to grasp than looking at a still image. The second main thing I gathered from this research was the sizes of watches at a real life scale. This was done at the shop as I could see the actual size, whereas when looking at a watch in a still photograph the size is hard to visualise.

MAGAZINES

The magazines where I collected images of watches from advertisements from include:

- "The (sydney) Magazine" from *The Sydney Morning Herald*, Issue #43, November 06

I also used these advertisements to get ideas for my own presentation renderings. In particular for the composition of the different elements of the image.

INTERNET

The manufacturer's web sites that I looked at to obtain design information and ideas include:

- TAG Heuer - www.tagheuer.com
- Casio - www.casiowatches.com.au
- Rolex - www.rolex.com
- Swiss - www.fhs.ch
- Seiko - www.seikowatches.com
- Omega - www.omegawatches.com
- Swatch - www.swatch.com
- Citizen - www.citizenwatch.com
- Breitling - www.breitling.com
- Raymond Weil - www.raymond-weil.com





Figure 2.1 - TAG Heuer Website
www.tagheuer.com

I also used Google Image Search to gather watch images.

CATALOGUES

- Although *Rip Curl* watches are generally sports watches and not the type of watch I intend to design, the 06/07 catalogue did prove useful.



WATCH DESIGN RESEARCH

INTERNET



- ✓ The compact body
- ✓ The watch hands and the minute markers
- ✗ The leather band
- ✗ Too much detail on the watch face

Figure 2.2 - TAG Heuer SLR
www.tagheuer.com



- ✓ The metallic band
- ✓ The band style
- ✗ Red colour (it does not relate to the aesthetics intended in my statement of intent)
- ✗ Spinning timer (it is too big)
- ✗ Use of plastic

Figure 2.3 - TAG Heuer Formula 1
www.tagheuer.com





- ✓ The layout of the watch face is simple and looks sleek
- ✓ The white watch face
- ✓ High precision of the watch hands along with the interval marks
- ✗ Gold colour, it makes it slightly feminine
- ✗ Watch band design, the strokes on the edge don't look good

Figure 2.4 - Omega De Ville Co-Axial Chronograph
www.omegawatches.com



- ✓ The layout of the watch face. It is simple and looks sleek.
- ✓ Blue colour scheme
- ✓ High precision interval marks
- ✓ Watch band
- ✓ Date is shown
- ✗ The fat watch hands
- ✗ Too many buttons on the side

Figure 2.5 - Omega
www.omegawatches.com





- ✓ It looks rigid and the band will only flex in one dimension
- ✓ The use of roman numerals makes it look of a high standard, which makes it look elegant and premium quality, the same as what I intended in my statement of intent
- ✓ The use of texture on the watch face
- ✓ Date is shown
- ✗ Date is shown in a bad place

Figure 2.6 - Raymond Weil Geneve - Parsifal
www.raymond-weil.com



- ✓ This shows me that leather bands can look good, and also that the fastening device shown is effective.

I used this image to develop ideas sketches of leather bands and the fastening device used.

Figure 2.7 - Leather Band and Fastener
Obtained from Google™ Image Search
<http://www.strappedfortime.com/images/Img1281.jpg>





Figure 2.8 - Stainless Steel Band
Obtained from Google™ Image Search
http://i9.ebayimg.com/02/i/06/2d/77/0e_1_b.JPG

- ✓ This gave me ideas of stainless steel bands.



Figure 2.9 - TAG Heuer
www.tagheuer.com

- ✓ This watch gave me inspiration for the design of my spinable timer device
- ✗ There are no markings on the watch face. This would make it hard to tell the time accurately and although it does make it look aesthetically pleasing, I intended to put function first and aesthetics second as mentioned in my statement of intent.



MAGAZINES/CATALOGUES/BROCHURES/ADVERTISEMENTS



Figure 2.10 - Various Brands (Source: MYER Brochure)

✗ I didn't like any of these watches. Their function was more fashion rather than to tell you the time. Although they look very aesthetically pleasing, this is not the kind of watch I set out to design in my statement of intent.



Figure 2.11 - G
(Source: Goldmark Brochure)

- ✓ The link between the body and the band
- ✓ The spinning timer
- ✓ Date is shown
- ✗ No labels for the time
- ✗ The pattern on the band doesn't look good





Figure 2.12 - Raymond Weil: Geneve (Advertisement from *The (sydney) Magazine*)

- ✓ The general style is elegant, which I wanted as specified in my statement of intent.
- ✗ Hard to quickly determine the time
- ✗ The date is small and hard to see
- ✗ I don't like the square shape



Figure 2.13 - TAG Heuer: Link (Advertisement from *The (sydney) Magazine*)

- ✓ The band looks elegant and stylish
- ✓ Date is shown
- ✗ The knob to change the time is too big
- ✗ The ring of diamonds gives it more of an aesthetic aspect and less function, which is the opposite to what I specified in my statement of intent.
- ✗ Integration of the band and watch body is lacking. Although it is better than other designs it is still not at the level I want.





Figure 2.14 - Rip Curl: Barrel (Rip Curl 06/07 Catalogue)

- ✓ The shape of the spinable timer
- ✓ There is a high level of integration of the watch band and body, which was reflected in my final design.
- ✗ Although the blue background looks aesthetically pleasing, it makes it difficult to quickly and easily see the time, again this deprives the function, which was mentioned in my statement of intent.
- ✗ There is no incorporation of the date
- ✗ The change time knob is in an awkward and unergonomic position

PRODUCTION/FOLIO RESEARCH

In this section I have used a variety of methods and mediums of research. Such as Internet, CD-ROM packages, reports, telephone, verbal communication, publications and non-fiction books.

PROJECT STRUCTURE

Project structure research includes researching about what was required for the management folio and production component, and to what quality they needed to be produced at. I did this by looking at some past major works, which I found at school in the store room. I used this as research as it was a readily available method to see what was required. I also used the Major Project Examiners Report from 2002 - 2006, and by using these I had a better understanding of what was needed and I found that these reports helped guide me through my development of both the management folio and the production components. I also used the Industrial Technology Standards Package (2002) CD-ROM which helped me a lot by giving me examples of what was required for a band 6. I also asked my teacher lots of questions about the structure and layout of the management folio, this along with a poster I found in the Graphics room at school which helped guide me through my development of the management folio. I found this research was extremely valuable to me, and was worth using.

RAPID PROTOTYPING

I had found out about this technology a long time ago, and I knew that it could be done. However, I had no idea if there were any local companies that offered this service, and how much it would cost. So the first thing that I wanted to research about this was how much approximately it would cost. Next I researched which local companies offered this service, and then also, how does it exactly work.

Firstly I did some searches on the internet about this, and found some companies, however I wanted Australian ones, and I could only seem to find International companies. In the end I found the company Intercad, who are leading suppliers of mechanical CAD/CAM & CAE solutions, by just finding it without even looking for it. It was when I was actually looking for resellers of Solidworks that I found Intercad, and then I saw that they offered rapid prototyping solutions. I found Intercad by refining the location of sales companies at www2.rhino3d.com/sales/.

From the Intercad web site (www.intercad.com.au) I found the table shown in figure 2.15.

Part Volume	Wax Finish	Resin Finish
Small 100 x 100 x 50	\$145	\$249
Medium 100 x 250 x 50	\$360	\$500
Large 200 x 250 x 100	\$1,360	\$1,600
Full 200 x 250 x 200	\$1,990	\$2,250

Figure 2.15 - Price Guide
Source: www.intercad.com.au



- 2. RESEARCH -

Now I had a company and some approximate prices. I knew that my part would be less than 100x100x50mm in size, so I knew that AU\$145 or even maybe AU\$249 was reasonable and I could afford it and this technology was one that was worth considering some more.

Now that I had an address I used www.whereis.com to find exactly where the place was located. This is shown in figure 2.16 and as you can see the place was too far away from Sydney CBD for me to actually visit. So instead, I used e-mail and telephone calls to get further information about the services that they offered. Upon doing this I found that they no longer offered their 3D printing service, so I look elsewhere for other companies.

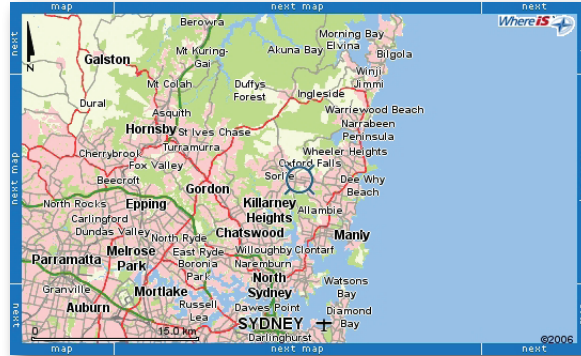


Figure 2.16 - Intercad's Office Location
Source: www.whereis.com

Through telephoning companies and finding more companies I found 3H Engineering Pty Ltd. I got a quote for the 3D printing of part of my model. A copy of the e-mail can be found below in figure 2.17, and reply in figure 2.18. I outsourced the 3D printing of my model to this company. Details can be found in the References section, page 220 and in figure 2.18.

(image unavailable)

(image unavailable)

Figure 2.17 - Quote Email

Figure 2.18 - Reply Email

PRESENTATION LAYOUTS

I knew that I wanted some less technical drawings as part of my production. This includes drawings like rendered perspectives. However I also wanted to combine multiple drawings and text into a series of presentation drawings. I got the idea for doing this, and how I could do it, from publications made by the Faculty of the Built Environment: University of New South Wales. In particular the Bachelor of Industrial Design Graduation Exhibition from 2004 and 2005. From this research I have gained ideas about how I can mix drawings together and create presentation documents, which can help fulfil part of my statement of intent about showing information that allows the product to be marketed.



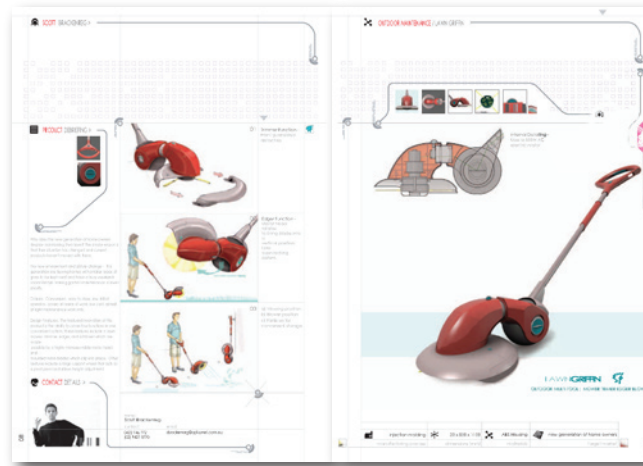


Figure 2.19 - Presentation Layouts

(Reproduced from "perform UNSW Faculty of the Built Environment Graduated Exhibition 2004 bachelor of industrial design" ISBN 0 7334 2170 9)

WORKFLOW

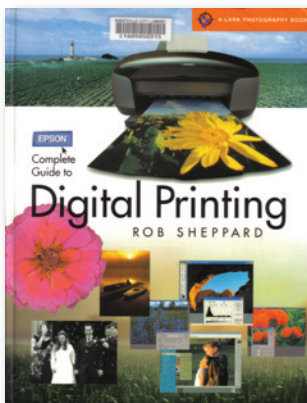
I talked to someone who is in their second year of Bachelor of Industrial Design at University of New South Wales about the workflow taken to design a product. I conducted this research through a one on one verbal conversation, and the information that I obtained through him was that it was standard practice to make a computer generated 3D model of a product and then have the software develop other 2D views of it. This person that I talked to specifically mentioned Solidworks, and although that I didn't use this software because I couldn't obtain it, I discovered that this workflow was commonly used in industry and this was what was gained from this research -knowledge of industry practice-, so I decided to do my project in this way. I don't have this person's contact detail's because I only got to talk for about 10 minutes during the University open day.

After becoming aware of Solidworks, I did some research about it through their web site, www.solidworks.com. Through promotional videos on their web site I saw the full potential of the software.

PRINTING AND PAPER

Printing was a major component of my project as most of my project was done on computer and it needed to be presented on a tangible medium. Both the management folio and the production components needed printing, and the printing processes used can make the product look professional, if it is done right, or terrible if it is not done correctly. Prior to starting my major work I only had limited knowledge about printing. For example I knew a little bit about DPI but that was all I knew. I knew that if I wanted my project to look professional, as outlined in my statement of intent, I would need to know in depth information about printing processes.

I gathered that I would need to research printing. So I visited my local library and found a book about Digital Printing. It talked about DPI, different model printers, paper, colour systems, etc. The details of the book are listed below.



Complete Guide to Digital Printing
Rob SHEPPARD
Lark Books
ISBN 1-57990-427-0

This book help guide me through the production of my major project and allowed me to get the best possible printing job of my management folio and certain production components.

I also looked at Reflex's web site (www.reflex.com) to find out about the different types of paper available.

Figure 2.20 - Digital Printing Book

(Source: Lark Books, © Lark Books)



3 DEVELOPMENT OF IDEAS

LOGO DESIGN

For my DD logo, I just drew a random spline¹ (see figure 3.1) and it ended up looking like a "D", so I made two of them, modelled them (see figure 3.3), rendered them (see figure 3.2), and applied some effects in Adobe Photoshop (see figures 3.4, 3.5, 3.6, 3.7, 3.8).

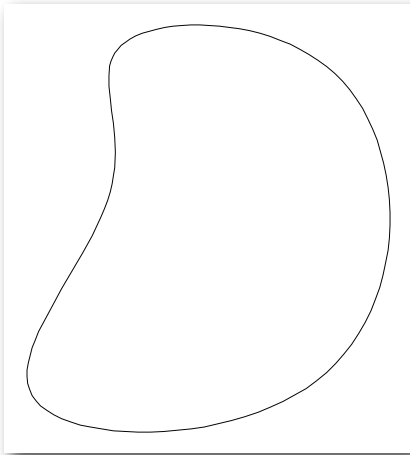


Figure 3.1 - Original spline

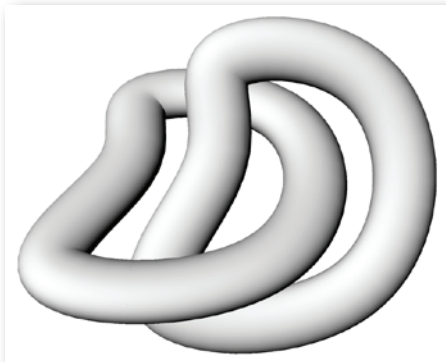


Figure 3.2 - Simple rendered DD

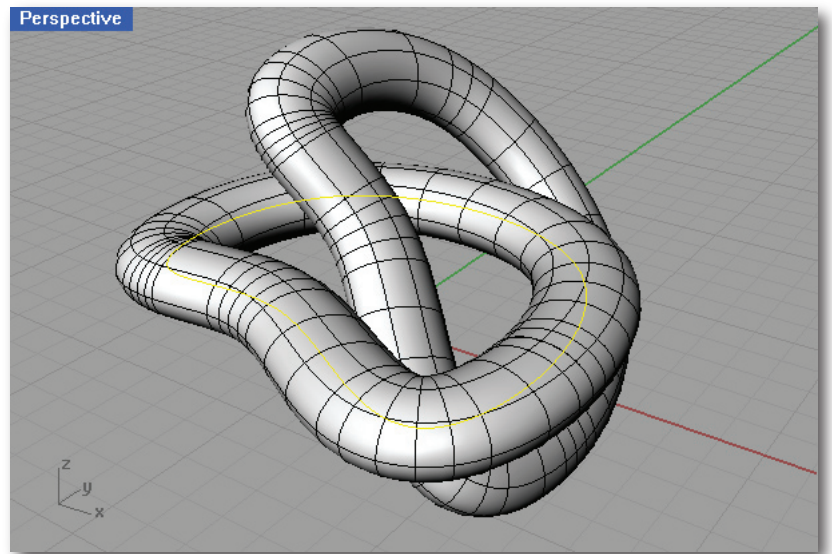


Figure 3.3 - Rhinoceros® model of DD logo

All my ideas for my logo are based on the rendered logo (figure 3.2). I then tried different things and applied different effects in Photoshop to obtain different results. In the end I had made several designs and then I chose the one that I liked best.



Figure 3.4 - Idea 1



Figure 3.5 - Idea 2



Figure 3.6 - Idea 3



¹ Curve



Figure 3.7 - Idea 4

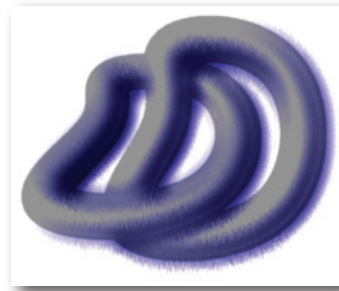


Figure 3.8 - Idea 5 (Final)

The selection of my final logo was based on looking at all my designs and choosing the one that I found most aesthetically pleasing. My development of ideas of my logo was influenced by the time I had to complete the logo, see Timeline Plan → Evaluation of Components → Logo, page 76. Another factor which made the creation of the logo difficult is that the logo must be able to look very much the same and good whether it is the size in the footer of this page, or the size of a small icon on a computer, or the size of an A3 piece of paper, or the size of a billboard (I did not need to meet the billboard size requirement though).

DESIGN DEVELOPMENT

Before I could begin production I needed to design my watch. I used freehand sketches to do this. These sketches were only rough, however they had enough detail to begin modelling the design in 3D. In these sketches I only made rough dimensions, if at all. Much of the fine and technical detail needed for manufacture was added during the 3D modelling stage. The conventional method of design involves developing ideas on paper, then doing final sketches of the product, then making 2D drawings of it. However as I planned (and did) on using a 3D modelling package in the middle of these two stages I did not need to have the design finalised with every detail shown in my concept sketches. Once I had a basic idea of what I was going to design, and once these ideas were sketched on paper, I could continue developing ideas throughout the modelling process. If I didn't like a change, then I went back a few steps and removed it, and if I thought that something should be added, I added it. So to me the 3D modelling stage proved an integral part of my design development and as I explain later, also an integral part of my actual production.

The sections that follow include; my initial design development sketches and any documentation that go with them; development of the 3D model and the documentation that accompanies it; and lastly design problems and evaluation of my final design once modelling was complete.

DESIGN DEVELOPMENT - IDEAS SKETCHES

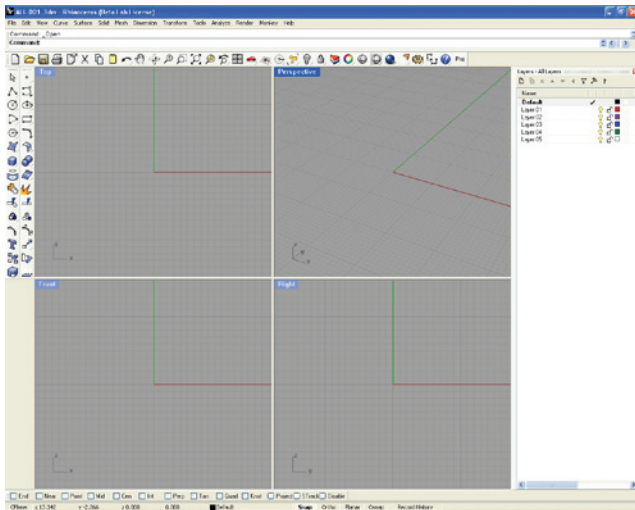
All of my design development sketches can be found in the *Design Development Sketches* folio. Please refer to that folio.

Modifications that I made to my design along the way are evident in these sketches. For example the original design idea for the timer was for many dense grooves, but then I decided to change this to a bevelled design for aesthetic reasons, but then I changed to the final design for a combination of aesthetics and ergonomics. Similarly with the hands, my original design was stick like for functionality, but I then changed it to a triangular design for aesthetics and to relate to my intent. Evidence of other modifications to the design can also be seen by comparing my final design sketches to the product.

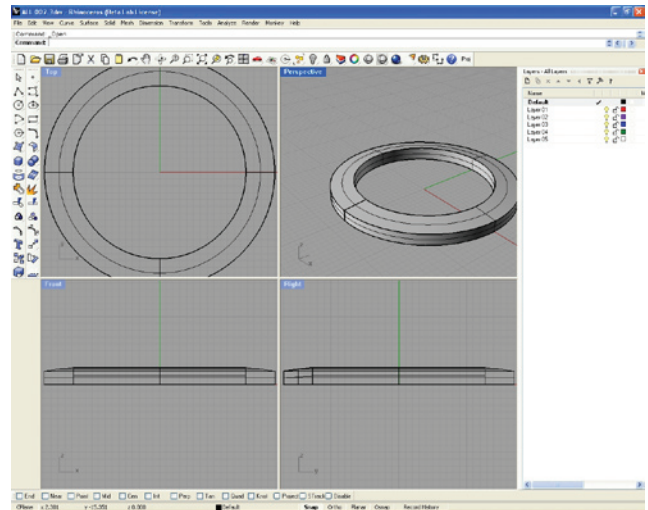
These sketches were done to help me make the 3D model, which I would make directly after the sketches, so accuracy was not an important factor, but rather the overall structure and working. I could use the 3D modelling stage to refine and ensure that my design was correct.



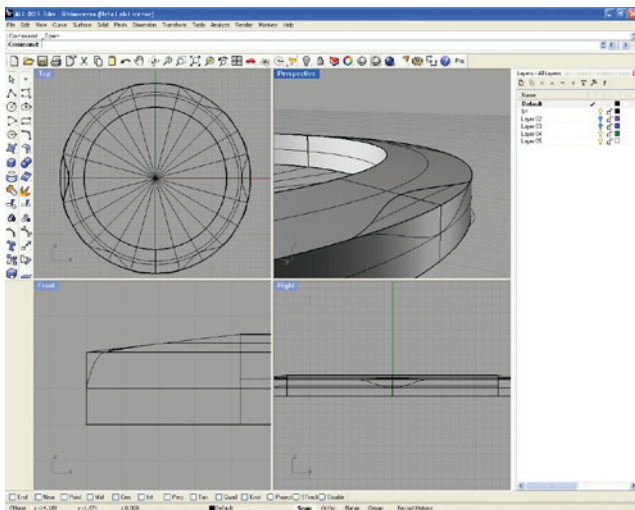
DEVELOPMENT OF DESIGN/3D MODEL



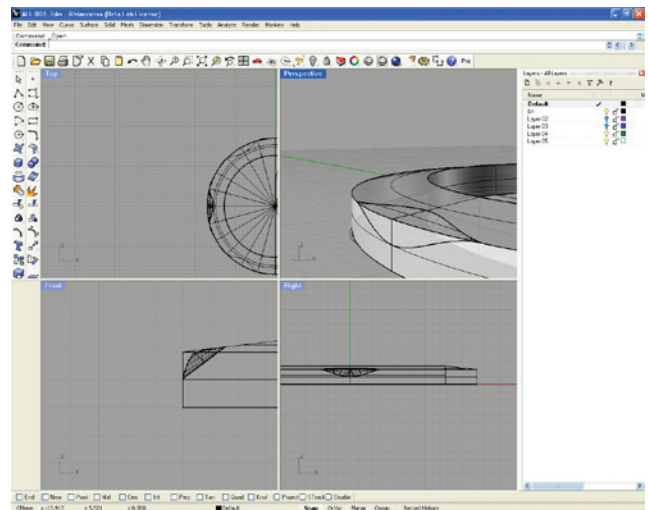
A blank beginning...



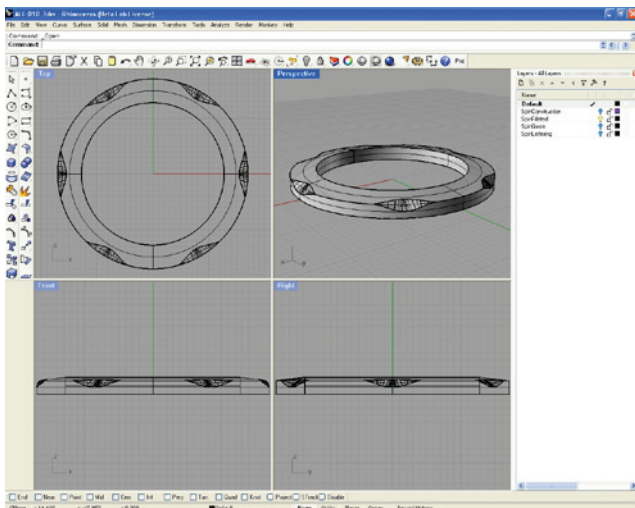
I model out the basic shape of the timber.



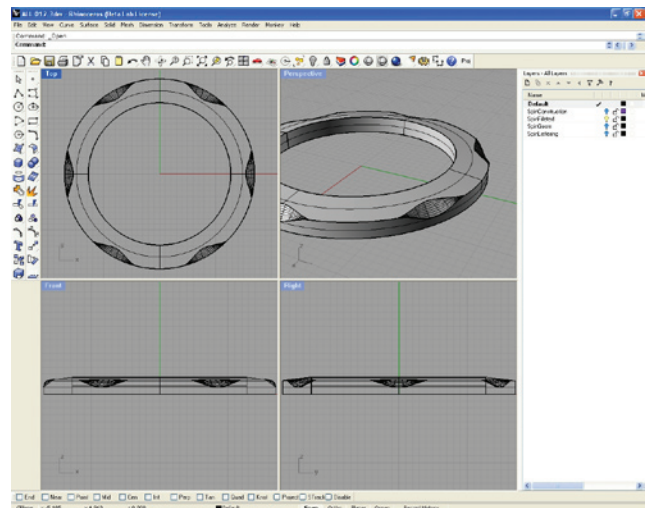
I add curves to define the grooves.



I interpolate a smooth surface between the two curves.

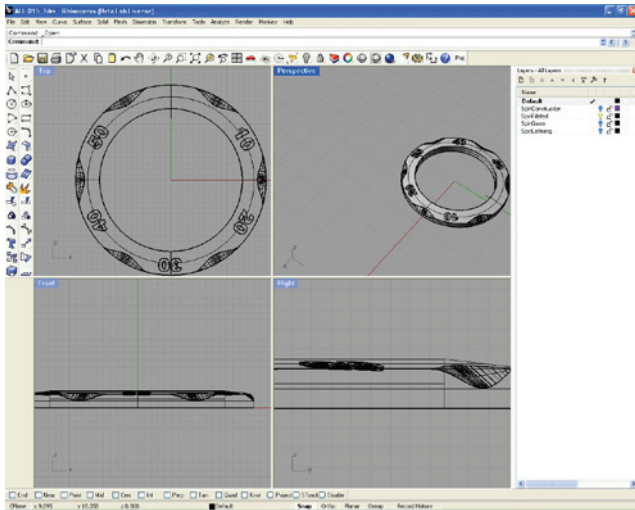


I trim the two surfaces.

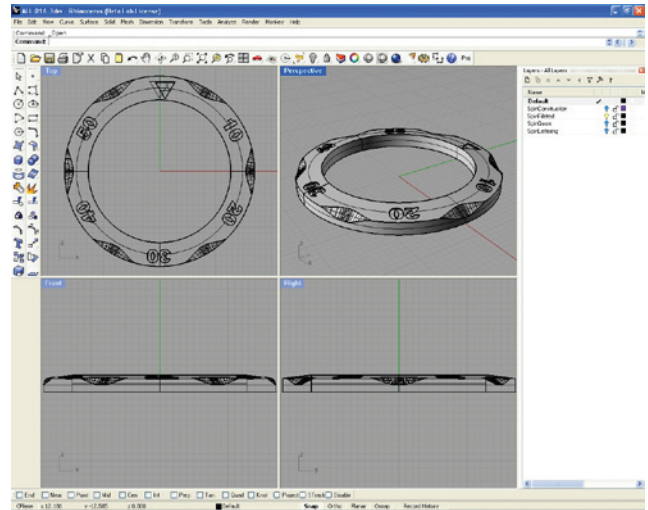


The surface is refined.

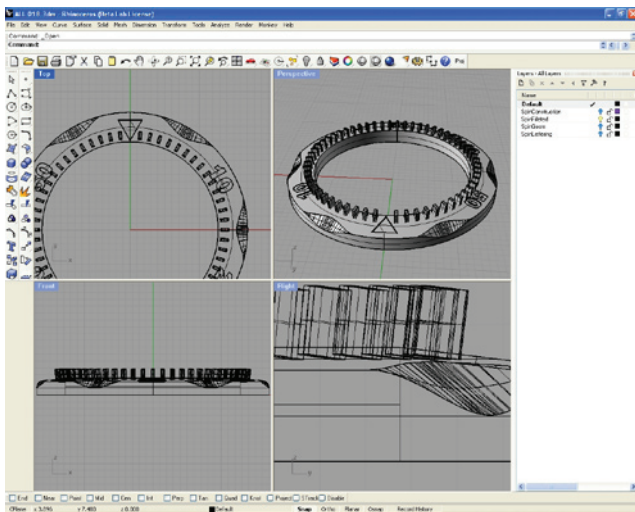




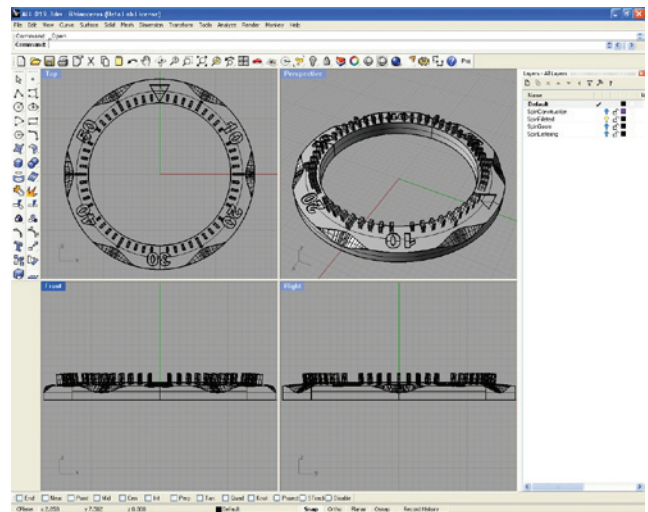
I cut away the 10 minute markers.



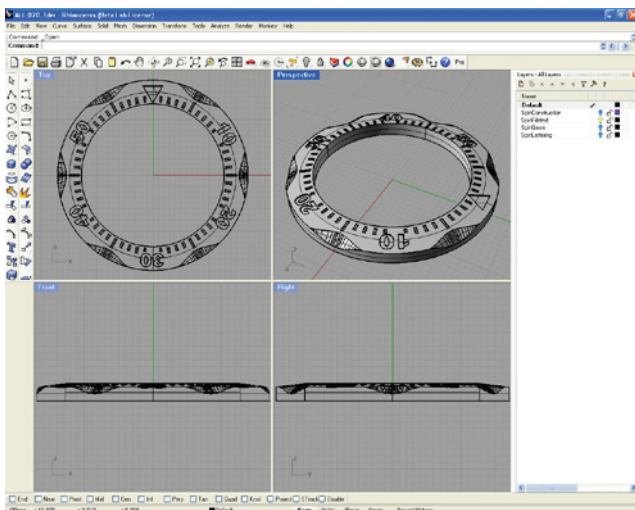
I add the top triangle.



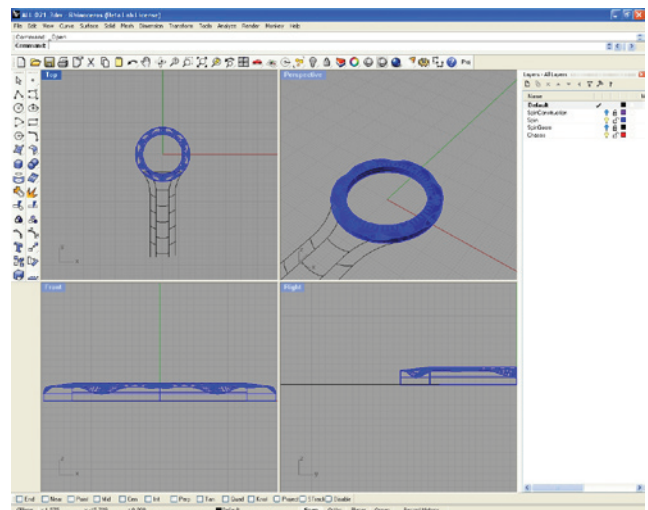
The minute markers are added.



I added the 5 min markers.



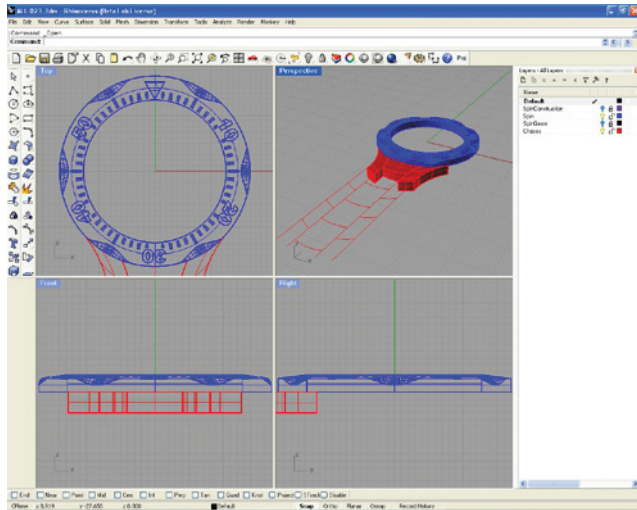
I subtracted the markers from the timer.



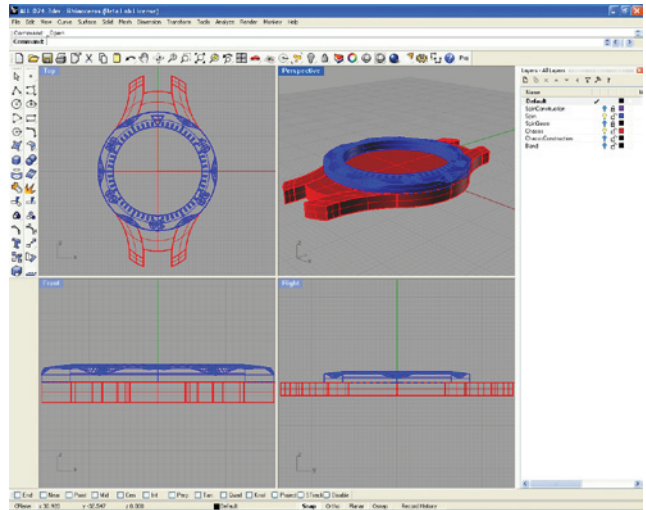
I draw in the 2D projected lines for the band.



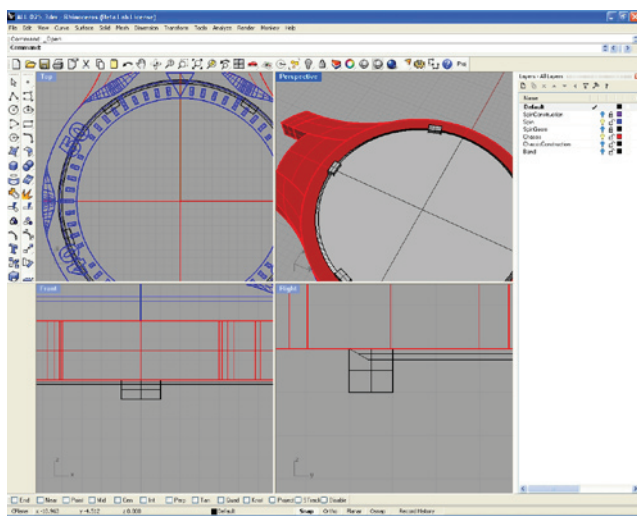
- 3. DEVELOPMENT OF IDEAS -



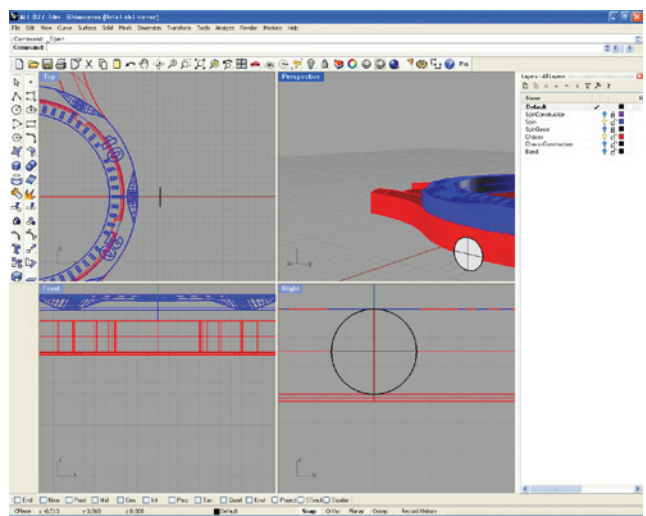
I extruded the curves that connect the watch body to the band.



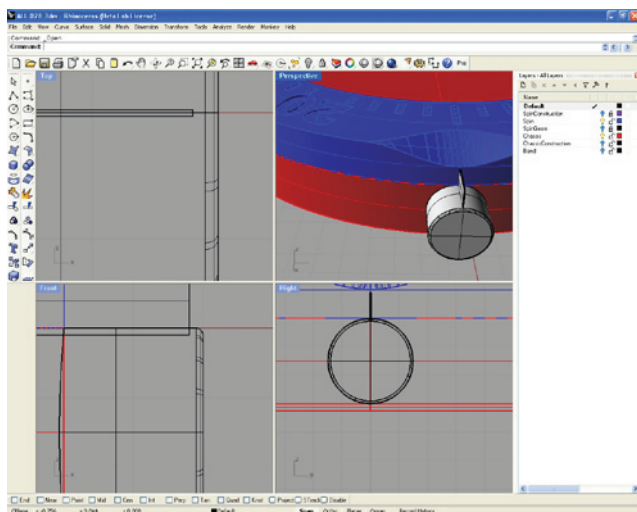
I make the watch body.



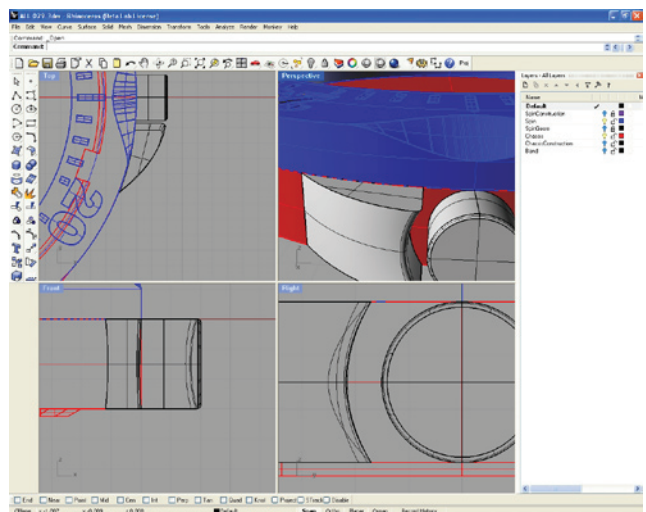
I make the cover that goes on the back.



I begin to make the change time knob.

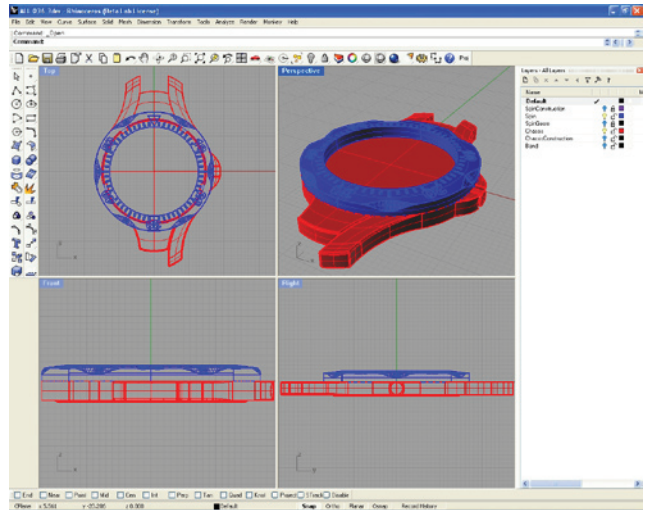
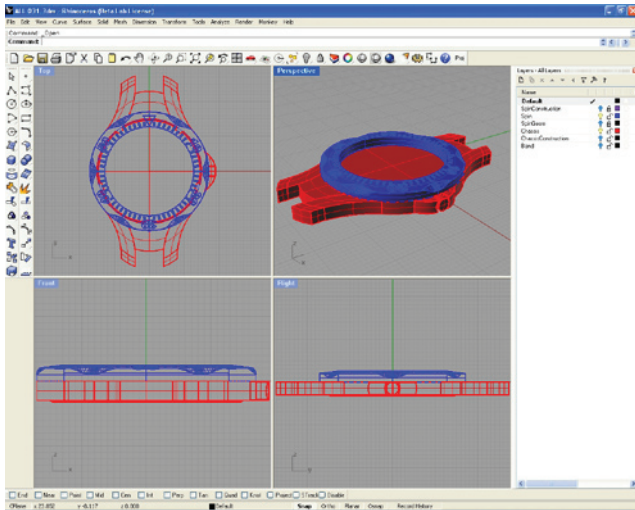


I extrude the change time knob.

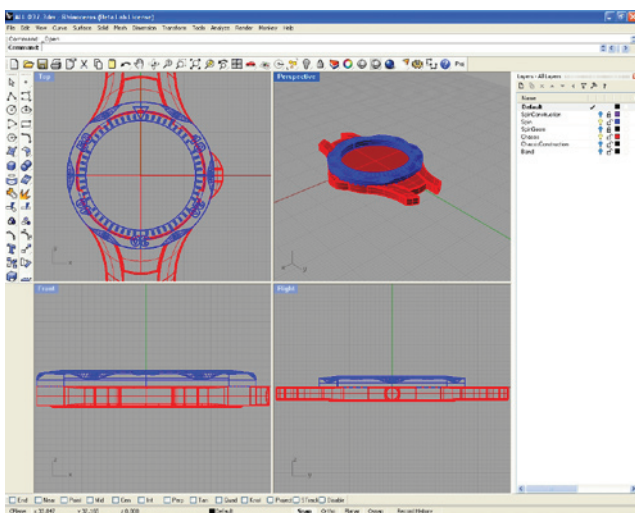


I make the casing for the change time knob, by lofting 3 curves.

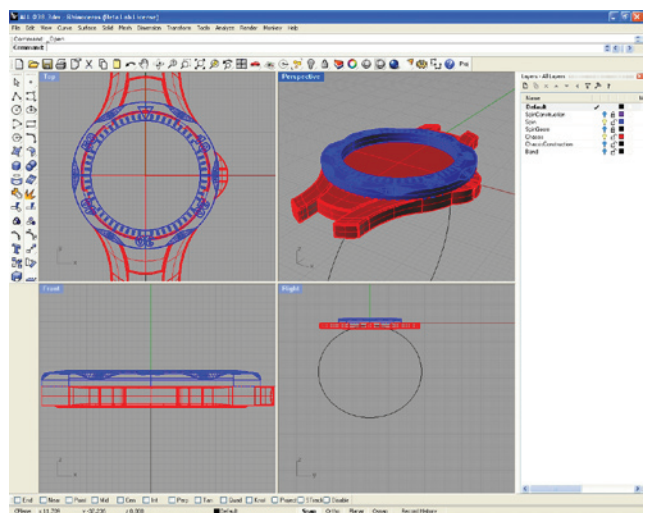




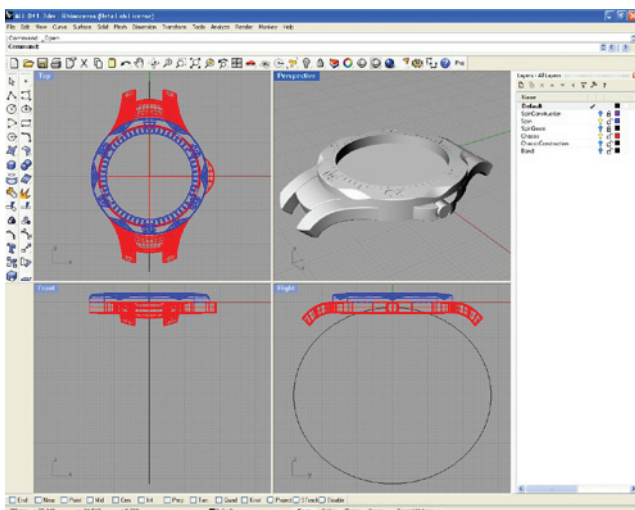
I fillet the edges of the part that connects the body with the band.



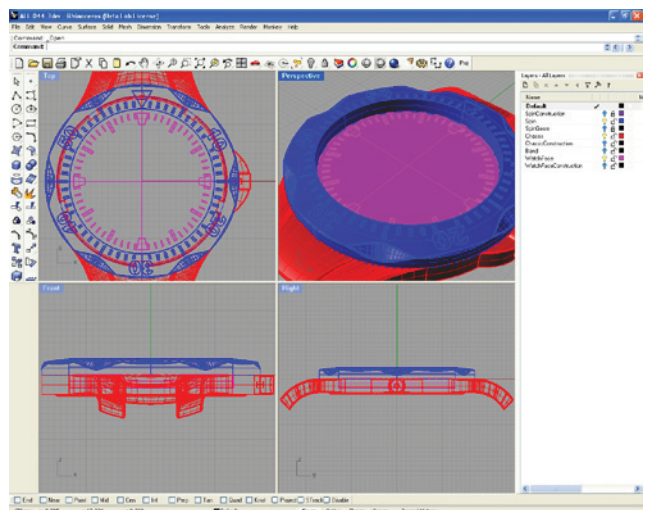
I use the mirror tool to apply this to all the parts easily.



I make the curve that will be the basis of the band.



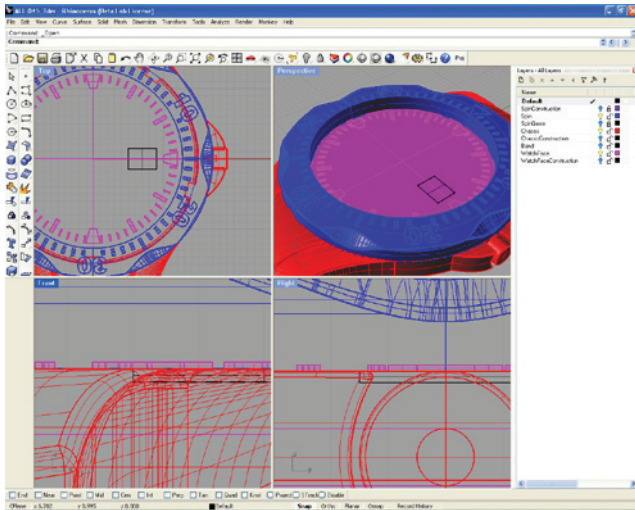
I bend the part that connects the body to the band.



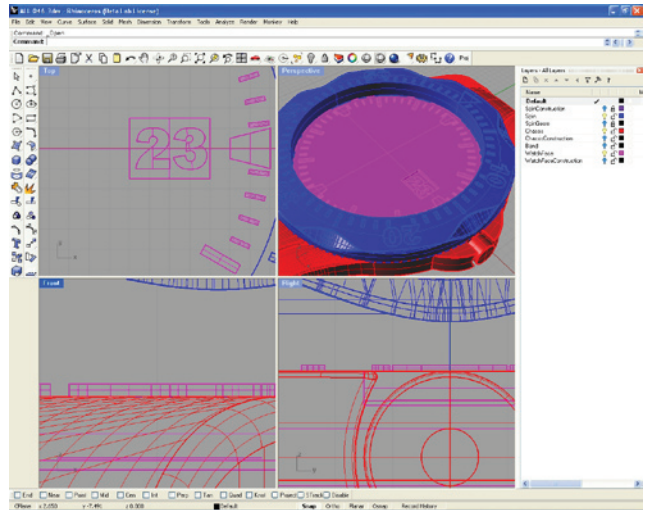
I add the watch face markings.



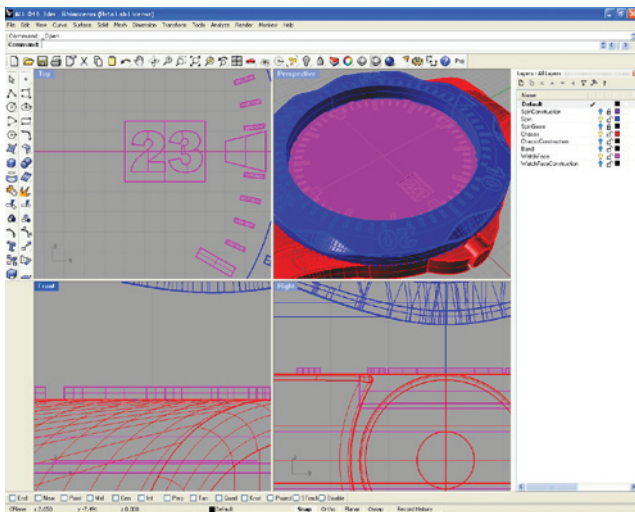
- 3. DEVELOPMENT OF IDEAS -



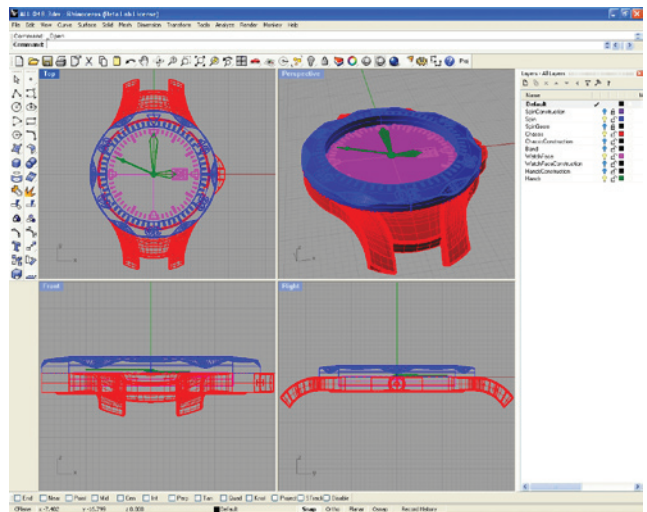
I add the date cavity.



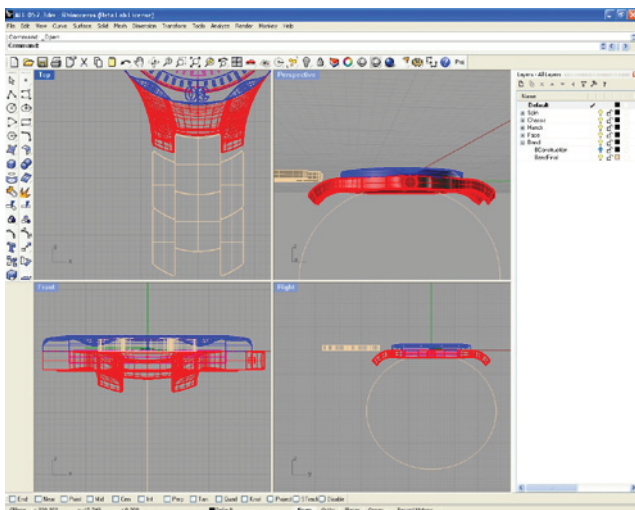
I add the surface of the date numbers.



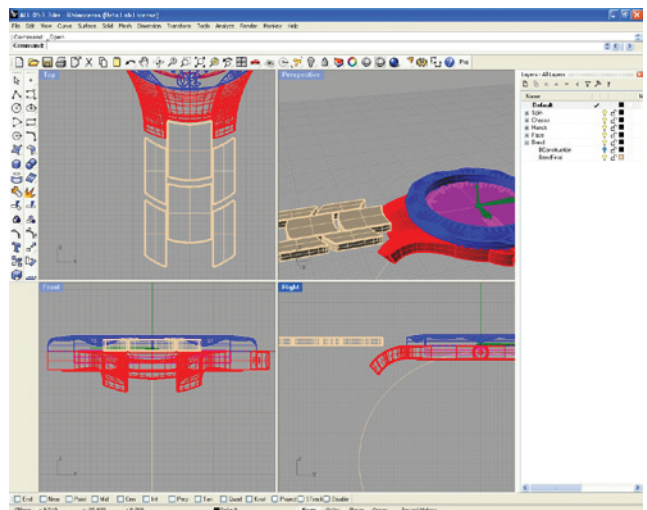
I extrude the date numbers.



I add the watch hands.

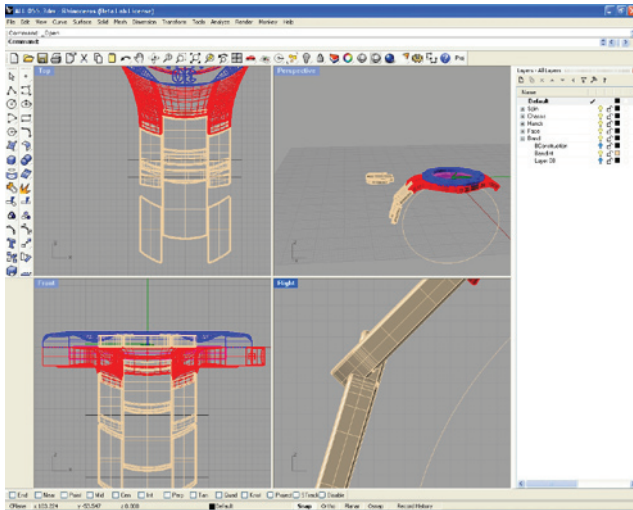


I make the projected shape of the band.

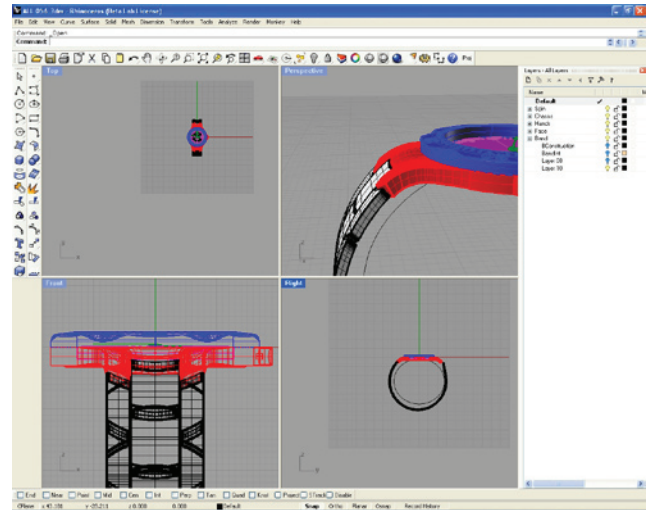


I add a gap between each band link.

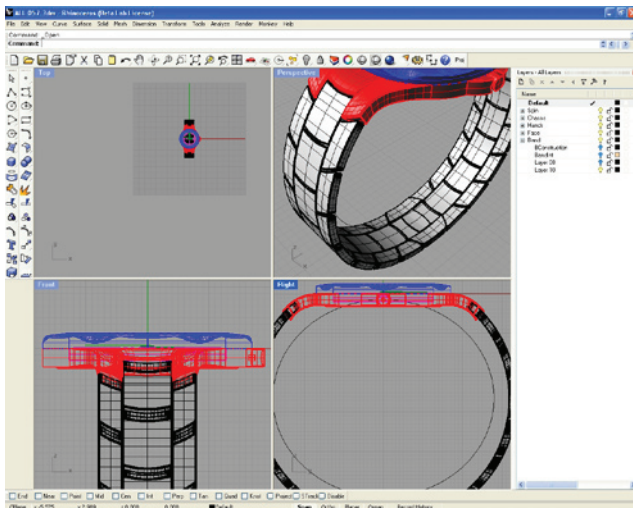




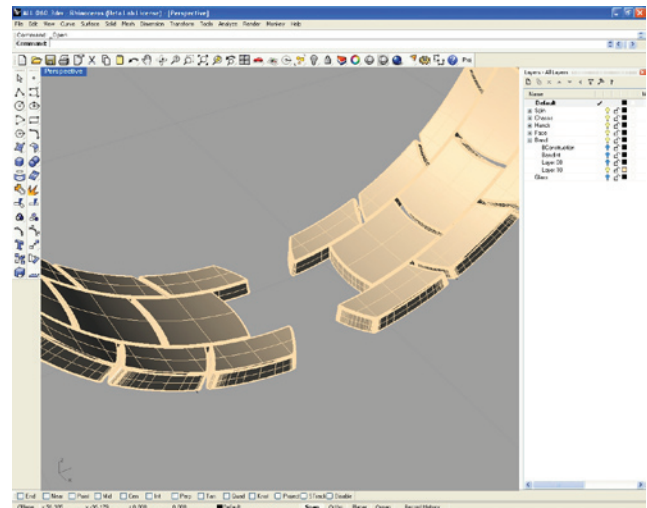
I tried to rotate each link individually to get the band to curve around. However this makes the ends stick out, which makes it less aesthetically pleasing, and less safe to use. This is not what I wanted, as mentioned in my statement of intent.



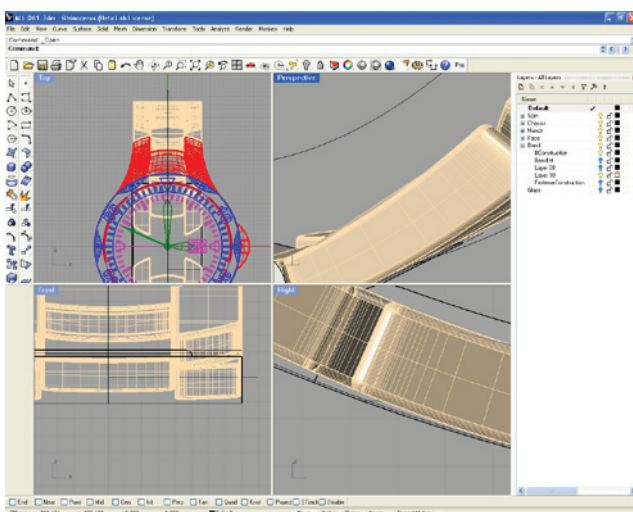
So I try bending the projected band around. This however caused problems with some of the related geometry, which I only uncover later.



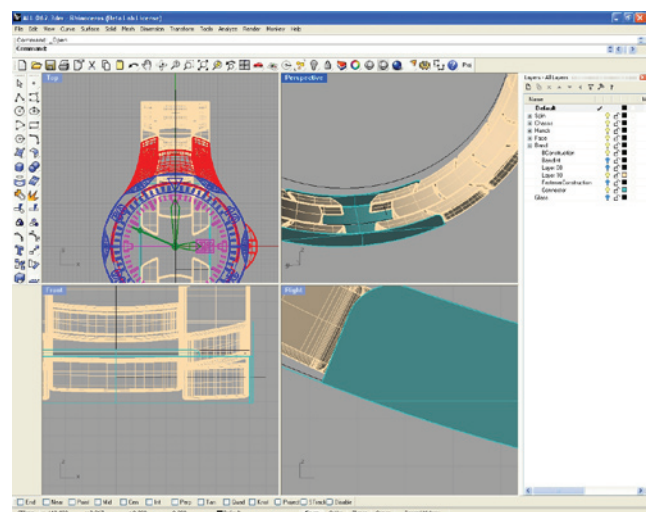
I take one half of the bent band and mirror it to create the other side symmetrically.



I remove the extra links left from before.



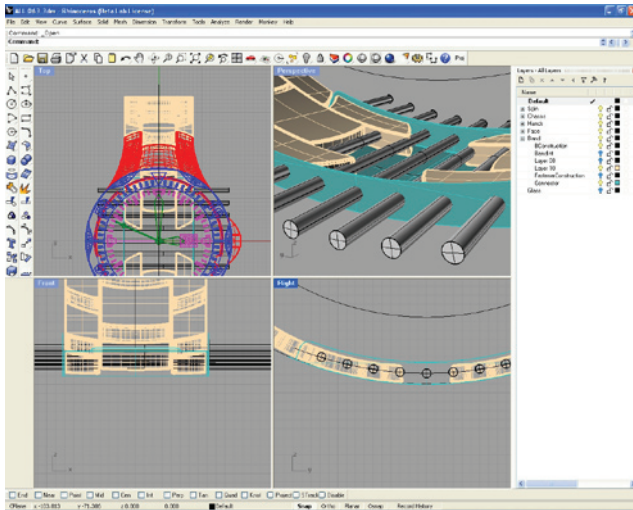
I make the curves for the clip shell.



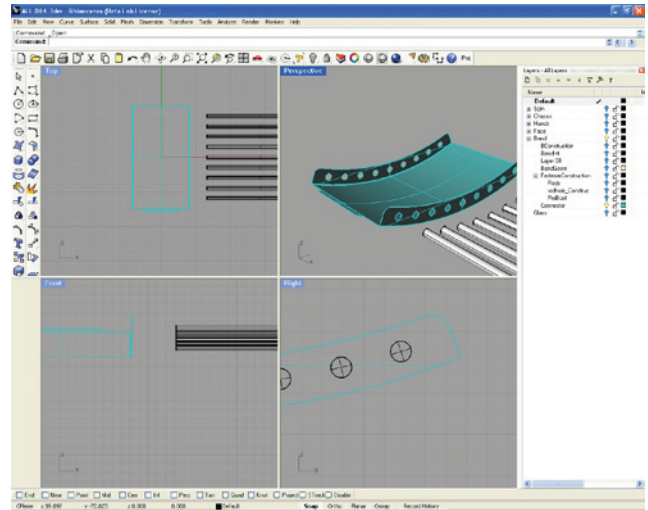
I make the clip shell surface. I will later extrude all the surfaces in the clip to make them have depth.



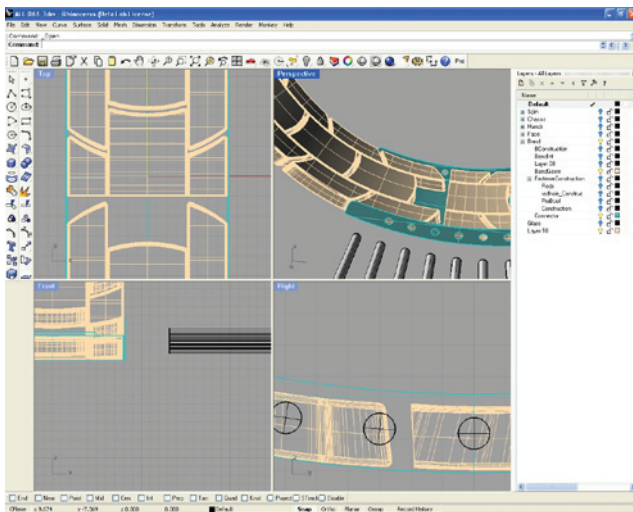
- 3. DEVELOPMENT OF IDEAS -



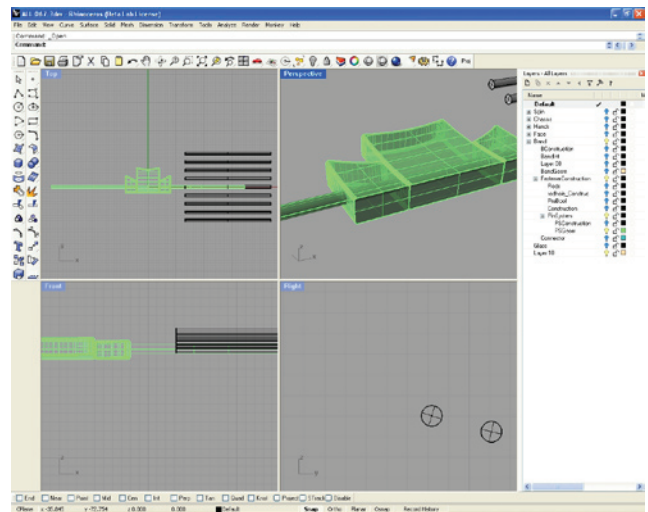
I add long cylinders to cut out holes for the pins.



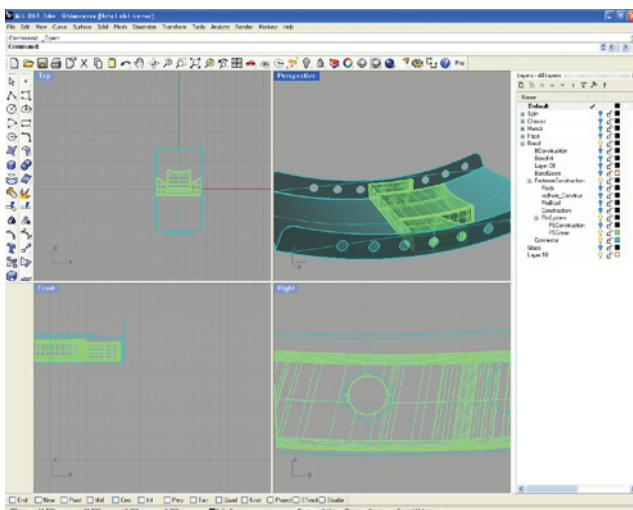
I use the cylinders to cut hole in the clip shell.



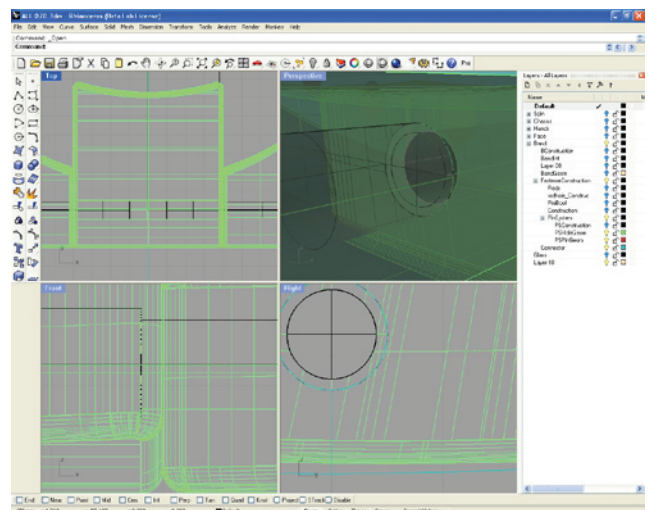
I cut part of the band off to allow the pin in it to be created.



I make a hole in the last band links to allow for a pin.

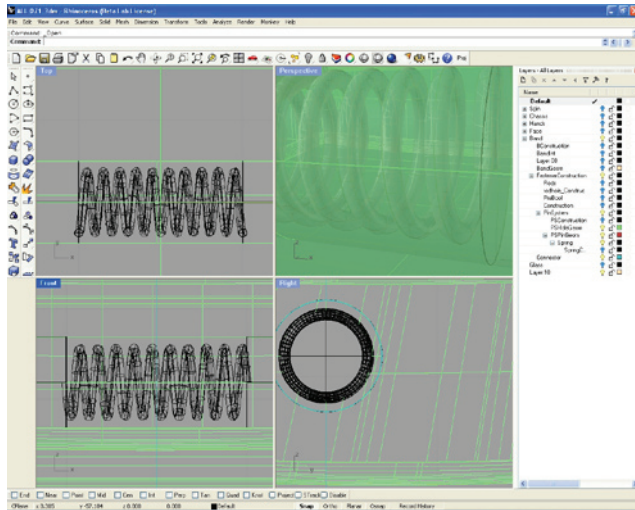


The hole is cut.

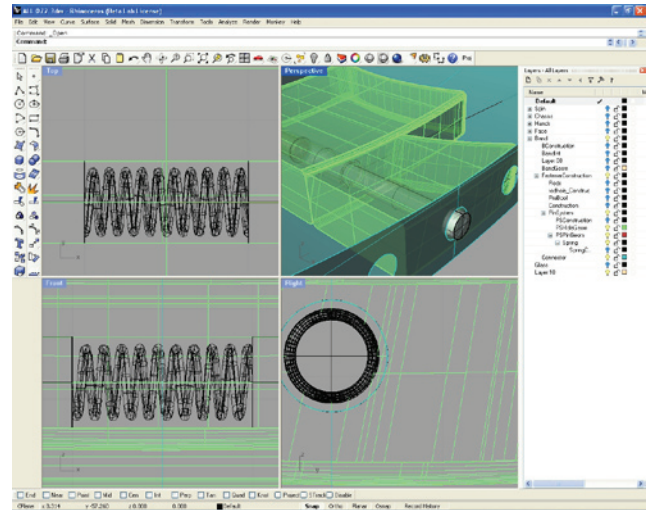


I refine the hole to make it mechanically work.

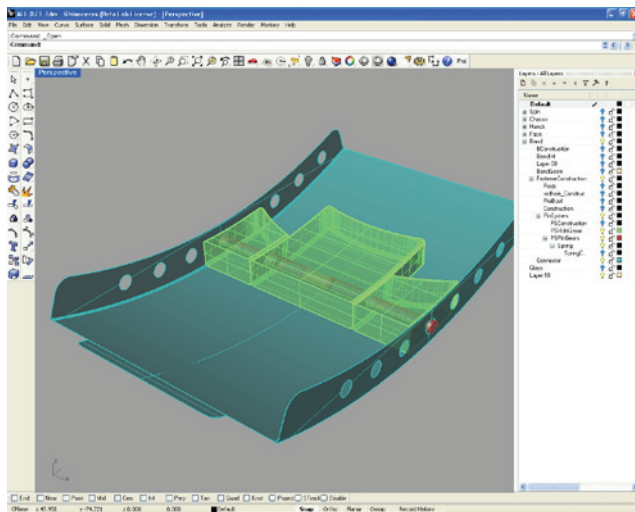




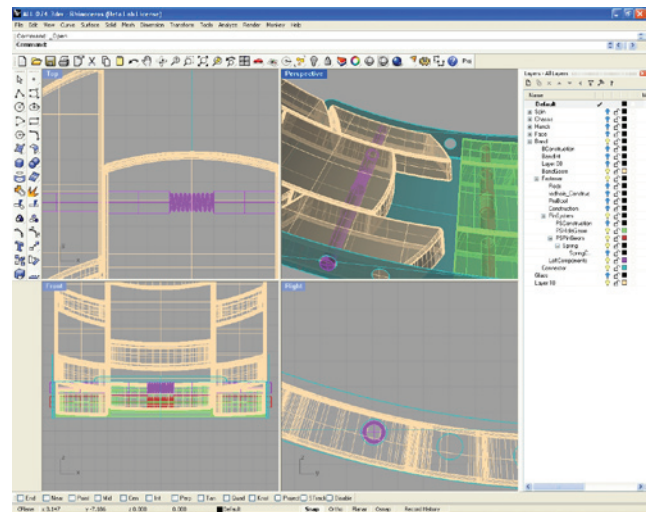
I make the spring.



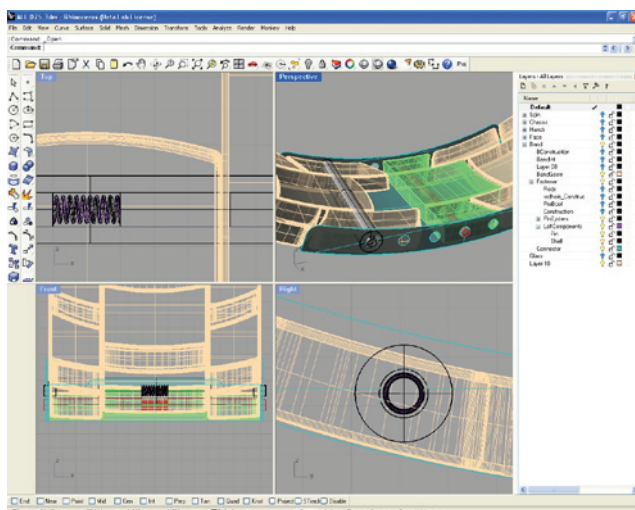
I make the pin which fits in the hole.



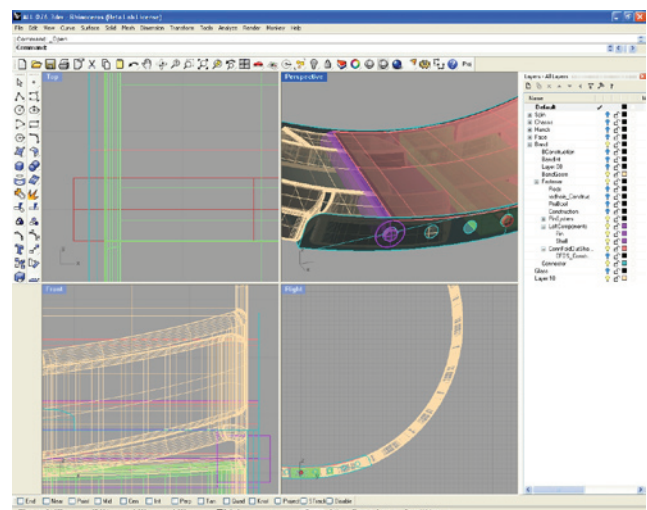
The pin and band and clip shell.



I made another spring for the other side of the band.

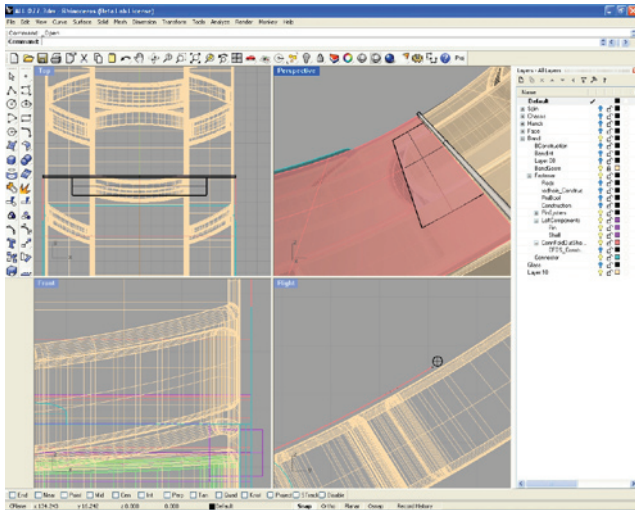


I add the casing for the spring.

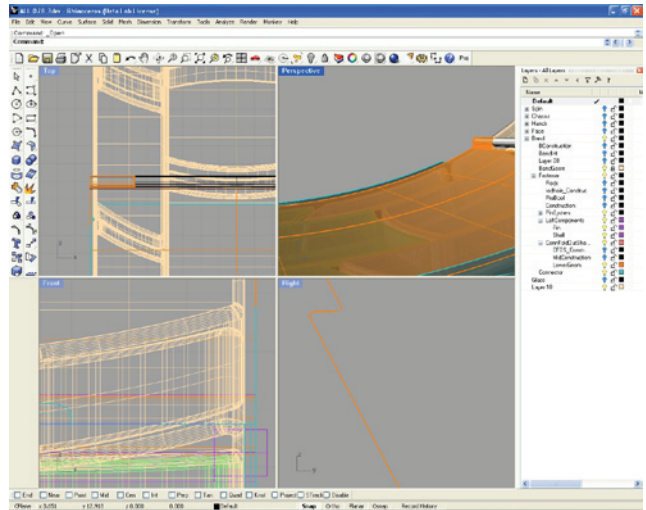


I added the lower sheet.

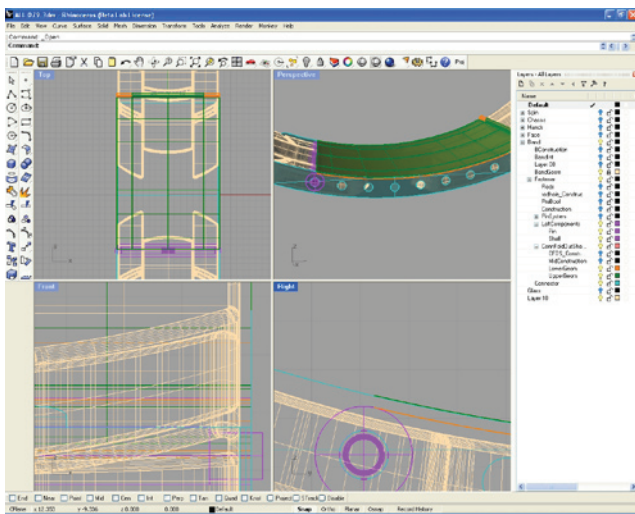
- 3. DEVELOPMENT OF IDEAS -



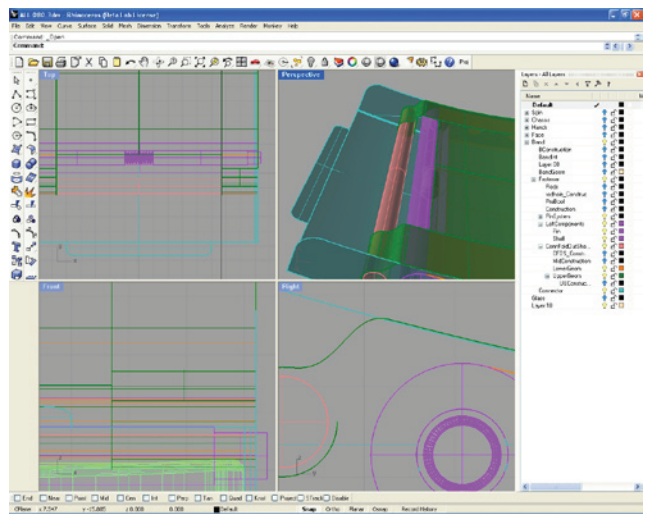
I worked on the connection of the two sheets that fold open to extend the length of the band.



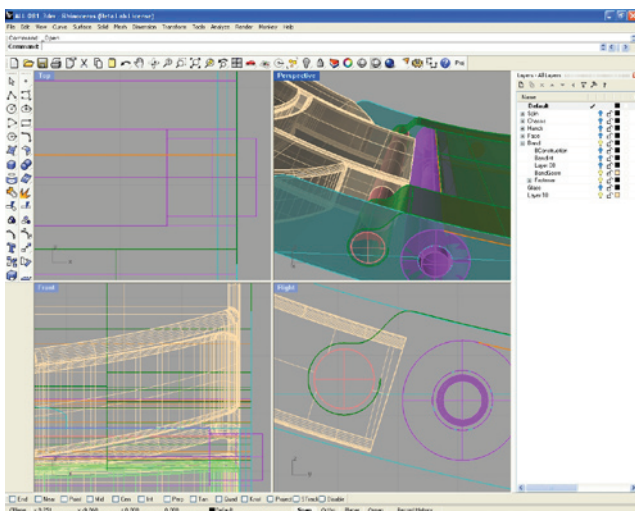
I continued to work on the connection between the two sheets.



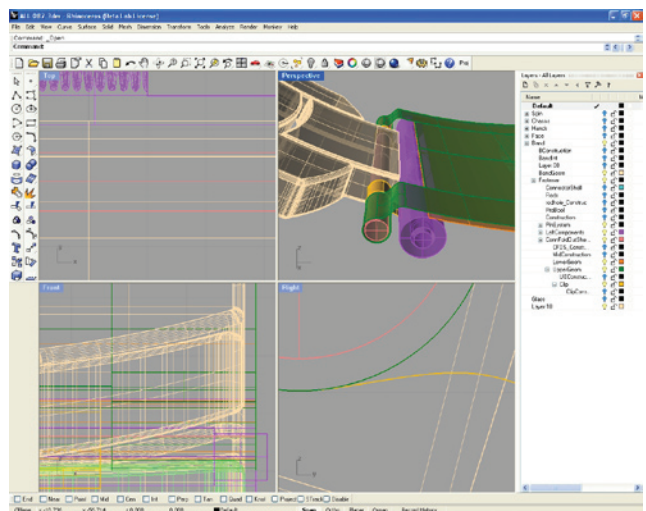
I added the upper sheet.



I worked on the 'clip' end of the clip, connecting the upper sheet to a rod.

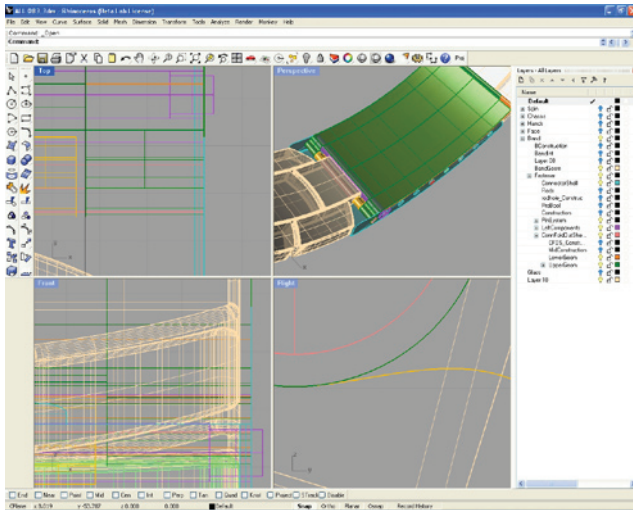


I refined the end of the band to fit onto the rod.

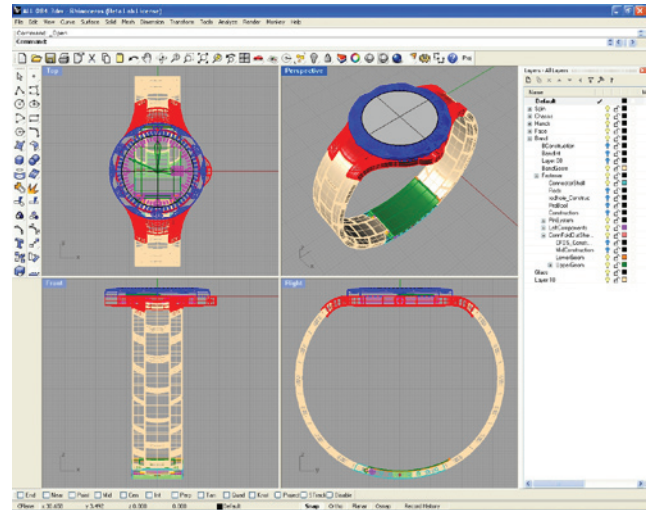


I added two flaps which hold the sheets together, but when a force is applied it will open.

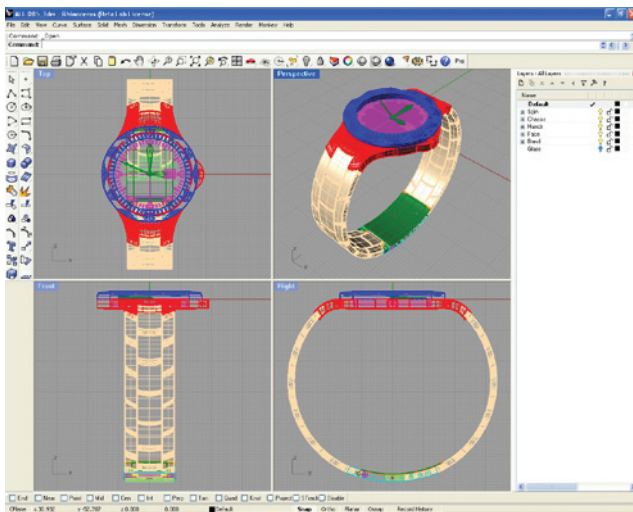




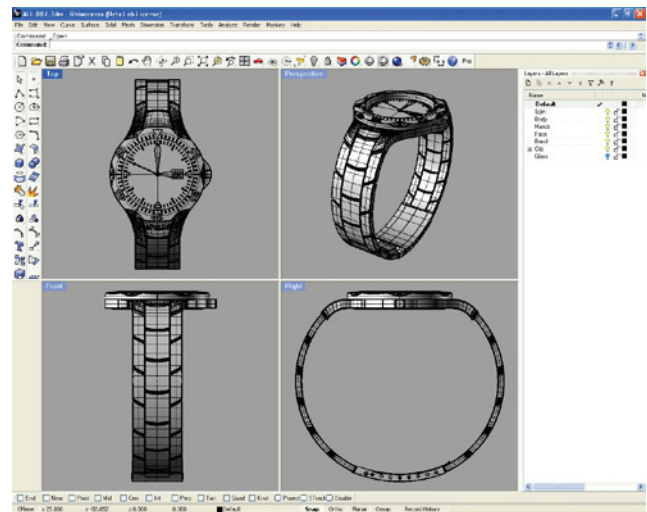
All together now.



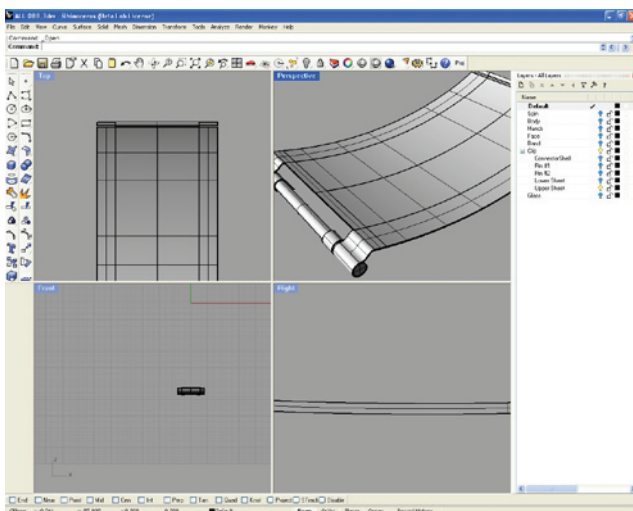
I added the glass on the top.



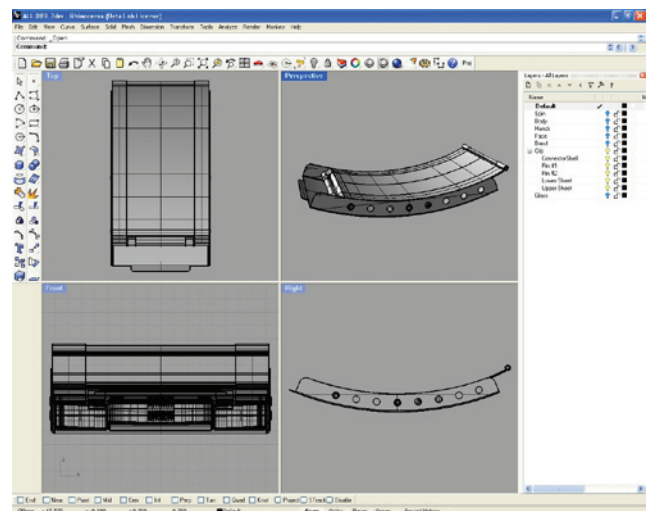
Again, all put together.



I cleaned up the layers, deleting any objects that were not part of the final model. These redundant items had been kept as sometimes I needed them to redo or make new objects



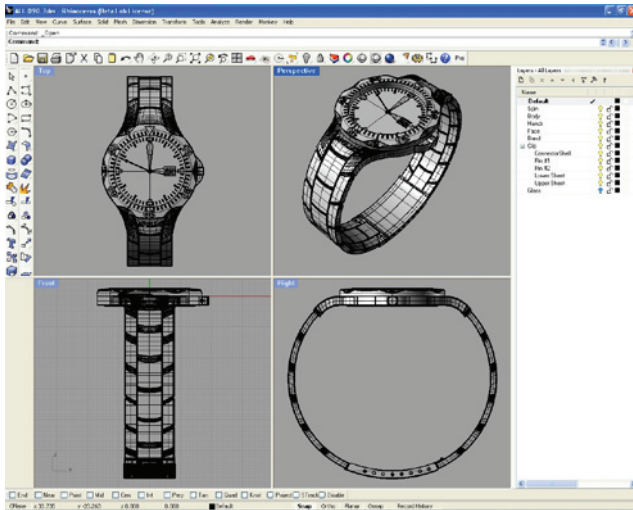
I thought that I had finished at the last step, but most of the clip was made of surfaces with no thickness. So I extruded the surfaces giving them thickness. Determining the thickness of these sheets was difficult.



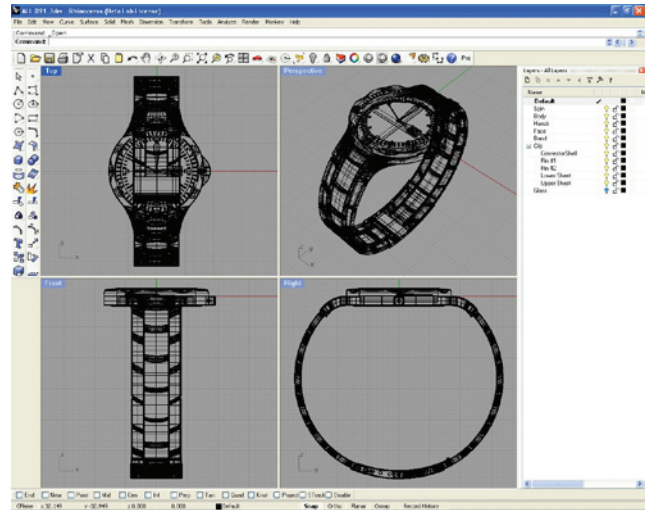
Clip complete, with thickness.



- 3. DEVELOPMENT OF IDEAS -



Final watch, cleaned and with any redundant layers or objects removed. The DD logo and other writing will be put on the watch face during the rendering process, as these will be manufactured by inkjet printing them onto the watch face surface. Therefore they do not contribute to the 3D geometry.



...the final watch design is completed.

From this model I made renderings, 2D drawings and other production items.

DESIGN PROBLEMS & DESIGN EVALUATION

CLIP

Once I had designed the watch body, face, hands and band. Apart from the inside mechanics (which I was not planning to design) and the materials from which the parts were to be made from, all that was left to design was the fastening device, aka the clip. As mentioned in my statement of intent I needed a design solution that accounted for a variable wrist size, so the length of the band needed to be able to be changed.

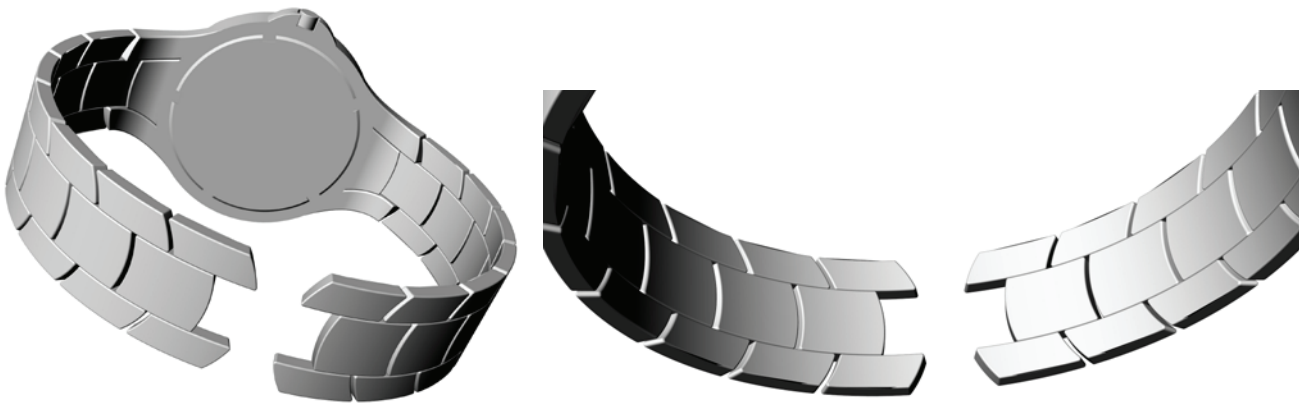


Figure 3.9 - Band without fastener

Through my research I decided that the best design was an existing clip design. In which two pieces of sheet metal would fold and sit on top of each other when fastened and would unfold and expand out in order to lengthen the band to allow the user to take the watch off. This is shown in figure 3.9.



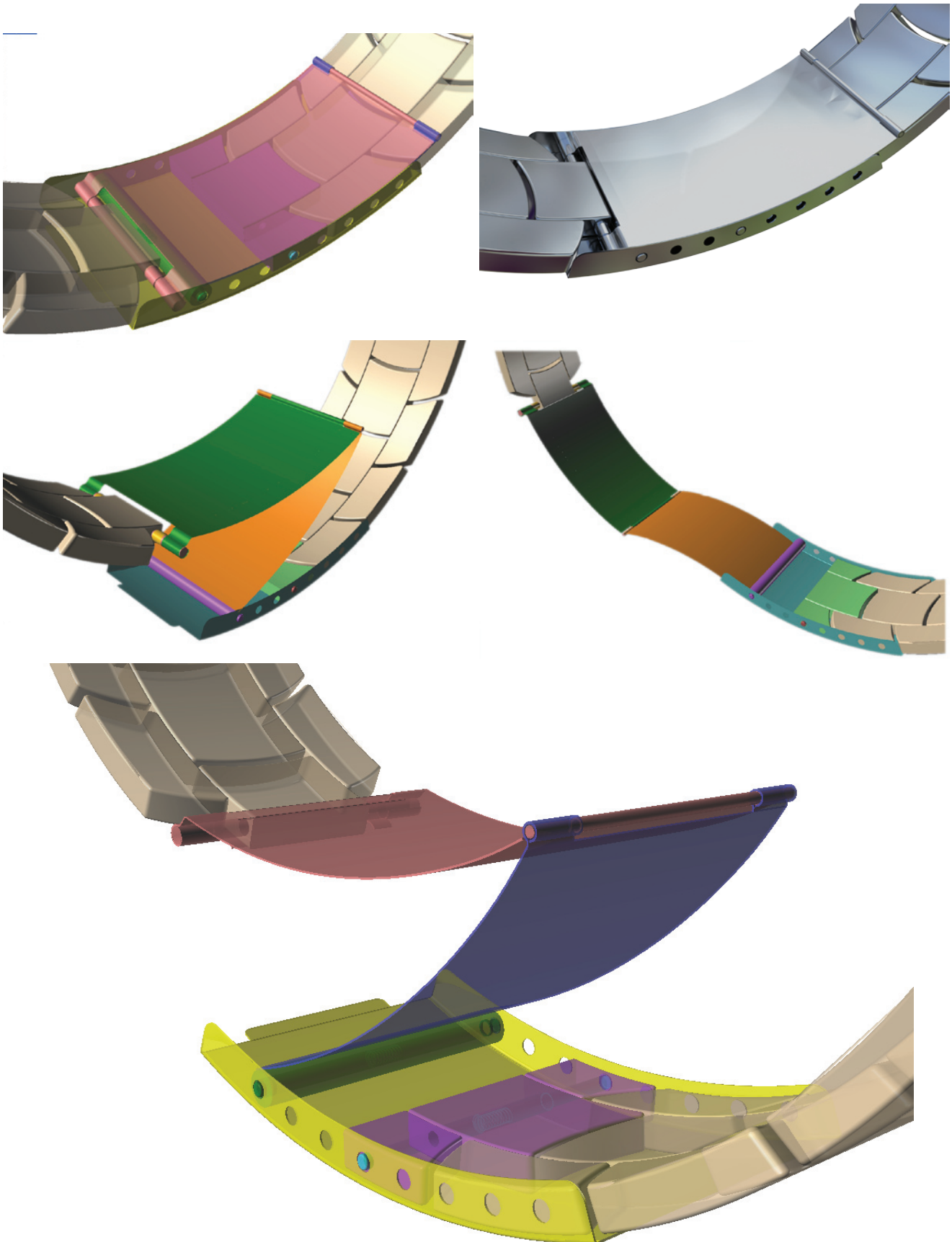


Figure 3.10 - Final Fastener Design

To make the size of the band changeable, the two pins that hold each side of the band to the clip can be held in different holes that are in the clip. To allow the user to change which hole the pin is in, a spring was inserted in the pin to allow the user to make the pin smaller and thus able to change



- 3. DEVELOPMENT OF IDEAS -

holes. A rendered section of the pin and spring is shown in figure 3.11 and a ghosted rendering is shown in figure 3.12.

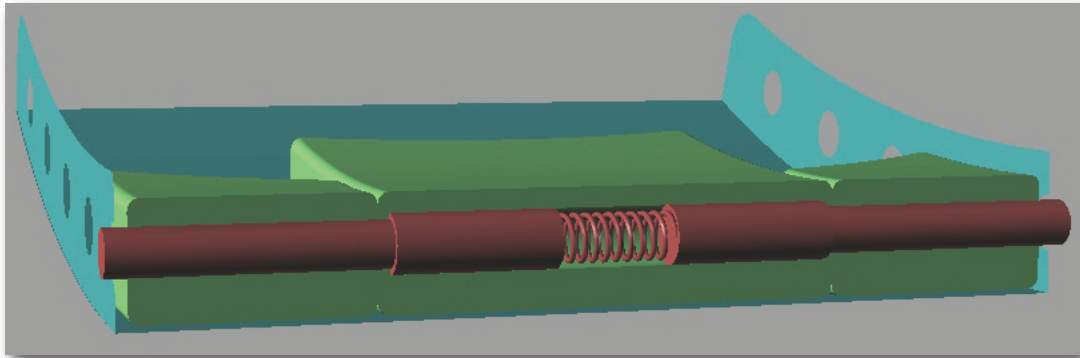


Figure 3.11 - Rendered Section showing pin and spring

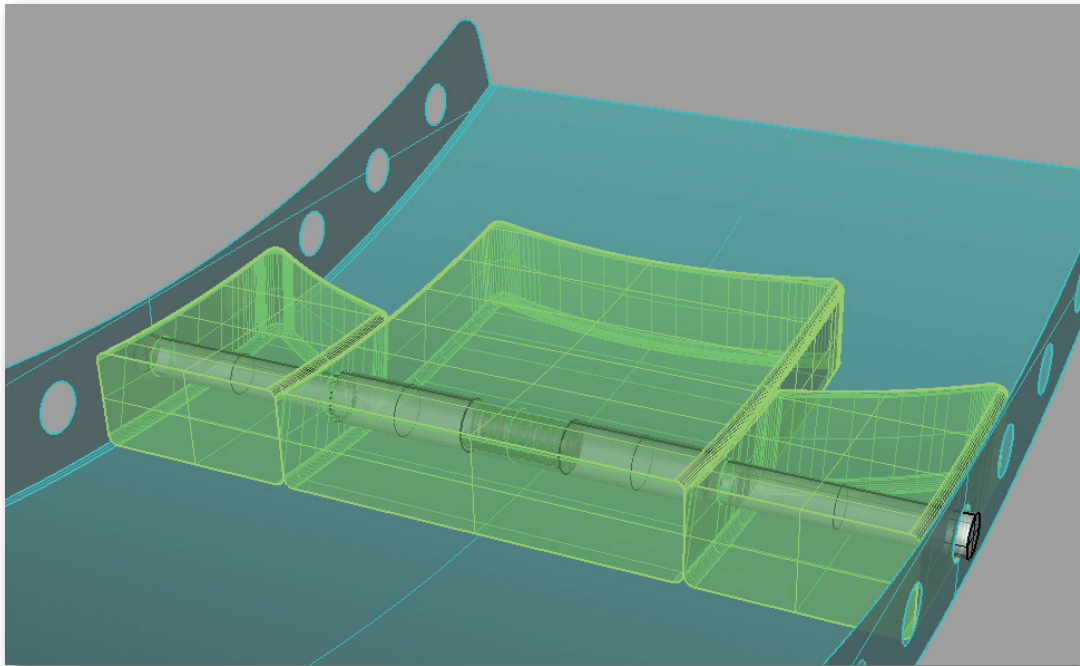


Figure 3.12 - Ghosted Rendering showing pin and spring

The clip turned out to be the most difficult part to design. This was because of several reasons including that it is a lot harder to research existing design solutions; the mechanism has to be adjustable for different size wrists; the parts don't simply fall together, they have to fold and move into place; and it must collapse together to hide all the inner workings, to be aesthetically pleasing as outlined in my statement of intent.

BAND

This was a huge design problem. However that said, it was more of a design/CAD incorporation problem. I had sketched my final design of the band and I knew what it should look like, however due to the complexity of the band I found it very difficult to recreate this design on the computer accurately. In the end I achieved a static model and related CAD drawings which along with the 3D CAD model allow for the production of the band. However I was not able to simulate how the band would behave without building a working prototype, which I did not do.

Due to my inability to simulate the band, when I opened the clip in the 3D modelling software, the band did not adjust and the best I could do is shown in figure 3.13. This is not how the manufactured watch would behave.





Figure 3.13 - Problem due to inability to simulate band dynamics

Rhinoceros is not able to simulate the band in the way I needed. 3ds Max is however able to do the simulation. I could use inverse kinematics, the same tools that are used in 3ds Max for animating characters. However as it is difficult to get accuracy in 3ds Max and as it could not transfer the full amount of data I needed about my model from Rhino to 3ds Max, and also due to time constraints, I was never able to run a simulation on the dynamics of the band. See also General Project Evaluation → Ideas That Never Made Reality → Dynamic Simulation of Band, page 90.

ERGONOMICS

This was not so much of a problem, however I could have been much more thorough when considering ergonomics during the design stage. Because through my research on the internet and various libraries I could not find an anatomical chart, I had to do a survey. I used a small sample size from a range of ages, and ethnic backgrounds. I measured the basic shape of their wrists and I came up with an average width of 55mm and an average height of 38mm. I could have used a larger and wider sample size or an anatomical chart to get more accurate results. I did the survey so that my watch would fit the majority of people, as outlined in my statement of intent.

However from the data I collected I only knew a width and a height. I had no data about the shape. This didn't matter so much because due to the dynamics of the band it would adjust to a different size. If I had used a 3D laser scanner to gather more detailed information about the sizes and shapes of peoples wrists, then this would give me a 3D model of a wrist. I could do this with a wide range of people, and collect the data. However processing and interpreting the data would prove difficult for me, and due to my inexperience in this area I would not need that detailed information, so I did not use a 3D laser scanner.

MATERIALS

Once I had finished the watch design in Rhino, I exported the model into 3ds Max where I added materials and did photorealistic renderings to see what my watch design looked like in different materials and environments.



- 3. DEVELOPMENT OF IDEAS -

Many of my test renderings can be found here. I used them to experiment with different real life materials. The quality of the render may be low as I lowered the quality to get the renderings done faster, however this did not affect how the material looked very much. Also some of the renderings were done before I had finished the final watch design, so certain parts may not be shown.



Figure 3.14 - No materials applied



Figure 3.15 - Wood material





Figure 3.16 - Not all renderings turn out as intended

As shown in figure 3.17 (below), I first considered a wooden background (or at least one that looks like wood). I think that this would make the watch more aesthetically pleasing, however in my final design I choose the plastic background. Also the band in this test rendering is plastic, I did this render to see what the watch would look like in plastic.



Figure 3.17 - Wooden Background and Plastic band





Figure 3.18 - Another plastic and wood test render



Figure 3.19 - I explored some polished wood



Figure 3.20 - Another wood





Figure 3.21 - Toon rendered





Figure 3.22 - Test rendering with stainless steel material



Figure 3.23 - Rendering with gold plated face markings and hands and logo



The black background shown in figure 3.24 (below) looks good, but made the time harder to read, especially in limited light environments.



Figure 3.24 - Rendering exploring a black background.

The final materials are shown in figure 3.25 (below), the word APOLLO was removed as it made the face too cluttered. I think that the watch would have looked better if I added some more small text to the face such as brand name and model, water resistant, etc.



Figure 3.25 - Final rendering showing final materials

Because metals are highly reflective, the watch can look vastly different in a different environment in different locations. The way the watch would look in day light would be different to how it would look in a disco room with coloured lights everywhere. For presentation purposes the watch was shown in a natural or artificially lit room. In the end the watch should be presented in a way that makes it look most appealing. Figure 3.26 (next page) shows the same watch with the same final materials however in a different environment with different lighting.





Figure 3.26 - Same materials in a different environment

OVERALL SIZE

I made my watch in Rhino, with all the parts sizes relative to the face diameter. I only measured main parts of the watch such as height, depth and face diameter and evaluated these. It seemed right and it would fit on different size hands. However as things look very different and of different sizes on the computer screen, I did not grasp the real life look of my watch until it was rapid prototyped (see Production Item No. 102, also see Evaluation of this item on page 109). Although I had got a partial idea from printing an orthogonal drawing at a scale of 1:1. Because none of my drawings were of a scale of 1:1 I did not see the actual size, until I noted the dimensions generated which seemed quite small. With this in mind I did a rendering of my watch with a real life object placed near it. This helped me to see the real life scale of the watch.

GENERAL DESIGN EVALUATION

This partly addresses appropriateness of design and/or design modification.

In evaluation of my final watch design, which is shown in figures 3.27 - 3.30, I feel that I have achieved the design that I set out to design in my Statement of Intent. I feel that I have chosen effective and appropriate materials that make the watch look top class. I feel that I have designed the operation of the watch in terms of changing the time, using the timer device, taking the watch on/off and changing the length of the band in an easy to use and understand way. I believe that my design is ethically suitable and is culturally appropriate. And by making the watch out of stainless steel, it will be durable and will last a long time before breaking. Also the smooth edges also made it safe for people to wear and use. So I believe that my final design has satisfied my intended solutions and that factors such as appropriateness of the design solution, function, aesthetics, ergonomics, use of the design, safety and durability have been addressed and meet.





Figure 3.27 - Rendered Final Design with correct materials, Top Perspective View



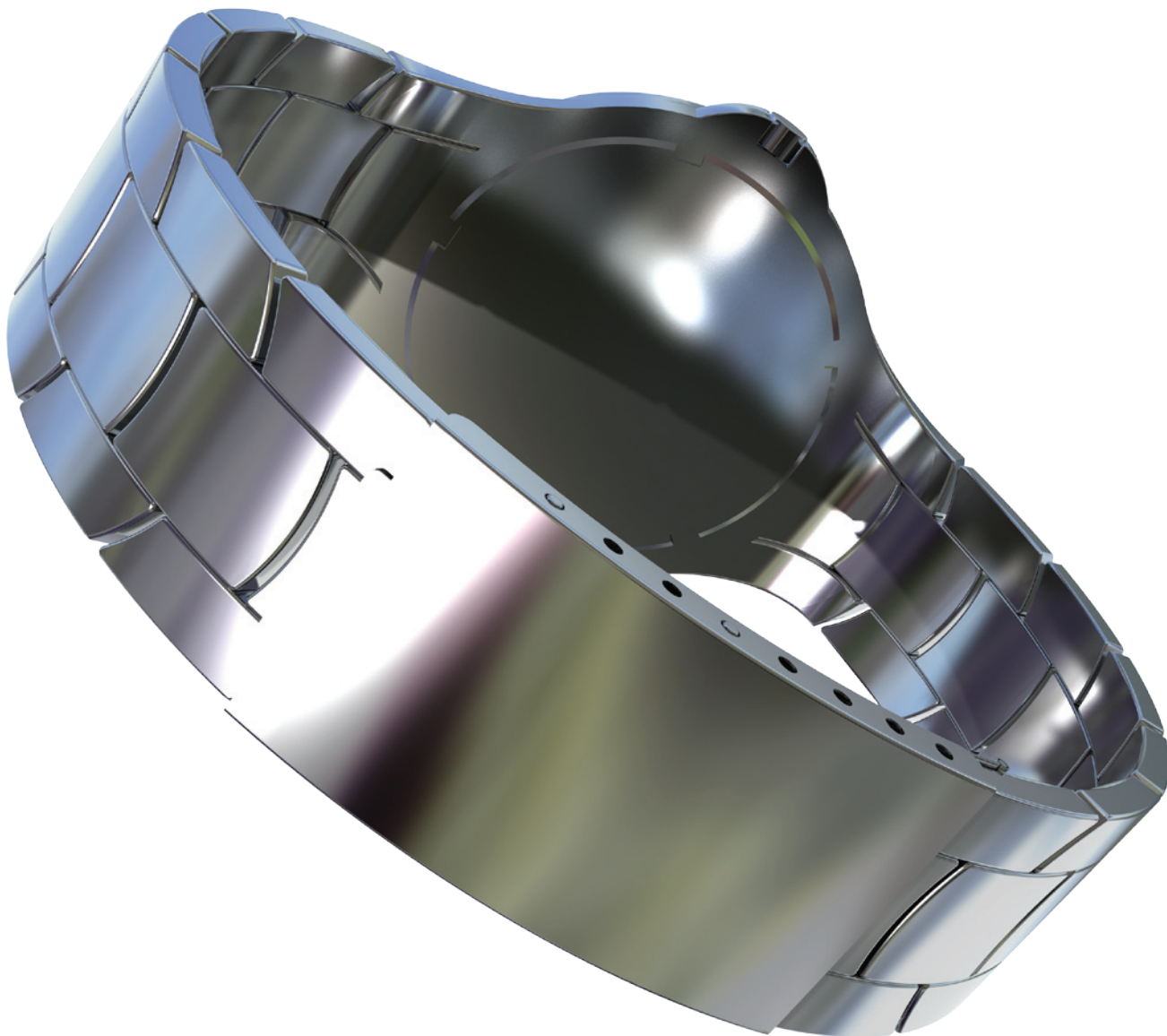


Figure 3.28 - Rendered Final Design with correct materials





Figure 3.29 - Rendered Final Design with correct materials





Figure 3.30 - Rendered Final Design with correct materials



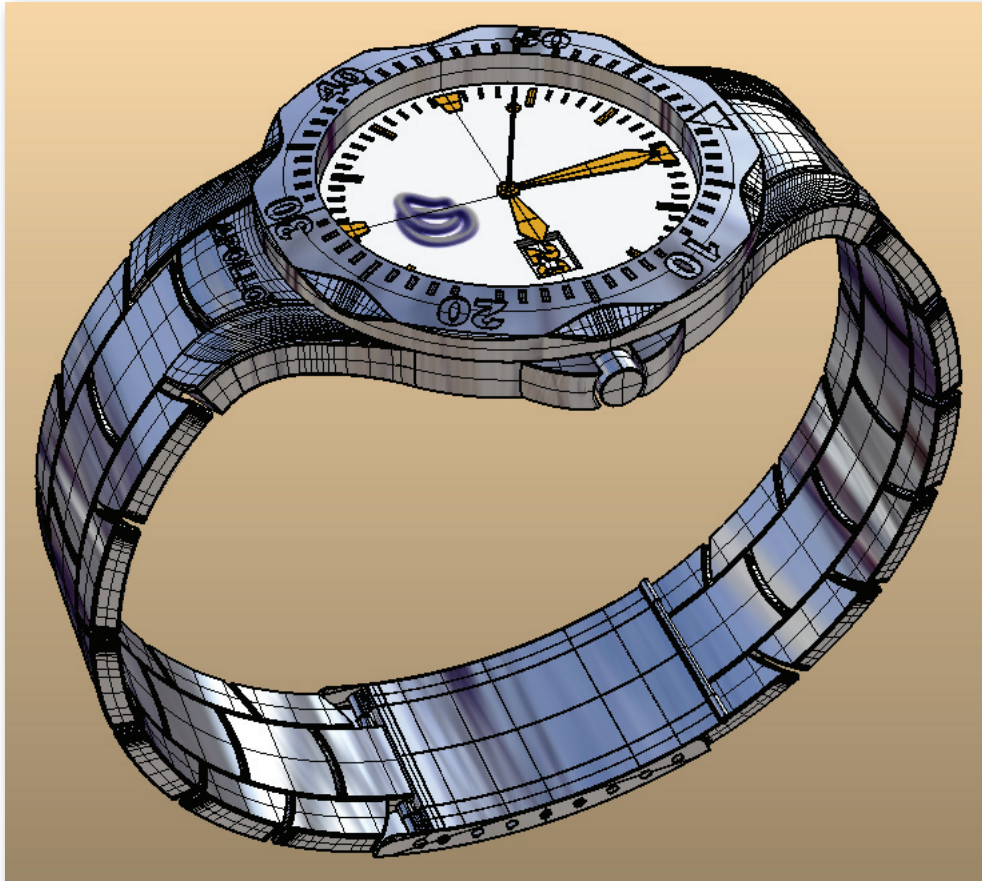


Figure 3.31 - Final Design in Rhino with approximate materials applied

RELATIONSHIP BETWEEN DESIGN AND PRODUCTION

This section fulfils the assessment criteria "Assesses the relationship between the design, and modifications if applicable, materials, components and processes in the development of the major project"

The design of both the watch and the drawings has influenced the materials, components and processes that I have used in the development of my major project. Foremost the design of the thing that I am drawing (watch) has meant that my production of drawings can not be done using conventional¹ (hand drawn, including on CAD and paper) methods due to the sheer complexity of assemblies and complexity in terms of the curves and shapes used. This has meant that I cannot use traditional manual processes, I needed to use automatic computerised processes where drawings are produced from 3D model data.

The materials (or due to the nature of my industry option chosen, computer software) I needed are much different and need to be able to do much different tasks than the materials that would be needed if I were drawing a simple stepped block.

¹ "Conventional" refers to drawing lines or other 2D geometry on computer. These lines are drawn manually for all drawings. This is contrasted to "automatic computerised processes" where a 3D model is designed on computer and all other drawings are generated from that 3D model. My production pipeline is a mix between these two methods however automatic computerised processes was much more predominantly used than conventional methods.



4

SELECTION AND JUSTIFICATION

BRIEF SUMMARY

Material/Resource/Process	Researched Options	Primary Selected Option	Justification of Selection
3D Rendering Software	Discreet 3ds max 6 Autodesk 3ds Max 8 Autodesk 3ds Max 9 Autodesk Maya 8 Rhinoceros Autodesk AutoCAD Cinema 4D Softimage XSI trueSpace VRay mental ray Pixal PR Renderman	Autodesk 3ds Max 9	Free Trial Available Industry Standard Produces High Quality Work Easy to Use and Learn
Page Layout/Publication Software	Microsoft Word 2003 Microsoft Word 2007 Microsoft Publisher 2007 Adobe InDesign CS2 Adobe Framemaker 7 QuarkXPress 8	Adobe InDesign CS2	Industry Standard Simple to Use Easy to Learn
Rapid Prototyping Method	3D Printing Stereolithography CNC Milling Cutting Machines	3D Printing	Cost Effective Easily Available via Internet Speed of Production Produced Locally
2D CAD Software	Autodesk AutoCAD IMSI TurboCAD Autodesk AutoSketch	Autodesk AutoCAD	Readily Available at School Already Competent in Use Industry Standard Most Features
3D CAD Software	Rhinoceros 4.0 Rhinoceros 2 Autodesk Inventor CATIA Solidworks Pro/ENGINEER UGS NX	Rhinoceros 4.0 BETA	Freely Available from School Easy to Use Most Features

MATERIALS

SKETCHING AND DESIGNING

My preliminary designs were sketched by manual methods. However I used drawing instruments to aid me in the drawing process.

STAEDTLER 3H, HB, 3B & 4B PENCIL

The 3H pencil was useful for doing initial sketching as it could be erased easily. The HB pencil was useful for darkening lines to show the main lines in contrast to construction lines. The 3B and 4B pencil was useful for adding shadows, and darkening lines to give a 3D effect. The 3H, HB, 3B & 4B pencils proved very useful and worked as I had planned them to.



PAPERMATE PACER (0.5MM LEADS)



The pacer was used to draw thin lines and small detail, as it never goes blunt.



STAEDTLER ERASER

The eraser was used so that I could erase unnecessary lines.

KENT 30-60-90 SET SQUARE

The set square was used for drawing straight lines, and also to draw lines parallel to the edge of the page.

FRENCH CURVES

The french curves were used to draw smooth curves easily and quickly.

REFLEX WHITE 80GSM A4 PAPER

A4 paper was used because it is just the right size to draw on, and can also fit into my A4 folder easily.

PRODUCTION OF PROJECT

REFLEX A4 INKJET GLOSS 150GSM

This paper was used because it was higher quality than normal paper and make the presentation look better.

REFLEX WHITE 80GSM A4/A3 PAPER

A4 paper was used because it is easily accessible to print on, and can also fit into my A4 folder easily. A3 paper was used because it provided a larger area to draw on.

SAMSUNG BLACK AND COLOUR TONER INK

I used this ink as it was designed to go into the printer. I used colour to make it look better and communicate more information clearly.

MANAGEMENT FOLIO

REFLEX WHITE 80GSM A4 PAPER

A4 paper was used because it was easy to print to, and fit into my plastic sleeve folder.

COMPONENTS

SKETCHING AND DESIGNING

FREEHAND SKETCHES

I produced freehand sketches as part of my development of ideas because they help to show *evidence of a range of skills* and they allowed me to quickly and easily get my ideas down on paper so that I could evaluate the design and show the process of design modification that I made throughout the project. Freehand sketches were also used because I could do the sketch anywhere and with minimal materials and resources. This meant that when I thought of an idea, wherever I was I could draw it up then and there. Ultimately these sketches were my primary resource in creating the 3D model, which was in turn my primary resource in creating the other production components. I used annotations in my sketches because annotations conveyed my ideas more clearly.

PRODUCTION OF PROJECT



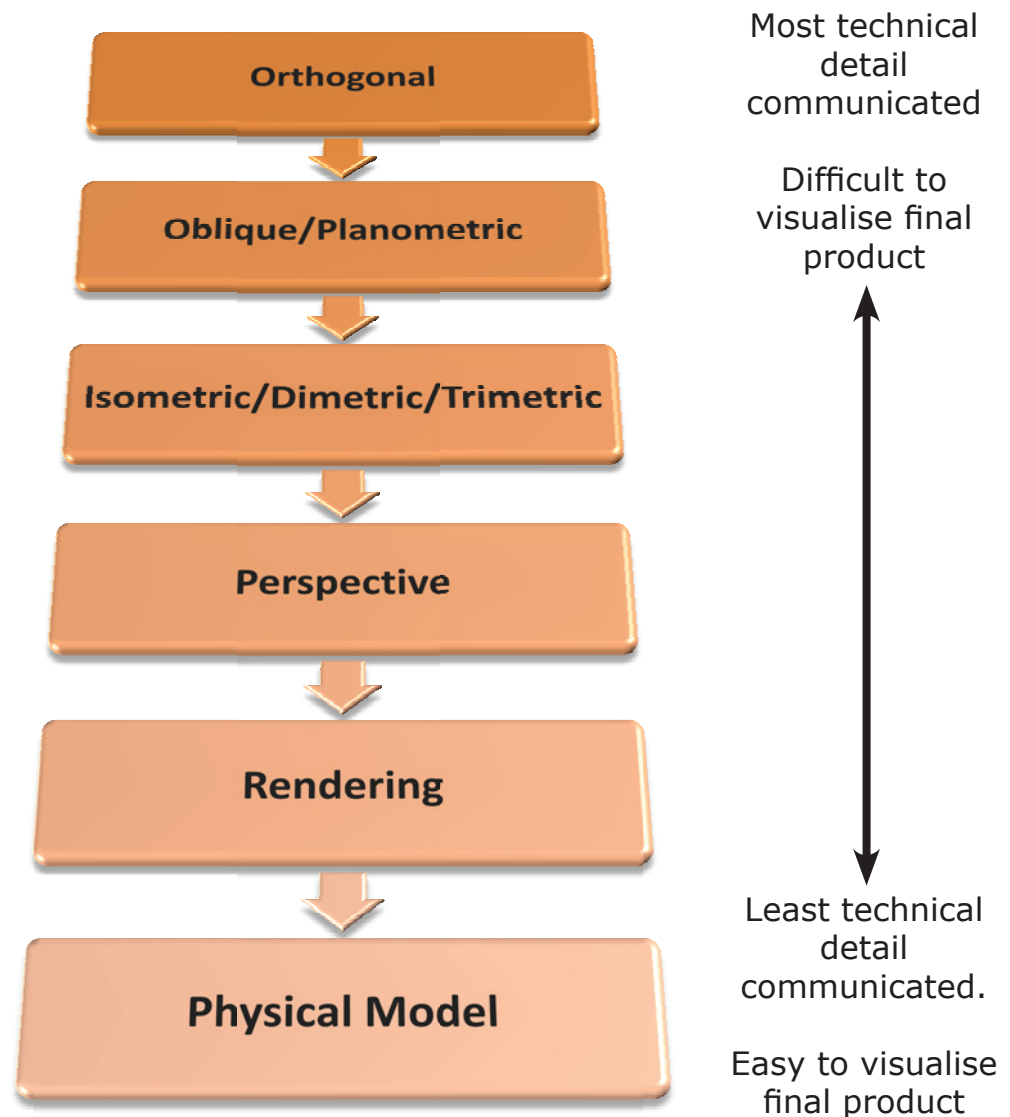


Figure 4.1 - Technicality Scale

3D MODEL

(See also, Selection and Justification → Processes → Production of Project → Workflow, page 55.)

ISOMETRIC DRAWINGS

I produced isometric drawings as part of my production because they help to show *evidence of a range of skills*, and they allowed me to accomplish what I outlined in the *statement of intent*, in respect to providing as much information and detail to manufacture the product. In addition I used isometric drawings because they show the product in an easy to visualise way, but also as there is no perspective distortion and measurements are to scale. Isometric drawings were used mainly for drawings of individual parts as the focus was technicality not photo realistic.

(See also, Solutions to Problems in Production → Graphics Problems → Isometric Projection vs. Isometric Drawing, page 132.)

OBLIQUE DRAWINGS

I produced oblique drawings as part of my production because they help to show *evidence of a range of skills*. I mainly produced cabinet oblique rather than cavalier or general as cabinet looks more realistic.

(See also, Problems in Production → Graphics Problems → Oblique Circles, page 131.)



(See also, Appendix A - Developed Software → Oblique Projection Tool, page 138.)

PERSPECTIVE DRAWINGS

Creating a perspective drawing was very easy, as this process could be automated, made possible by having a 3D model of my wristwatch. Rhinoceros has built in algorithms for perspective projection, as with almost all 3D products, but Rhinoceros could take that perspective projection, and create vector data from it, instead of only displaying in on-screen as a bitmap (in the perspective viewport). Rhinoceros also took into consideration hidden detail and tangent edges. In the end I used Rhinoceros to do my perspective drawings for two reasons. Firstly because it would be far too hard to make a three point perspective drawing to the detail in my design, by manual methods (mechanical and measuring point). And secondly because Rhinoceros was set up to do it, it could do it well and my model was made in it. So why would I, waste time finding another product, if one even exists, and then trying to worry about compatibility issues, or if I cannot find another product then why waste my time and effort writing my own program from freely available algorithms? That is why I used the built in function in Rhinoceros.

I produced perspective drawings as part of my production because, they help to show *evidence of a range of skills*, and they allowed me to accomplish what I outlined in the *statement of intent*, mainly in respect how I intend to provide as much information and detail to market the product to the public, and show how it looks in a simplistic way.

PLANOMETRIC DRAWINGS

Planometric drawings are more commonly found in architectural drawings, however I produced them to show *evidence of a range of skills*. To assist me in the production of planometric drawings I used the software I created solely for this project, ViewIt. (See also, Appendix A - Developed Software → ViewIt, page 141.)

RENDERED DRAWINGS

I produced rendered drawings as part of my production because, they help to show *evidence of a range of skills*, and they allowed me to accomplish what I outlined in the *statement of intent*, mainly in respect to how I intend to provide as much information and detail to market the product. I rendered a range of drawing projections. In my rendered drawings I showed texture and materials that the watch was made of. I was limited to the projection types that I could render. Documentation of this can be found in Evaluation → General Project Evaluation → Ideas That Never Made Reality → Rendering in Less Common Projections, page 90.

ORTHOGONAL DRAWINGS

I produced orthogonal drawings as part of my production because, they help to show *evidence of a range of skills*, and they allowed me to accomplish what I outlined in the *statement of intent*, mainly in respect to how I intend to provide as much information and detail to manufacture the product.

The two main types of orthogonal drawings are 1st angle and 3rd angle. Under AS1100 3rd angle is preferable, so I did my drawings in 3rd angle.

PHYSICAL MODEL

I produced a physical model as part of my production because it helped to show evidence of a range of skills, and it allowed me to accomplish what I outlined in the statement of intent, mainly in respect to how I intend to provide as much information and detail to manufacture and market the product. I chose to produce the specific type of modelling technology of Rapid Prototyping because, it is very accurate with no room for error, it was fast and easy for me to have produced and is an emerging technology in the graphics industry.

EXPLODED DRAWINGS

I produced exploded drawings as part of my production because they help to show *evidence of a range of skills* and they allowed me to accomplish what I set out to do as outlined in the *statement of*



- 4. SELECTION AND JUSTIFICATION -

intent, mainly in respect to how I intend to provide as much information and detail to manufacture the product. This drawing type is especially important for this project as there are many small intricate parts that go into a wristwatch. In order to show how all these parts fit together and what the parts look like I used exploded drawings.

DRAWING TEMPLATE

(See also, Selection and Justification → Processes → Production of Project → Templates, page 55.)

DIMETRIC DRAWINGS

Dimetric drawings are not commonly used, however I produced them to show *evidence of a range of skills*. To assist me in the production of dimetric drawings, because it would be almost impossible to do one of my watch by hand, I used the software I created solely for this project, ViewIt. (See also, Appendix A - Developed Software → ViewIt, page 141.)

TRIMETRIC DRAWINGS

Again trimetric drawings are not commonly used, however I produced them to show *evidence of a range of skills*. To assist me in the production of trimetric drawings, because it would be almost impossible to do one of my watch by hand, I used the software I created solely for this project, ViewIt. (See also, Appendix A - Developed Software → ViewIt, page 141.)

ASSEMBLY DRAWINGS

I produced assembly drawings because they help to show *evidence of a range of skills*, and they show how all the parts fit together and what the product looks like with all the parts in place. This is extremely important for me as there are many small and intricate parts to a wristwatch. Parts are also labelled and a parts list is provided for further details about the parts.

SECTIONED DRAWINGS

I produced section drawings because they help to show *evidence of a range of skills*, and they show internal detail. This was extremely important for a watch because there are lots of parts hidden away on the inside, and these are shown by a sectioned drawing.

STRESS ANALYSIS DRAWINGS & DATA

I planned on producing stress analysis drawings & data as part of my production because they allowed me to provide as much information and detail to manufacture the product, and which materials and what size pins, etc. should be used. They would also helped me in the design process, to get an engineering point of view in the design, rather than just a designers. However I did not have access to the software that could do the engineering analysis for me.

STEREO IMAGES

I produced stereo images as part of my production as they allow the product to be visualised in real 3D, although not as good as a rapid prototype or other physical model. I only used anaglyph stereoscopy because it was cheap and simple. Other methods required expensive specialised hardware (eg. polarizing glasses and monitor), although it did not look as good than if it were done with other methods.

360 DEGREE TURNABLE ANIMATION

I produced these animations to show the watch at different levels of rendering in a dynamic motion. This animation was much more compelling and more likely to grab a clients or consumers attention than a still image.

MANAGEMENT FOLIO

This is included in the Introduction on page viii.



PROCESSES

SKETCHING AND DESIGNING

ONE ASPECT AT A TIME

When designing my watch I designed each aspect at a time. I usually only moved on to the next aspect once I had completed the previous one. I did this because I found it too hard and time consuming to draw the perfect watch as a whole just from looking at research images. So I designed it one aspect at a time. Then I could take my sketches from all my drawings of each aspect and combine them together to complete the watch design. The aspects of my watch design and details about them can be found in Development of Ideas, page 22.

ANNOTATIONS

Annotations are the words and notes and labelling of my freehand design development sketches. These annotations played a key part in my communication of ideas through these sketches. Although the sketches alone provide a great deal of communication about the design, other technical detail, my thoughts and reasons as to why I did a certain thing, etc. could only be communicated through annotations. Also annotations are a recurring theme in the examiners report. That is why I included annotations in my design development sketches.

PRODUCTION OF PROJECT

WORKFLOW

(See flowchart in figure 4.3, opposite next page) One of the main things that I have done differently in this project is the workflow or production pipeline I took. After making the final design sketches I would normally then go on to produce an orthogonal drawing from the sketch, and then an isometric drawing from the orthogonal drawing. But for this project I took an alternative approach, which in the end turned out to be a much better one. I decided that once I finish the final design sketches that I would then model my design in 3D on computer using Rhinoceros. From this one 3D model I could then generate orthogonal drawings, perspective drawings, isometric drawings, and then using that 3D model, I could use other applications to create rendered drawings. I could also take this 3D model and using rapid prototyping techniques, have a physical 3D model produced. However I would still need to use CAD software to edit the generated 2D drawings. If you look at the production flowchart in figure 4.3, you will see the general workflow that I took, from research, to designing, to producing. I have also shown where QC (quality control) occurred. Basically I looked at the product so far and checked it for errors or parts that were not of high quality, as specified in my *statement of intent*. If it was not up to scratch then I would fix it up before continuing. However once I had fixed it up, checked the quality, and after making any adjustments I got my teacher to check it for any advice on how to improve the quality.

TEMPLATES

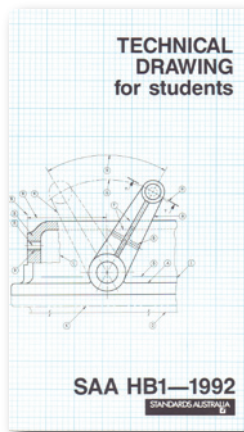
During the production of my project I took advantage of *templates*. By using templates I could load up my template which I had already made and start drawing, then when I saved the file it would be save in a new file, not overwriting the template file. In the template I specified things like;

- **Layers** - I set up different layers for different components such as hidden detail, centre lines, outlines, title block, annotations, etc. This meant that I could turn off certain layers so that I could see things clearly. It also meant that I only had to specify *lineweights* and *linetypes* once for the layer and then every line in that layer would have that *lineweight* and *linetype*. Also I used a layer which contained any construction lines, this layer was not set to plot. That means that I could view the lines when working but they would not appear on the printed drawings.
- **Standards** - By default AutoCAD is not set up to use AS1100. The advantage of implementing AS1100 into a template is that I only need to specify them once and the computer will use them for all my drawings. Also this method reduces the risk that I might forget or miss out on setting a setting. One of the things that I set was *dimension styles* (see figure 4.2), this let



- 4. SELECTION AND JUSTIFICATION -

me choose the specifications of sizes of parts or dimensions and set them to AS1100. I also used ISO A standards for drawing sheet sizes (eg. A4-A0), however the Australian Standard for these paper sizes is the ISO standard. I used the following publication to get the specifications of AS1100.



TECHNICAL DRAWING for students
SAA HB1-1992
Standards Australia
ISBN 0 7262 7876 9

- **Layouts** - I could have a range of preset layouts to choose from. More information about layouts is below.

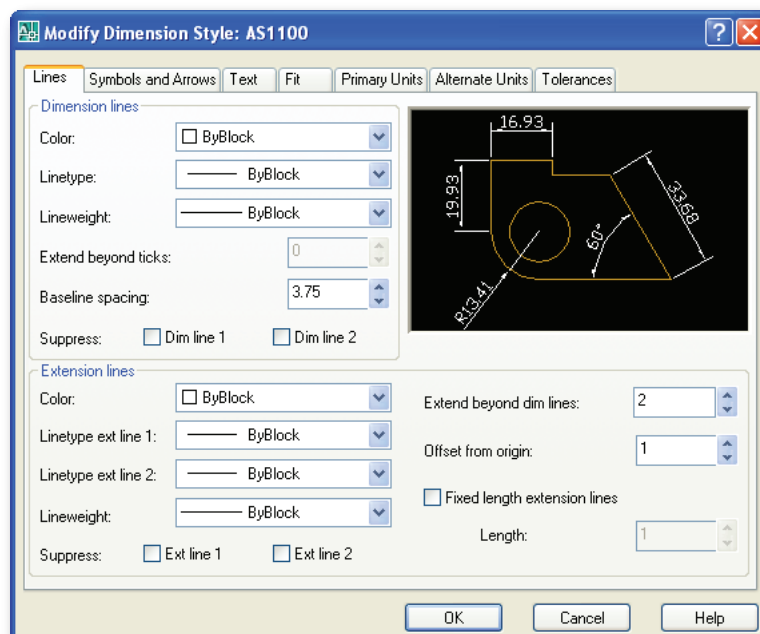


Figure 4.2 - Dimension Style Dialogue Box
(from AutoCAD® 2007)

LAYOUTS

During the production of my project I also took advantage of *layouts*. I used *layouts* to apply title blocks to my drawings. In AutoCAD, you draw in *model space*, then you put this drawing into a *layout*, which looks like the paper you will print on. In my template I set up five *layouts* (A4, A3, A2, A1, A0). In each of these I made a title block, this way I could draw in *model space* at full scale, and then choose which size paper I wanted and plot that *layout* with the *layout* managing any scaling.

LAYERS IN RHINOCEROS®

Although layers have been previously mentioned I used layers in a different way between AutoCAD and Rhinoceros. The layers previously mentioned was how and why I used layers in AutoCAD. In Rhinoceros I used layers to separate the different elements and components of my watch. So I had a layer for the timer, body, watch hands, watch face, band and clip. For each of these layers I had sub-layers of final, construction, sub components etc. This meant that I could keep construction



PRODUCTION FLOWCHART

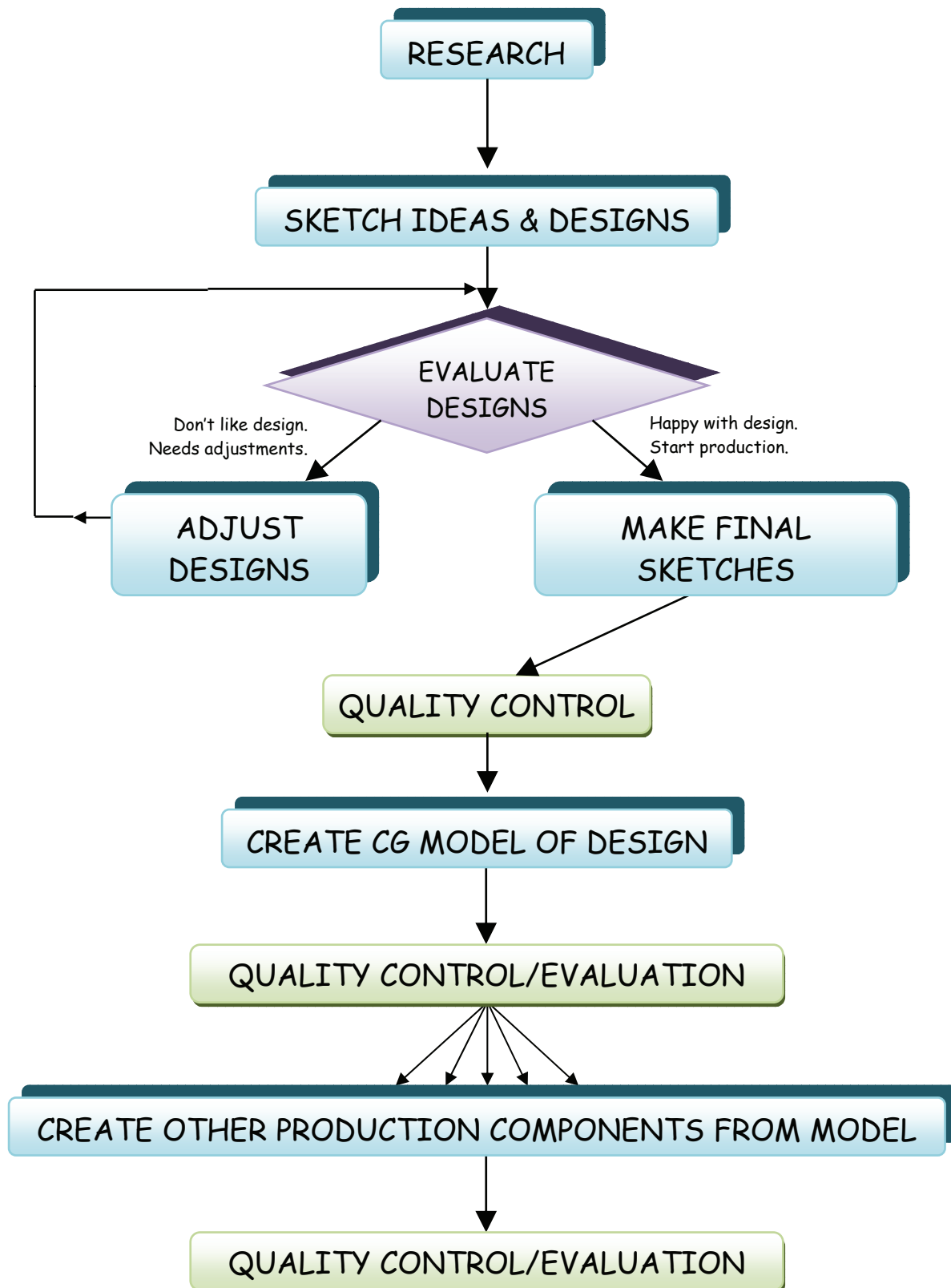


Figure 4.3 - Production Flowchart



- 4. SELECTION AND JUSTIFICATION -

data and other items which were not part of the final model geometry. I could easily turn certain layers and sub-layers off so that I could only see certain parts at a time. This increased efficiency and accuracy when I was producing separate drawings for different parts. Also the fact that I could turn off layers that I was not working on at the time meant that the software ran faster and crashed less as there was less geometry in the scene to display.

APPROPRIATE DPI

From my research I found that I should use at least 300dpi when printing. For my production component there was two main types of printing done. Plotting, which was used to print all the big (bigger than A3) line drawings, such as my orthogonal drawing. The other type I used was inkjet/laser printing for printing of photorealistic images, such as my rendered drawings. For the printing of the second type I had to devise a table, shown in figure 4.4. I used an inkjet A3 printer for A3 line drawings and a laser A4 printer for A4 line drawings because of easy of use.

	Dimensions	Aspect Ratio	300dpi	600dpi
A5	210mm × 148mm	1.42	2480 × 1748	4961 × 3496
A4	297mm × 210mm	1.41	3508 × 2480	7016 × 4961
A3	420mm × 297mm	1.41	4961 × 3508	9921 × 7016
A2	594mm × 420mm	1.41	7016 × 4961	14031 × 9921
A1	841mm × 594mm	1.42	9933 × 7016	19866 × 14031
A0	1189mm × 841mm	1.41	14043 × 9933	28087 × 19866
	6" × 4"	1.5	1800 × 1200	3600 × 2400
	8" × 6"	1.33	2400 × 1800	4800 × 3600

Figure 4.4 - Image Resolutions for specific sheet sizes

PRODUCTION ITEM DATABASE

I used a database to keep track of all my drawings, renderings, physical models, design sketches, etc. I used Microsoft Access 2003 for the database because it came bundled with Microsoft Office 2003 which I already owned. Information that I stored in the database (ie. the fields) is shown in figure 4.6, which is also an image of the form that I created to ease data entry. Figure 4.5 shows an one of the printouts of the information that the database held.

One thing that I would have liked to do but could not do due to the difficulties in communicating between Access and AutoCAD, was to link the fields in the title block to the data base so that editing on will update the other. This means that I could easily and accurately keep track of revisions and latest dates of drawings, and also the drawings that I had produced. In industry this would be important as a manufacturer could easier access the database and see if the drawing they had was the latest revision. They could also electronically get access to each of the drawings and each revision of each drawing. Autodesk Vault is software that allows for drawings and revisions of drawings to be managed. It also allows for easily collaboration between multiple workers on a single project.

The range of data that I stored was diverse, and ranged from the drawing projection, to if it was sectioned, to the sheet size, to the rendering engine used, to the resolution. However not all the fields were applicable to all my production items. For ease of entry and to increase efficiency I used VBA technologies to customize my database entry form. An example of this the the 'now' command button that I programmed to when clicked, it would set the field to today's date.



Drawing No.	Technical Drawing?	Freehand Sketch?	Other?	Drawing Type	Rendered?	Exploded?	Assembly?	Sectioned?	Engineering Data?	Detailed Drawing?	Date Initiated	Date Finalised	Sheet Size	File Type	Application	Rendering Engine	Date Rendered	Rendering Duration	Image Resolution X	Image Resolution Y	Other Details	Location/File Path
28	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Orthogonal	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	24/12/2006	24/12/2006							0	0	0 Centre Connector	
29	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Orthogonal	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	24/12/2006	24/12/2006			AutoCAD 2007				0	0	0 Outer Connector Details	
30	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Isometric	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	24/12/2006	24/12/2006							0	0	0 Face Isometric	
31	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Orthogonal	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	7/01/2007	7/01/2007	A3		AutoCAD 2007				0	0	0 Clip Shell Casing	
32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Isometric	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	7/01/2007	7/01/2007							0	0	0 Clip Shell Casing	
33	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Orthogonal	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	8/01/2007	8/01/2007	A3		AutoCAD 2007				0	0	0 Clip Pin	E:\mydata\ITG_MAJOR\Drawings\Orthogonal\Parts\Pin\Ortho 01 001.dwg
34	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Isometric	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	8/01/2007	8/01/2007							0	0	0 Clip Pin #1 Isometric	
35	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Perspective	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	8/01/2007	8/01/2007	A3	3ds Max 9	mental ray 3.5				0	0	0 Rendered Clip Pin #1	
36	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Isometric	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	8/01/2007	8/01/2007							0	0	0 Expanded Clip	
37	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Orthogonal	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	8/01/2007	8/01/2007							0	0	0 Detailed Part C06 - Flap	
38	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Orthogonal	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	8/01/2007	8/01/2007							0	0	0 Upper Clip Orthogonal	
39	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Isometric	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	8/01/2007	8/01/2007							0	0	0 Upper Clip Isometric	
40	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Orthogonal	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	8/01/2007	8/01/2007							0	0	0 Upper Clip Development	
41	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Isometric	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	9/01/2007	9/01/2007							0	0	0 Pin #2 Assembled Section, Exploded Isometric	
42	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Various	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	9/01/2007	9/01/2007				mental ray 3.5	9/01/2007		0	0	0 Pin #2 - Part C09 Details	
43	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Orthogonal	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	9/01/2007	9/01/2007							0	0	0 Part C07 Details - Pin #2 Casing	
44	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Isometric	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	9/01/2007	9/01/2007							0	0	0 Pin #1+Casing Exploded Iso	
45	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Isometric	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	9/01/2007	9/01/2007							0	0	0 Pin #1 & Casing Assembled + Exploded Rendering	
46	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Orthogonal	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	9/01/2007	9/01/2007							0	0	0 C12 Details	
47	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Orthogonal	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	9/01/2007	9/01/2007							0	0	0 Part C13 Details	
48	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Orthogonal	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	10/01/2007	10/01/2007							0	0	0 Part C14 Details	
49	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Perspective	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	10/01/2007	10/01/2007							0	0	0 Ghosted Rendered Assembled Clip	
50	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Perspective	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	10/01/2007	10/01/2007							0	0	0 RENDERED OPEN ASSEMBLED CLIP	
51	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Isometric	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	10/01/2007	10/01/2007							0	0	0 Clip assembly details	
52	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	3D	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	11/01/2007	11/01/2007							0	0	0 3D Model Representation ACAD	
53	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Perspective	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	11/01/2007	12/01/2007	TGA	3ds Max 9	mental ray 3.5 Renderer	mental ray 3.5	11/01/2007	5h 14sec	4961	3508		E:\mydata\ITG_MAJOR\Drawings\Renderings\3ds Max - metalray\PI 53\ 1280.jpg
54	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Perspective	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	11/01/2007	11/01/2007	TIFF	3ds Max 9	mental ray 3.5 Renderer	mental ray 3.5	11/01/2007	52min 32sec	1280	1024		E:\mydata\ITG_MAJOR\Drawings\Renderings\3ds Max - metalray\PI 54\08 01 (01)0000.jpg
55	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Perspective	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	11/01/2007	11/01/2007	TGA	3ds Max 9	mental ray 3.5 Renderer	mental ray 3.5	11/01/2007	3h 45min	0	0		E:\mydata\ITG_MAJOR\Drawings\Renderings\3ds Max - metalray\PI 55\08 05.jpg
56	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Perspective	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	17/01/2007	17/01/2007							0	0	0 Band #1 Persp	
57	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Ste Perspective	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	22/01/2007	22/01/2007	PNG	Rhinoceros 4.0	Rhino Render	Rhino Render	22/01/2007		0	0		

Friday, 4 May 2007

Figure 4.5 - Printout of Database
(Generated from Microsoft Access Database)

Item/Drawing Number

- Category -

Drawing? ☐

Rendering? ☐

Freehand Sketch? ☐

Other? Specify>

Drawing Type

- Options -

Rendered? ☐

Exploded? ☐

Assembly? ☐

Sectioned? ☐

Engineering Data? ☐

- General Details -

Date Initiated

Date Finalised

Sheet Size

Location/File Path

File Type

Application

- Renderings -

Rendering Engine

Date Rendered

Rendering Duration

Image Resolution X

Image Resolution Y

Other Details:

Thumbnail

Figure 4.6 - Database Form Entry
(Form created in Microsoft Access)



IN-HOUSE DEVELOPED SOFTWARE

This was used so that I could do things that were not directly built into the software that I was using. This can be split into two different types of in-house developed software. Plug-ins, which were developed to extend the functionality of an existing externally developed software solutions. And Utilities, which did not work directly inside another program, rather they worked independently. In some cases however they either used or produced data which came/went into other externally developed software solutions. See Appendix A, page 138, for details about the In-House developed software used and created.

XREFS

I used Xrefs in AutoCAD because it meant that if I made a change to one drawing it would be changed in subsequent drawings that were based upon that first drawing. This was especially made use of in the orthographic drawings of the whole watch. As I made a range of orthographic drawings each slightly different, but all still had the same raw model data. Using this process (as part of quality control) ensured quality of the product as it meant that no two drawings would conflict with each other. It also made it easier for me, as if I found that a change was needed I only had to make the change once on one drawing instead of changing all the drawings that had that part in it.

RAPID PROTOTYPING

Rapid Prototyping is the process of creating a real life physical model of 3D computer model data. As the name suggests it is mostly used for producing prototypes of designs and solutions, quickly. Although there are many methods of visualisation available (many I have explored in my project, including anaglyphs and real time 3D visualisation), the only way to see your design exactly as it will be in real life and be able to pick it up and test it in location is to produce a prototype. Some methods of rapid prototyping include cutting machines, CNC milling, stereolithography and 3D printers.

Cutting machines (ROLAND is one company that produces the equipment) basically just either by laser or other cut out 2D shapes. This is useful for cutting out developments and then manually constructing the design from the development. I did not use this technology because I did not have time and I knew that it would be very hard and very impractical to generate and cut out a development of my watch and then construct it, especially when there were much easier, better and more accurate technologies available.

Another technology is CNC milling, in which a milling machining is controlled by a computer and the data is sent to the computer from 3D model data. Simple CNC milling machines use only two axes, however the better ones have six degrees of freedom. But even with the best CNC milling machine I could not produce my whole watch prototype as one piece (due to the small detail in hard to reach places), it could however be produced in parts and then assembled manually. CNC milling allows for the model to be produced from many materials including steel and aluminium. Milling is a subtractive process, meaning that you start with a block of material and parts are removed, so when finished you are only left with the model.

The stereolithography process hardens liquid polymer with two computer controlled laser beams. Stereolithography is a additive process, meaning you start with nothing and the material of the model is added. It can contain much detail and can be formed from a range of polymers, including transparent polymers.



Figure 4.7 - A prototype created using stereolithography
<http://ocw.mit.edu/OcwWeb/index.htm>, see reference [11]



3D printers build 3D models in layers, where layers of material are added. 3D printers are made by companies such as ZCORP and Solid Scape. 3D printing was the cheapest and easiest technology for me to use. The only problem is that the machines can be expensive. So instead of purchasing my own I used a services which provided the 3D printing of 3D computer models.



Figure 4.8 - A prototype created using a 3D printer
<http://ocw.mit.edu/OcwWeb/index.htm>, see reference [11]

There are also other technologies available but I did not investigate them in as much depth.

MANAGEMENT FOLIO

STYLES

I used the process of styles in the production of my management folio. A style is just a way something is formatted. For example, in this folio I had a *body text* style which I applied to any regular text that I typed, like this here; a *caption* style which I applied to all the captions; various levels of titles and many more. The use of this process has increased efficiency of my folio production. It has meant that I only have to choose a style for the text I am typing instead of changing all the components individually, like the size, colour, font, etc. Using styles has also meant that when I wanted to change the look of my body text, which I did have to do at one stage, I could just change the style without needing to individually select all the body text and then changing the font, size, etc. individually.

MASTER PAGES (AKA. TEMPLATE PAGES)

I used master pages to put things that all the pages would have. For example I used master pages to get the footer at the bottom on all the pages, without having to copy and paste it to all the pages. This also meant that when I wanted to make a change to the footer, which I did (I changed the linewidth of the line), I could change it once for each of the two master pages (right and left pages used different templates), and the changes would be applied throughout the folio instead of having to change each page individually.

FOMULAE IN SPREADSHEETS

I used many features in Microsoft Excel to aid me in my timeline plan. Some of the data that I decided to include in my timeline plan is shown below as I arranged it into different sheets. My timeline plan can be found on page 72.

Proposed Workload	Actual Workload	Overlap Workload	Proposed Time	Actual Time	Overlap
-------------------	-----------------	------------------	---------------	-------------	---------

I used *formulae*, *charts*, *multiple sheets* and *conditional formatting* to automate many tasks. This reduces the amount of manual labour and time that I spent working on it. It also reduces, if not



- 4. SELECTION AND JUSTIFICATION -

eradicates, the chance of error. Once I set up the layout of my timeline, I made two versions of it each on a separate sheet, one for proposed and one for actual. Along the left I had the task and along the top I had the week number. So for my proposed and actual sheets if I had either proposed or actually done a task in that week I respectively set the value of that cell to 1. I also turned on *conditional formatting* and I choose the option that if the cell value was equal to 1 then the colour of the text and the background colour would be set to a colour dependent on which sheet it was in, proposed or actual. Now I could use the sum function to sum all the cells in each column and then I could obtain a workload value along the bottom. I could then take all the workload values and create a graph using the built-in graph wizard.

For the overlap sheet, I could set the cell value to either 1, 2, 3 or 4, being none, both, proposed only and actual only respectively. This was done with the formula:

```
=IF('Actual Time'!D5=1,IF('Proposed Time'!D5=1,2,4),IF('Proposed Time'!D5=0,1,3))
```

I then set a different colour depending on the cell value. Now everything would automatically update and change colour by only typing 0 or 1 in the proposed and/or actual sheet. This improved efficiency and ensured quality control and ease of use.

RESOURCES

SKETCHING AND DESIGNING

GRIDS

I used a variety of grids such as square, isometric and perspective. I also made my own grid which was based concentric circles. This helped me in sketching top views of my watch designs. Instead of drawing on the grid, I placed the grid paper behind the paper I was sketching on. That way I could see and use the grid but it didn't show up on my drawings. I did this because I believe that my drawings without the grid are easier to see, whereas if a grid was present certain parts might be confusing, clear or misinterpreted.



Figure 4.9 - Various Grids
(Grids obtained from Alias Sketchbook Pro samples)

GRAPHICS TABLET: ACECAD® ACECAT® FLAIR (5" x 3.75")

I used a graphics tablet to help me do some of my freehand sketches on computer. Advantages of this are that I can work with layers, I can undo steps much easier, and if done well it can look good. Using a graphics tablet made sketching much easier than using a mouse. It also allowed for pressure sensitivity. I used this brand over the more common and popular Wacom brand because it was cheaper.



ALIAS® SKETCHBOOK™ PRO 2.0

I used this for sketching my watch designs on computer. SketchBook Pro was easy to use and learn quickly. It was very responsive and fast which was the dominating factor making it the best product, as Photoshop often lagged when drawing lines with a graphics tablet. This makes sketching almost impossible and thus I used SketchBook Pro.

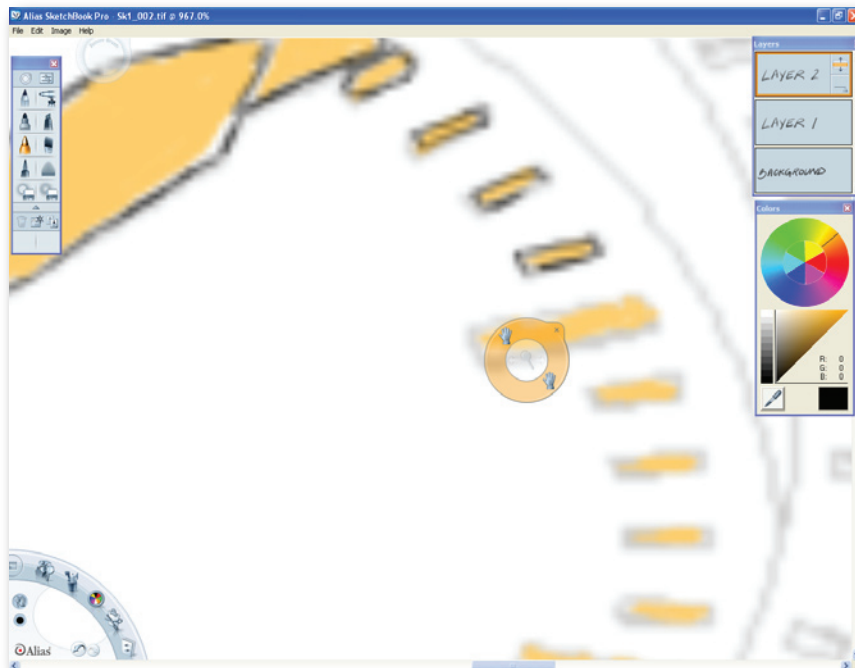


Figure 4.10 - Alias® SketchBook™ Pro 2.0, featuring my watch sketch

PRODUCTION OF PROJECT

Almost all of my project was done on computer. I used a variety of software to produce the product. The following is a list of the resources I used and why I used them. *(Software boxes and logo's from internet source)*

OPTIMA WORKPRO D9S PC (MICROSOFT® WINDOWS® XP)



Intel Pentium 4 Processor 3.2GHz
1GB RAM
160GB Hard Drive
DVD-RW Burner
17" LCD Monitor

I choose to use a Windows based system mainly because most of the software that I planned to use was compatible with Windows. Also because I was familiar with Windows XP, unlike Mac or Linux which I don't know much about how to use. Also I already had a Windows system up and running so it would be quicker, easier, and cost less



- 4. SELECTION AND JUSTIFICATION -

for me to use the Windows system rather than to buy a Mac and have to set everything up. I used this model of computer because it reached the system requirements required by the software I used and I already had the computer.

AUTODESK® AutoCAD® 2007

There are two aspects to consider here, one is why I choose to use CAD rather than manual drafting methods and the other is why I chose AutoCAD rather than another CAD software.



In regards to why I chose to produce my drawings by CAD rather than manual methods, I chose CAD because I found by previous experience that CAD methods:

- allowed for faster production
- made the drawings look neater and more professional
- allowed for greater precision and accuracy
- allowed making modifications easier
- allowed certain tasks to be automated, eg. dimensioning.
- and many more...

I chose AutoCAD 2007 because I had already learn how to use AutoCAD at school and I was competent in using it. Also I tried other CAD software such as IMSI's TurboCAD but I found it difficult to use. I used the 2007 version because it had more functionality and an easier to use user interface than any other previous version.

RHINOCEROS® 4.0 BETA

I chose to use Rhinoceros because I found it to be the best product of its kind on the market whilst still being affordable. By using Rhinoceros I could make a 3D model of my wristwatch, to high precision, and then generate a range of drawings from that 3D model. Although fine tuning and dimensioning would still need to be done in AutoCAD. This 3D model could then be exported into other applications and be rendered to create my rendered drawings. Another factor as to why I used it is because my school had purchased a Lab Licence, which meant that I could use it for free. I also used Rhinoceros because it could run as an OLE Automation server. This means that Rhinoceros could link up with Microsoft Visual Basic allowing me to generate my own algorithms and do things that could have never been done, or as easily, in another program. The use of this can be seen in Selection and Justification → Processes → Production of Project → In-House Developed Software, page 60. The most contributing factor as to why I used Rhino is because I was more competent at using it than Autodesk Inventor, PTC Pro/Desktop or SolidWorks. (See also Evaluation → General Evaluation → 3D CAD/CAM Design Software, page 88.)



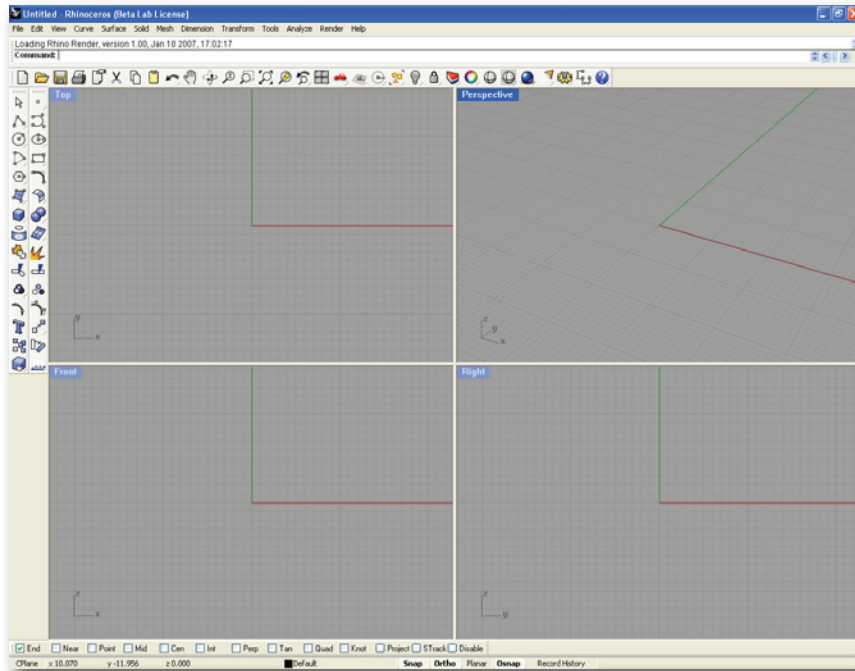


Figure 4.11 - Rhinoceros® 4.0 BETA Screenshot

AUTODESK® 3ds MAX® 9

I used 3ds Max 9 to produce rendered drawings. I used it to apply materials to the model from Rhinoceros and then render images of it in perspective projection. I used various rendering plug-ins including *Default Scanline Render*, *mental ray Render* and *VRay Render*.



I investigated into using Autodesk Maya 8.0 through the free Personal Learning Edition. However I found it extremely difficult and confusing to use. So I used 3ds Max 9 instead of Maya.

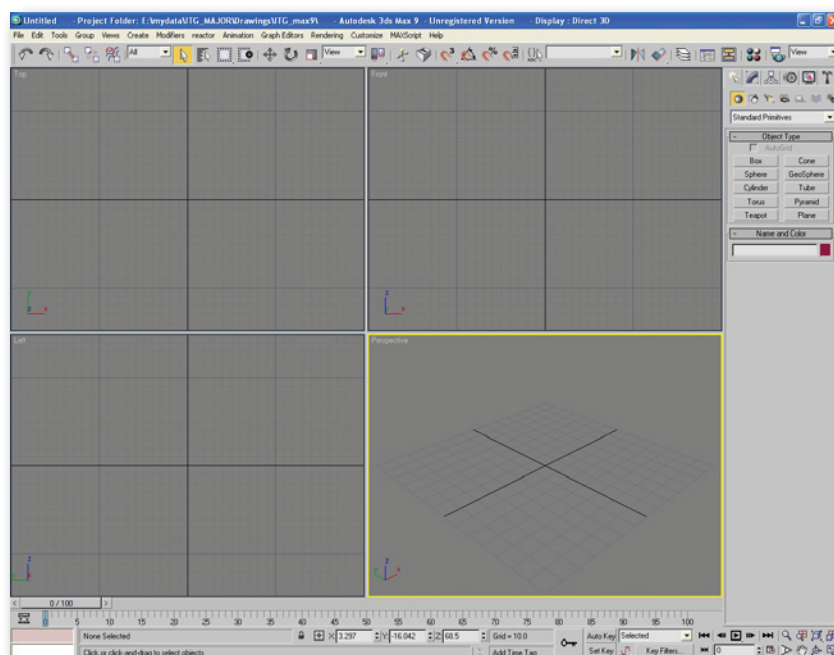


Figure 4.12 - Autodesk® 3ds Max® 9 Screenshot



ADOBE® PHOTOSHOP® CS2

(This item also appears in the *management folio* section)

I used Photoshop because it allowed me to apply adjustments to my rendered drawings. Things like depth of field, exposure, gamma adjustments, brightness and contrast adjustments could be made. I could also add backgrounds, logos, text, etc. In addition I could set up my image with the correct size and DPI for printing. I could also edit images and change them into different colour spaces depending upon the medium that it would be presented on. I could also edit and manage 32bit high dynamic range images (HDRI).

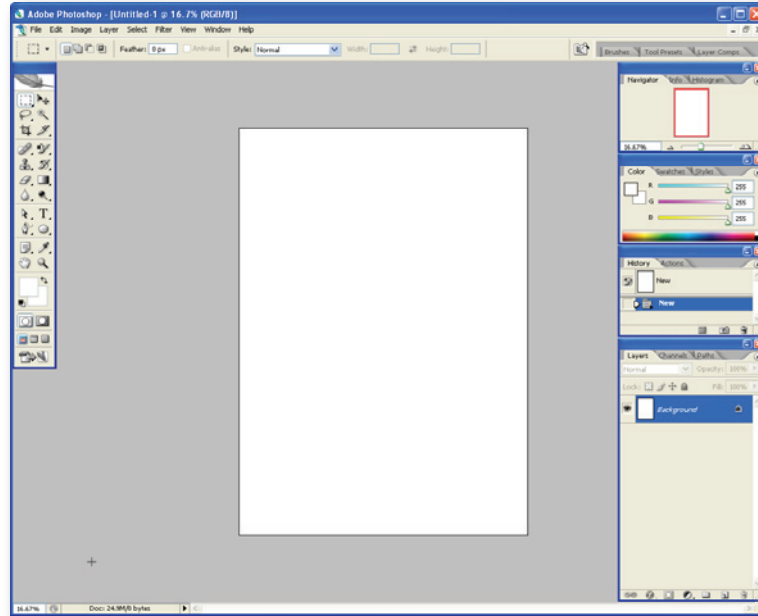


Figure 4.13 - Adobe® Photoshop® CS2 Screenshot

SAMSUNG® CLP-550N COLOUR LASER PRINTER

I use this printer because it could print at high DPI. This is important as the lines will appear more smooth and can be finer, without having to use an expensive plotter. I used this brand and model of printer because it was the most easily accessible one, since it was at school.



SCANNER

I used a scanner to scan images onto computer. I mainly used it for my freehand sketches and some images of watches from research to include them in my management folio. I also used the scanned design sketches to compile and edit them easier and also make combined and painted rendered sketches. I used the scanner available in the school library.

DIGITAL CAMERA

A digital camera was used to include photographic evidence of OHS issues in my management folio and for some parts of my production component. I borrowed this camera from the school to use for this project. When using the camera I set it to save images as RAW data. This allowed me more power when editing the photograph on computer.



MICROSOFT® VISUAL BASIC® 2005 EXPRESS EDITION

I used the express edition because it was free. I used the 2005 version because it was the newest and had the best features and functionality. It also allowed me to create better GUI's (Graphical User Interface's). I used Visual Basic rather than the more powerful and popular C++ because I was more familiar with Visual Basic. I used it to create self developed software, which can be found in Appendix A, page 138.

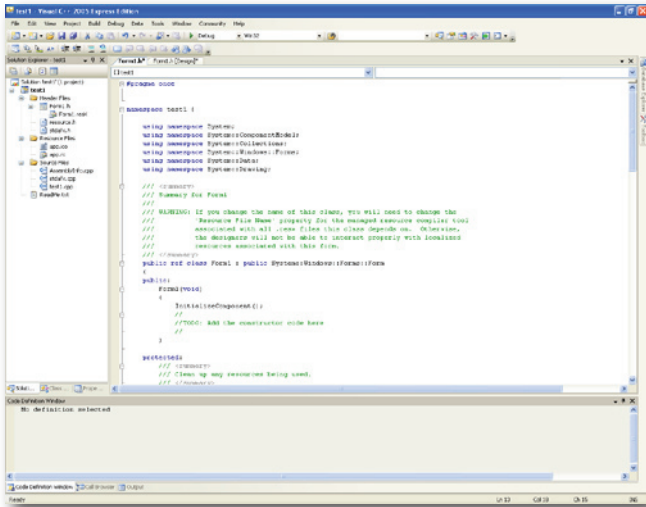


Figure 4.14 - Visual C++ 2005

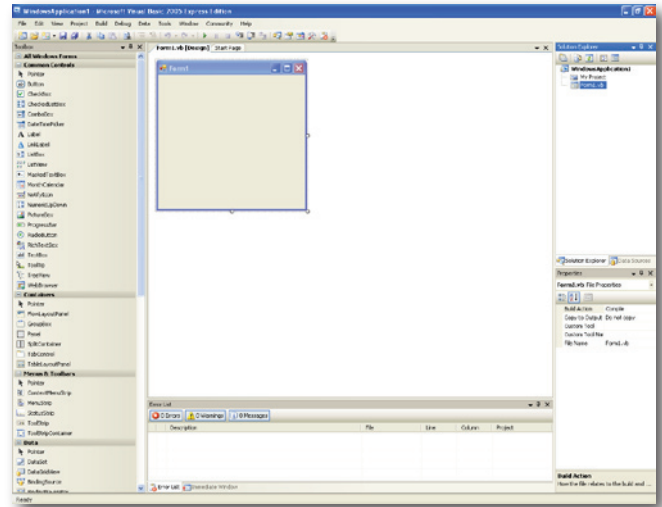


Figure 4.15 - Visual Basic® 2005



- 4. SELECTION AND JUSTIFICATION -

AUTODESK® IMPRESSION

Impression is software that allows you to produce drawings that look like they were produced by freehand and rendered freehand. The software takes your 2D line drawings and then you can make the lines look like pencil, charcoal, pen, etc. You can also add fill's such as gradient, watercolour, marker, texture, etc. I used this tool as I was not very good at producing freehand rendered sketches so I used it to aid me and still allow me to produce high quality sketches even though I am not skilled in that area.

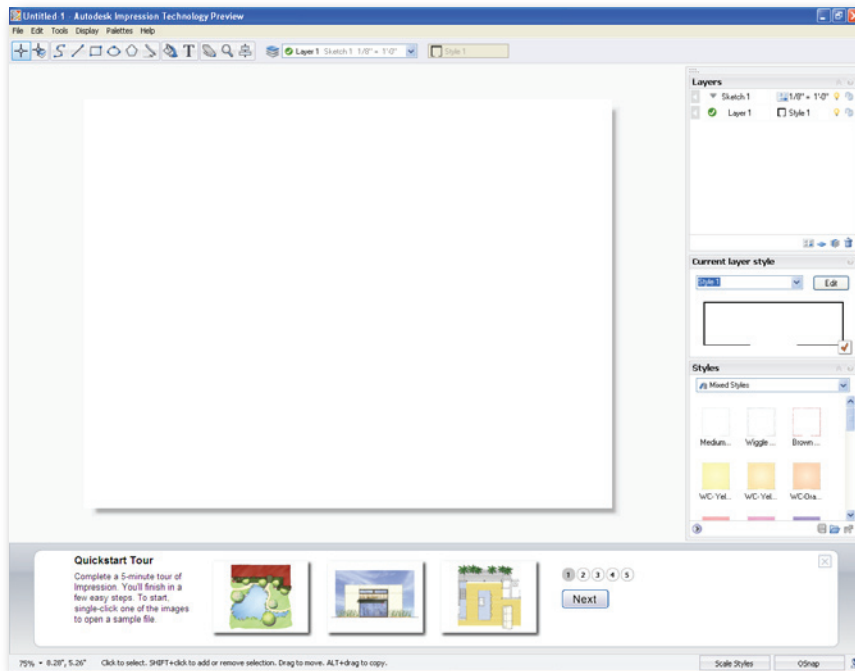
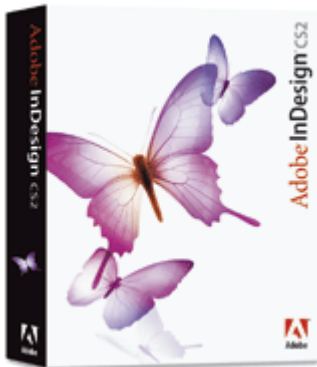


Figure 4.17 - Autodesk® Impression

MANAGEMENT FOLIO

ADOBE® INDESIGN® CS2

For the management folio I used Adobe InDesign CS2. I used InDesign because I thought it was the easiest to use and was cheaper than the other main desktop publishing software on the market including QuarkXPress and PageMaker, which I tried but as they seemed very similar to InDesign I used InDesign. Although I had never used InDesign before, so it was slightly harder to use than Microsoft Word, which I was competent in using, however I chose not to use Microsoft Word because I could not obtain the quality I wanted by using it. That is why I switched to InDesign. In the end it didn't take me long before I learnt how to use InDesign.



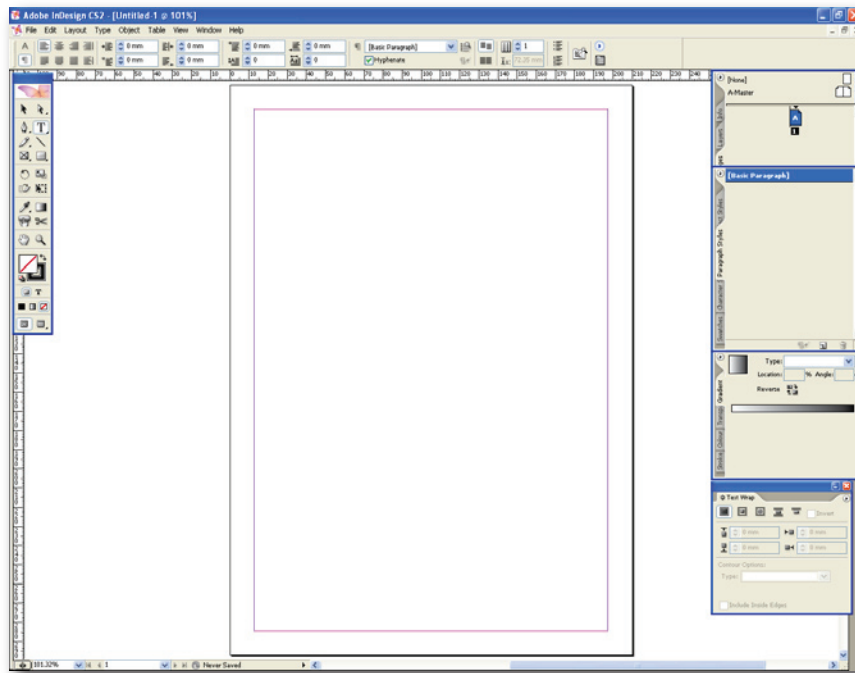


Figure 4.18 - InDesign® CS2 Screenshot

MICROSOFT® EXCEL 2003/2007

I used Excel for creating spread sheet's for my timeline plan and finance plan. I used it because I already knew how to use it, and I already had it on my computer. These two points were strong enough for me to use it rather than go searching for an expensive, more productive tool. In general, I used a spreadsheeting package, especially in the finance plan, because it can add up costs and preform other calculations automatically.



During the production of the project I upgraded to Microsoft Office 2007, which included many new features which made things easier for me.

MICROSOFT® WORD 2003/2007

I used Word for the flowcharts, as it was simple to use and allowed me to produce my flowcharts quickly and easily. Although they don't look as good as if I had have done them with a different program, using an alternate program would have taken way too much time.



ADOBE® PHOTOSHOP® CS2

(This item also appears in the *production* section)

I used Photoshop because I could easily edit graphics, including screenshots and images, and save them in the best format for its use. Some of the file formats I used and the advantages and reasons I used that specific format are found in the table below.

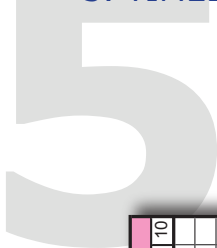
JPEG .jpg	I used JPEG format when the image was not going to be used for printing and no extra data apart from the RGB image was needed. Examples included test renderings.
PNG .png	I used PNG when I needed a lossless compression format, usually for print. Examples include screenshots.
TARGA .tga	I used TARGA format for all my non-test renderings. I used it because it used lossless compression and could contain transparency information, which proved very useful when creating presentation compositions.
PDF .pdf	I used PDF to save files which would be used for printing or distributing. Examples include CAD drawings and spreadsheets for import into folio. This is because PDF would save the lines in vector format, not raster which would result in extremely large file formats in order to retain detail.



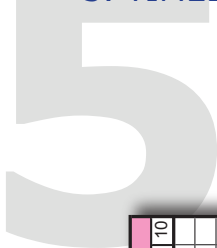


10		
----	--	--

10		
----	--	--





10		
----	--	--



10		
----	--	--

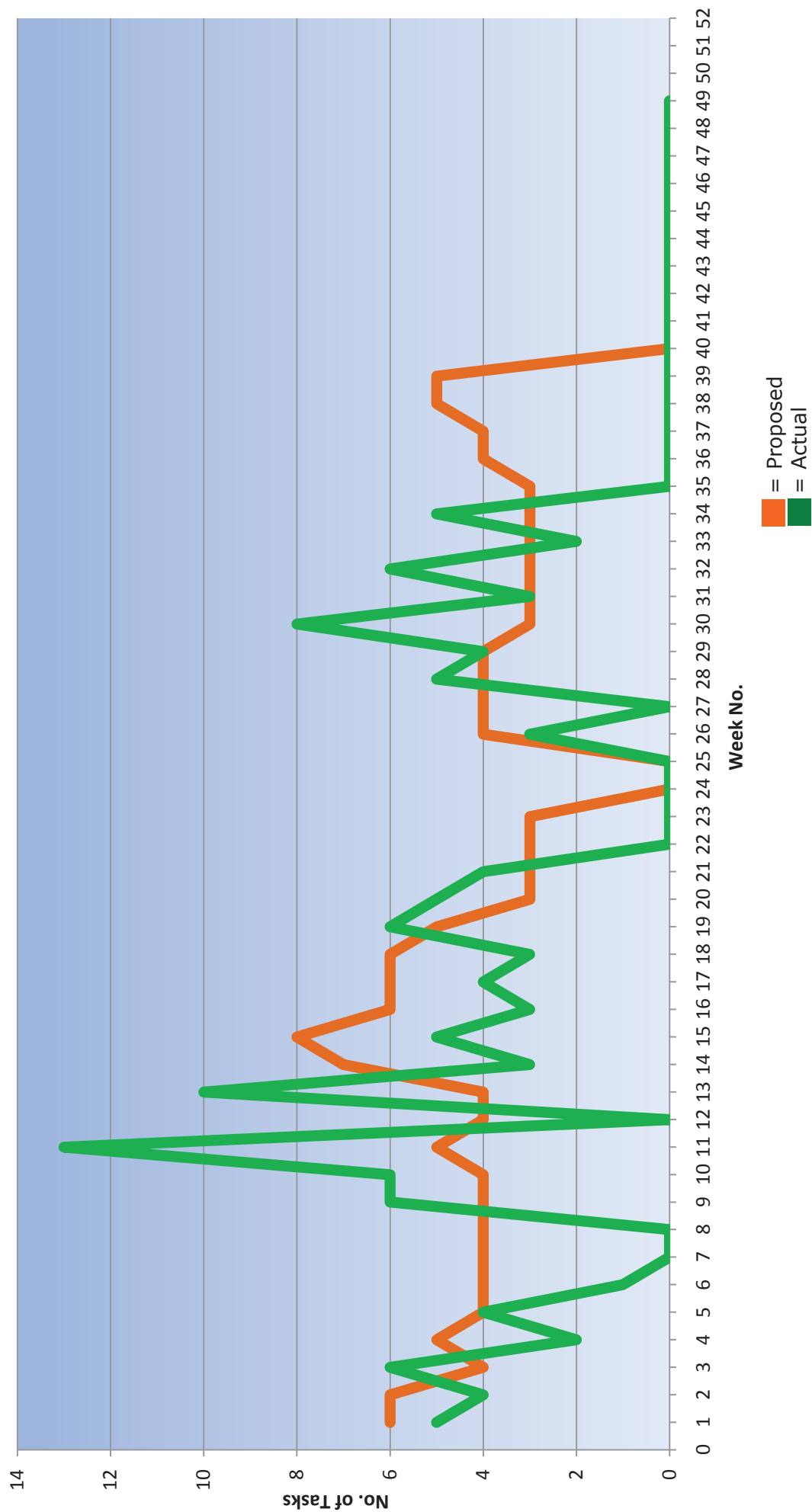
- 73 -

 = Proposed
 = Actual

NB. Weekends appear the end of the school week.

NB. Weekends appear the end of the school week.

Workload



GENERAL EVALUATION

I designed the timeline plan with school weeks, not dates as I thought that this would be easier than dates as I could plan when I would have the most free time to work on the project. I could also see when school events such as exams were on so that I could work around them. This means that hopefully I will have kept to the timeline plan much closer. Also the scale of my timeline plan is weekly, I chose this over daily and monthly, because monthly is too big blocked and there is not enough detail and I didn't chose daily because it is too detailed and there would be no room for readjustments of tasks within a week. By working on a weekly scale I could move things around inside that week. For example if I couldn't work on a particular day for an unexpected reason, then I could catch up on the next days in that week, without disrupting the whole plan.

Because I was so keen on working on my project and because I enjoyed it and spent too much of my time working on it, I did not need to worry about not finishing the project on time. What the timeline did help me with was to ensure that I did not get side tracked too much and spend too much time on tasks that were less important.

Some key things that I had to think of when planning the timeline plan were:

- What would my school workload be at that time. This means that I tried to minimise the work to do during exam periods and school terms in general, so I tried to do large amounts of the work during school holidays. To incorporate this into my timeline I marked exam periods and school holidays along the top.
- Not to plan too many task at the same time. So as you can notice along the bottom I put the workload at appropriate times. However this is not an exact measure, as not all the tasks have equal workload. A graph of this can be seen on the opposite page.
- From previous experience in the preliminary year, I found that the design development stage takes a lot of time. I also learnt that this is an important stage, as the rest of the drawings will be based on my final design. I learnt that this stage needs to be perfect before moving on to the production stage. With this knowledge I planned my timeline with that in mind, I allowed the whole of the first school term to work on the design stage.

Also when I planed out my proposed timeline plan the final due date of the project had not been published, however this did not affect me because I knew it would be sometime in term 3 and I planned to be finished before term 3 began anyway.

Processes used for the timeline plan can be found in Selection and Justification → Processes → Management Folio → Fomulae in Spreadsheets, page 61.

EVALUATION OF COMPONENTS

STATEMENT OF INTENT

I planned for this to be completed in the first week, actually I thought that I could write it up in the first day. It ended up taking the whole week, I started it on the first day, but each day I read it and found things that I thought needed changing, so after making several alterations over several days, I managed to finish it in the first week.

RESEARCH

My proposed time for research was correct relative to the development of ideas section. I did the two together. I collected some images, details and specifications of existing watches whilst developing my own design solutions. However as I have described in the section below this process was done much later than proposed.



- 5. TIMELINE PLAN -

DEVELOPMENT OF IDEAS

From previous experience (ie. a year 11 assessment task I completed) I had found that this development of ideas stage could take a long time. So I planned to allow nine weeks for this. Which included using the research I collected about watches and developing a design of my own. Which would satisfied my *statement of intent*, along with producing sketches of these designs. However I only actually started to do wristwatch research and start designing the watch by the end of term 4 2006. The reasons for this include; because the other parts of this management folio took longer than I expected; and I had underestimated the amount of work I had from other school subjects. This was not too much of a problem because I used the school holidays to get the designing of the watch and documentation of this in the development of ideas section completed.

SELECTION AND JUSTIFICATION

I only proposed two weeks for this part, and it really only took a bit over two weeks. However I found that I could not do the whole section at weeks 2 and 3 as proposed, because some materials, processes, components and equipment were selected after that date. So I added those things later once I had used them. So the selection and justification was ongoing.

TIMELINE PLAN

The proposed timeline plan was done in as predicted in the first two weeks. However I used another week later to make slight adjustments to the timeline plan infrastructure. The actual timeline plan, was designed with the proposed, except each week I would update it with the tasks completed. As these additions were done every week and were minor and only required me to change a cell value to "1", I have not shown this on the timeline plan.

FINANCE PLAN

This only took two weeks to design and implement however this was actually delayed compared to the proposed. This was because I didn't have enough time to complete it at the beginning. However the delay didn't cause any problems. Similarly with the timeline plan I made minor adjustments along the period of the four terms, however these were just item amendments not design or infrastructure changes so I didn't show this on the actual timeline plan.

COMPILING THE FOLIO

This part turned out to be redundant. The reason is that when I was doing the proposed timeline, I thought that I would write up each part, ie. statement of intent, research, etc. and then once I had done it all, I would put all the text and images together in a word document. However, once I started the project, I realised that that wasn't a very good approach, so instead I just added the text for each part to the folio document as I worked on it. So that is why on the actual timeline, you will see that I have filled in this part with all the times that I am working on any part of the management folio. Compiling the folio was an ongoing task.

EVALUATION

Evaluation is a constant theme that came with every aspect of the project. *Ongoing evaluation* is required, that means evaluation at all times and of everything. That is why I planned to do evaluation the whole time, and I did. Whenever I did something I evaluated it. The evaluation is documented throughout the sections of this folio, however in the timeline plan all the evaluation is put in the evaluation time frame. However towards the end of the project I spent most of my time doing evaluation and working on the evaluation section, however this cannot be shown on my timeline plan due to the way it is designed.

Logo

I knew that the logo would need to be produced very early on, as it would be put on every part of the project. I allowed the first week to do it. After the first week, I had completed it. It only took about three hours or so, and I could now implement it into the rest of my project. I didn't put much thought or time into the development nor the design or the DD logo. Given that it has turned up on every part of my project, I should have spent more time on it, however, I just wanted to make



one really quickly so that I could begin with the rest of my project, and it would be much harder to implement the logo post production of the folio and the product, so I stuck with my initial design.

OH&S ISSUES

I would consider OH&S issues before I started and then comment on them at the end. That is why the proposed parts are at the start and then again at the end.

PROBLEMS IN PRODUCTION

When making the proposed timeline, I realised that during the production of my project I would encounter many problems and I knew that I needed to document these. So I allocated the whole time that I was working on the production of the project to also documenting any problems. I didn't know exactly when problems would occur until I uncovered them, that is one reason why the allocated time spreads out over the whole production time. But the second reason is that I thought, and it turned out to be, easier to document the problems when they occurred, rather than leave it to the end and try to remember all the problems and how I overcame these.

DEVELOPED SOFTWARE

This appears on the actual plan but not the proposed plan. This is because I never planned to have to develop my own software to do what I wanted to do. Only once I actually tried to do it, and thought about how I would do it, is when I realised that I would need to develop my own software to do what I wanted. However when I realised that I did need to develop the custom software, I thought that it would take one week at maximum. However I soon realised that it would take much longer. This was due to the problems that arose, which are partially documented in, Appendix A - Developed Software → ViewIt → Problems and Solutions, page 195.

The bulk of the program was completed by week 2, term 1. I used week 5-6 to make small adjustments and improvements and also work on the documentation. This documentation took the bulk of the two weeks.

FREEHAND SKETCHES

As I did freehand sketches during my development of ideas, see Timeline Plan → Evaluation of Components → Development of Ideas, page 76.

3D MODELLING

This is shown as taking two weeks, however I had most of the modelling completed in the first two days of beginning this process. The reason for this is because I spent the whole of those two days on the modelling and nothing else. The spinning timer, watch face, watch hands and watch body only took these first two days to model. However the band and fastening device took longer and were done after producing some 2D drawings of the parts that I had already modelled.

Originally I had predicted that the 3D model would take the whole five week holiday to complete and the 2D drawings would each only take two or so weeks during the school term. However I found that the 3D modelling took only a small amount of time and the 2D drawings took much longer to complete.

ORTHOGONAL/ISOMETRIC/OBLIQUE/PERSPECTIVE/EXPLODED DRAWINGS

As stated above these drawings took much longer than I originally planned. Although according to my timeline this is not the case. On my timeline it shows that I took less time on these drawings than originally planned. Although the truth is that it took longer than estimated. The reason for this discrepancy is because I actually completed these tasks mostly during the holidays where I could spend much more hours on the task that I could have if it was done during school term, as I had originally planned. The reason why I only proposed a small amount of time on these drawings because I thought that since I could generate these drawings from the 3D model, they would just take a couple of hours, but then I realised only the model lines would be generated and I would need to make many adjustments for the drawings to be correct to AS1100 and of best quality.



RENDERED DRAWINGS

Once I had finished and even when I had not finished the 3D model I could start making rendered drawings. My initial rendered drawings that were created before I had completed the model were only simple and I didn't consume much time, whereas the more photorealistic rendered drawings which were produced after I had completed the model took much longer to produce and also much longer to render. Some of the high quality renders took up to 48 hours to render. Solutions to this problem can be found in Solutions to Problems in Production → Computer Rendering Problems → Too Slow, page 134.

Rendering was kind of an ongoing process, due to the long rendering times and thousands of options getting an image that looked good took a long time.

PLANOMETRIC DRAWINGS/DIMETRIC/TRIMETRIC DRAWINGS

Relative to the other drawings produced these were done much later. This is because I could only start these once I had finished producing ViewIt. Once the software was complete I could make drawings in these projections, which took relatively little time, less than one week.

PHYSICAL MODEL

If my school or I owned a rapid prototyping machine, then this task would not take longer than a couple of hours. I would just need to fine tune my model and then send it to the 3D printer. However as I had to outsource the 3D printing because my school or I did not own the machine, the task took longer. This was because I had to get quotes, then send the file off and then receive the physical model through the mail.

DRAWING TEMPLATE

I completed this a week earlier than planned as school finished mid week 10. It took the same duration as planned, it was actually less than a week, just two days.

STRESS ANALYSIS DRAWINGS

I never completed these drawings. This is mainly because I did not have time, and I could not obtain the appropriate engineering CAD software, such as Solidworks, NX, Pro/ENGINEER or Inventor with engineering add-on's.

QUALITY CONTROL

I had not pre-allocated time to do Quality Control because I thought that it would take a relatively small amount of time. However during production I found that assuring that my product was of high quality was much harder and took much more time than I originally thought. More details about the quality control processes I used can be found in Selection and Justification → Processes → Production of Project → Workflow, page 55.





FINANCE PLAN

The funding of this project will come from my personal finances. My budget for this project is \$400.00, however for this project additional finances are available if I go over my budget.

PRE-PURCHASE PLAN (PROPOSED)

	Item	Supplier	Quantity/ License	Estimated Cost	Researched Cost	Difference
MANAGEMENT FOLIO	InDesign CS2	 studentdiscounts	1 Student Lic.	\$0.00	\$0.00	\$0.00
	Microsoft Office 2003		1	\$0.00	\$0.00	\$0.00
	A4 Paper		500 sheets	\$5.00	\$4.95	\$0.05
	A4 Plastic Sleeve Folder		3 @ \$1.50	\$4.50	\$1.50	\$3.00
	Internet Use		Cable	\$0.00	\$0.00	\$0.00
COMMON	Laser Printer		1	\$0.00	\$0.00	\$0.00
	Printer Inks (BW+Colour)		1	\$0.00	\$0.00	\$0.00
	Home Computer		1	\$0.00	\$0.00	\$0.00
	School Computer		1	\$0.00	\$0.00	\$0.00
	SanDisk 512MB Cruiser	Dick Smith Electronics	1	\$0.00	\$0.00	\$0.00
	Electricity			\$0.00	\$0.00	\$0.00
	Photoshop CS2		1	\$150.00	\$150.00	\$0.00
	Telephone Calls		5 @ \$0.20	\$3.00	\$1.00	\$2.00
	Standard Glossy Photo Paper		75 sheets	\$20.00	\$19.95	\$0.05
	Sketching Tools	VARIOUS		\$5.00	>\$5.00	\$0.00
PRODUCTION	AutoCAD 2007		Student License	\$0.00	\$0.00	\$0.00
	Rhinoceros 4.0 BETA	Blakehurst High	Student Lab License	\$0.00	\$0.00	\$0.00
	3ds Max 9		30 day trial	\$0.00	\$0.00	\$0.00
	Prototype (Physical Model)		1	\$50.00	\$145.00	-\$95.00
	A4 Paper		500 sheets	\$5.00	\$4.95	\$0.05
	A3 Paper		250 sheets	\$0.00	\$0.00	\$0.00
	A3 Plastic Sleeve Folder		1	\$12.00	\$27.00	-\$15.00
	A0-A2 Plotting	Friend		\$0.00	\$0.00	\$0.00
				\$254.50	\$359.35	-\$104.85

Key:

blue = already owned (non consumable)
green = free from school

Corporation and company logo's in this finance plan may be registered trademarks and/or copyright of their respective owners.



FINANCIAL STATEMENT (ACTUAL)

Date	Transaction	Debit (-)	Balance
16-Oct-06	Opening Balance		\$200.00
21-Oct-06	SanDisk 512MB Cruzer	\$49.95	\$150.05
23-Oct-06	A4 Paper (500 Sheets)	\$4.95	\$145.10
4-Nov-06	A4 Plastic Sleeve Folder (Qty: 5)	\$7.50	\$137.60
5-Nov-06	Sketching Tools	\$5.00	\$132.60
13-Jan-07	A4 Plastic Sleeve Folder (Qty: 1)	\$1.95	\$130.65
18-Jan-06	A3 Plastic Sleeve Folder	\$27.00	\$103.65
19-Jan-06	DVD ± RW (10 pack)	\$9.95	\$93.70
12-Jun-07	A3/A2 Colour Printing (Office Works)	\$33.50	\$60.20
20-Jun-07	Physical Model (3H Engineering)	\$37.40	\$22.80
22-Jun-07	Closing Balance	\$177.20	\$22.80

Some items were not included on my financial statement, because I did not buy the software solely for this project. Therefore it was not included. In an industry situation, almost all of the software needed would be bought by the company to be used on all projects, not specifically to only be used for one project, but to be used for all project. This cost would not contribute to the project budget.

Two finance plans are shown. The Pre-Purchase Plan (Proposed), on the opposite page, was done when I was planning the project before I began production. Many of the items that I would need were provided free from school or were non-consumable items that I either already owned or would not just use solely for this project. This is shown by the green or blue text. Also I did not end up needing all the things that I originally planned, similarly some items I had to buy which I didn't plan on needed. The Financial Statement (Actual) above shows exactly what I did buy and how much it cost.

The total cost of my project was \$177.20 which was well below my budget of \$400.00.

GENERAL EVALUATION

I found producing a finance plan the most challenging part of the management folio. It was made difficult by several factors including that some items I had already purchased before the production of the project. That means that yes I did consume money on those items, but I had already purchased them for other things not related to the project, so they didn't really contribute to the cost of the project. Also other items were provided free from the school, so in the real world they would cost money, but in this case they didn't.

Most of my resources were purchased from Officeworks. This is because it was convenient for me to buy most of the things there rather than visit different shops and compare prices.

For the physical model, I got quotes from several different companies, and choose 3H because it was cheapest and reliable.



7 USE OF APPROPRIATE INDUSTRIAL PROCESSES AND EQUIPMENT

This section has been placed here because it is part of the assessment criteria for the management folio. However everything that would be found in this section can also be found under other criteria. So instead of just repeating things that have already been documented I will refer you to the relevant sections.

Documentation and evaluation of industrial processes and equipment used can be found in Selection and Justification → Processes, page 55 and Selection and Justification → Resources, page 62. Evidence of the use of these processes and equipment can be found by examining the production component and the evaluation of the production components. Evidence can also be found in the Evidence of Industrial Processes/Technologies section on page 118.





8 EVIDENCE OF SAFE WORKING PRACTICES AND OH&S ISSUES

The number of OH&S issues that I dealt with, was limited due to the industry option undertaken. However each one was still extremely important. First I assessed the risks, then I tried to prevent them, then I tried to minimise the risks, then I kept myself well informed about the risks and how to avoid any problems.

Though my OH&S investigation that I made before I began production, I uncovered many OH&S issues. The issue, why it is an OH&S issue and what I did about it is listed below.

COMPUTER RELATED ISSUES

EYESIGHT/MONITOR USE

As most of my major project is computer based, this issue was very important for me to address. Using a computer monitor for prolonged time can cause eye strain and can cause damage to your eyes, although this can take years. Using a monitor could not be avoided so to minimise the impact on me I will take regular breaks where I will look outside. I will also used darker backgrounds in AutoCAD, Rhinoceros and other software I will use to make it easier on my eyes. Evidence of this can be seen through the many screenshots of these software throughout this management folio.

RSI/KEYBOARD USE

My management folio will be full of text, all of which is going to need to be typed into the computer. Overuse and use in the wrong way of a computer keyboard can cause RSI (repetitive strain injury). The use of the school keyboard will not be as safe as my home keyboard. However I could not do much about the school computer. On my home keyboard there is plenty of support for my wrists and arms, this dramatically reduced the stain on my wrists. On the school computer however, there is not enough desk for wrist support (see figure 8.1), I will therefore limit the amount of typing I will do at school. A better solution would be to use an ergonomic keyboard where the keys are split to suit the natural human posture. However I cannot afford a new keyboard and through research I found that it was difficult to adapt to. As I am already comfortable and used to using my current keyboard I will not opt for this option.



Figure 8.1 - Wrist Space on School Computers



SEATING POSTURE

Having an incorrect seating posture can lead to adverse affects of my health, in particular my back. Through research I found that a computer chair with lower back support was best for my health in terms of ergonomics. So at home I will use a computer chair with lower back support (see figure 8.2). At school however the chairs are not very good in terms of ergonomics. Little back support is present (see figure 8.3). However I could not do anything about this.



Figure 8.2 - Home Computer Chair



Figure 8.3 - School Computer Chairs

POWER SUPPLY

When using computers, printers, laminators and other electrical devices I will ensure my safety by not overloading the power supply and not using electrical appliances that had been deemed unsafe by a certified electrician.



GENERAL ISSUES

PRINTER INK/TONER

When I need to change printer inks, toners and empty waste toner tanks, I will use latex gloves. I will do this because if I get ink on my hands it can take a while to get off, and it is not good for your skin. So to avoid it I will use the PPE of latex gloves.

LIFTING HEAVY OBJECTS

When I need to lift a heavy object, such as large stacks of paper, drawing boards, computers or printers, I will use the correct procedure of bending my knees and keeping my back straight, this will be done to avoid back injury.

FIRE EXTINGUISHER

As I will do a lot of work with electrical devices such as computers, plotters, laminators, etc. there is a high risk of fire. To help reduce the damage a fire would do, I will make myself aware of school procedure to follow in case a fire started. I will also make myself aware of where the fire extinguisher is located in the Graphics Lab (see figure 8.4), and what procedures to follow if it needs to be used.



Figure 8.4 - Fire Extinguisher in Graphics Lab

GENERAL FIRST AID

To ensure that I will receive and be able to give adequate care to anyone that might get injured in my work environment I located first aid equipment through the signage in my school (figure 8.5) and received training in administering first aid. I obtained a resuscitation certificate from this course. So if I or other in my work environment need assistance I can help out best I can whilst notifying the teacher.





Figure 8.5 - First Aid Sign



Figure 8.6 - First Aid kit in Graphics Lab



EVALUATION

GENERAL PROJECT EVALUATION

PROJECT EVALUATION IN RELATION TO STATEMENT OF INTENT

I feel that I have completed this project as I envisioned in my statement of intent. I feel that this management folio has fully achieved the level and breadth of documentation that I envisioned in my statement of intent. I feel that the standard of all my drawings are to the highest standard that I could achieve as I envisioned in my statement of intent. I feel that through the use of technical drawings, photorealistic rendered images and all the other miscellaneous production items that I produced have provided the detail and information needed to allow a manufacturer to produce the watch and for a marketer to market the watch.

DRAWING PRODUCTION

The drawings that I completed and the drawings that I planned on producing were different. This shows that I didn't plan the production component well enough, however it did not cause a problem. Originally I thought that I would produce an orthogonal drawing, isometric drawing, oblique drawing, etc. of the whole assembled watch. However very early on in the production I realised that I would need to do much more than this. I would need to produce the above and also a range of orthogonal drawings, isometric drawings, rendered drawings, etc. of different parts of the watch. This is reflected in my timeline plan, where I had listed the drawing types rather than the parts. In hind sight I see that I should have listed the parts of the watch and managed the time spent on each part, rather than the time spent of each drawing type. As I worked on one part at a time rather than one drawing type at a time.

3D CAD/CAM DESIGN SOFTWARE

As already mentioned I used Rhinoceros 4.0 BETA for creating my final design in 3D. The other three main tools that are currently on the market are PTC Pro/Desktop (Pro/Engineer), Solidworks and Autodesk Inventor (possibly CATIA and NX as well, I am not sure). One of the bad points about Rhinoceros and inversely good points about Solidworks, Inventor and Pro/Desktop is that it is not set up to deal with parts and part assemblies. However as I didn't focus on the mechanics and the insides of my watch this didn't matter much. I would have liked to use Solidworks but I could not afford it or spend the time to learn it. The modelling tools in Rhinoceros, at first glance, seemed easier to use than those in Inventor and Pro/Desktop. From samples that came with the software they all appear powerful tools. Tools which I would have liked to use, but in the end I only needed to design my watch in one of them and I chose the one with the least steepest learning curve, which was Rhinoceros. I did try and explore all of the above software, except Solidworks which I could not obtain.

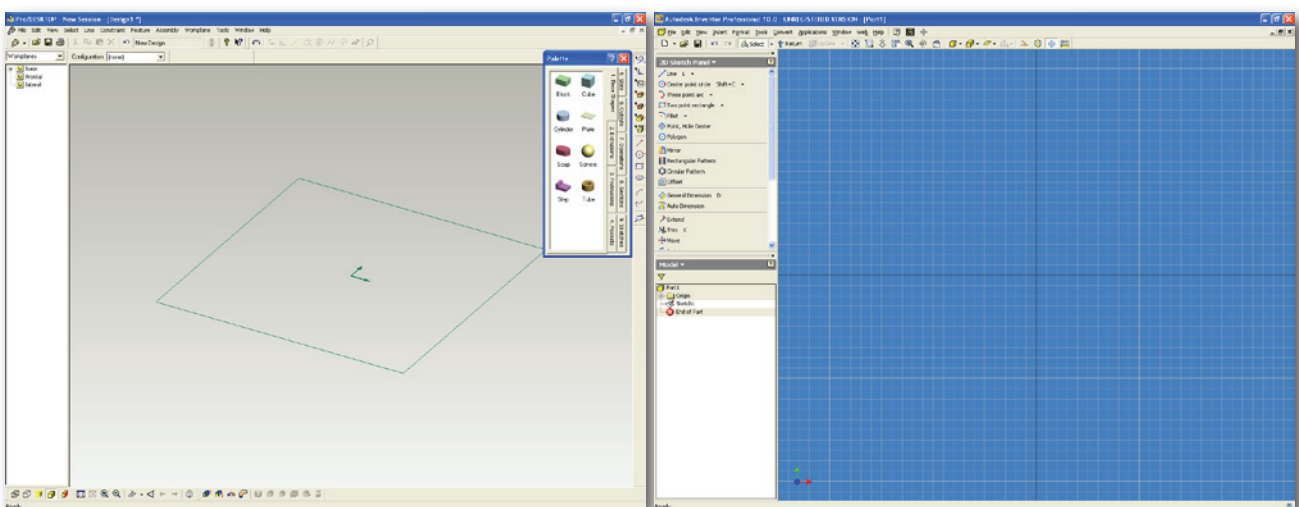


Figure 9.1 - Left: Pro/Desktop® 9.0, Right: Inventor® 10



IDEAS THAT NEVER MADE REALITY

There were things that I had planned to do however I never did them. Either because of cost, time or other reasons.

HOLOGRAM

I investigated this technology and found that a hologram could be produced from my 3D data. I found this information at www.zebraimaging.com

LESS TECHNICAL USER REFERENCE DRAWINGS

I would have like to produce less technical drawings that could be used in the watches user reference which show assembly drawings of how to go about doing thing like changing the battery, changing the time, adding and removing band links, adjusting the band length and clipping and unclipping the band from your wrist (although many of these are obvious and don't need to be explained). However as I am not a professional in Adobe Illustrator (these drawings could be made by drawing in Illustrator on top of technical drawings and renderings) and as I ran out of time to work on my project I never completed these drawings. Although on further inspection, these drawings could be created easily though a toon rendering engine. I had experimented with toon rendering as shown in Development of Ideas → Design Problems & Design Evaluation → Materials, page 37, but I never achieved results that I though were really good. Perhaps I if had the right software I could take my 3D model and render it and get drawings such as those shown below in figure 9.2 much easier, quicker and looking better.

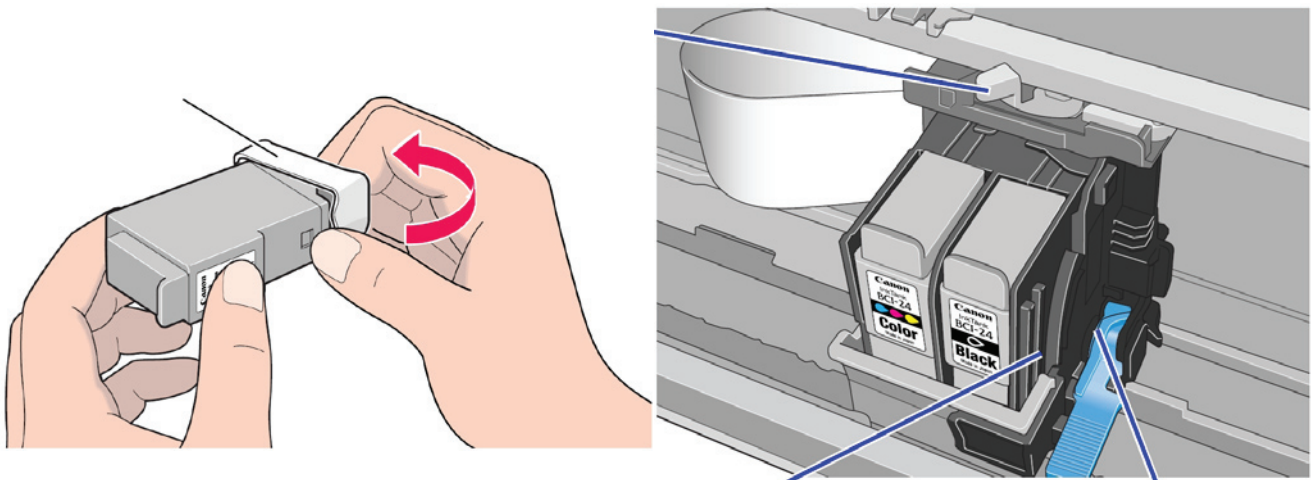


Figure 9.2 - Instructional Drawings using Toon Rendering
Image Source: Canon Bubble Jet Printer S200SPx User's Guide

REAL-TIME 3D VIEWING OF 3D MODEL

This was achieved but to a limited extent. There are many ways that this could be achieved, many technologies could be used, and I had many ideas. But I didn't have enough time to turn these ideas into reality. I was also limited by my budget and the resources and processing power I had available.

I partially achieved this by creating animations. Animations of the assembly and turntable animations. However these are fixed and not real-time. A user cannot interact with it. To give user interaction I used DWF and the free DWF viewer to creating real-time viewing of my 3D model of my watch design. However this was not as good as I hoped. It only had the static model and could not be exploded easily nor could moveable parts such as the clip and the band be interacted with. Also a simple default grey material was applied to the watch. So material information was not present. I could get a slightly better real-time 3D visualisation using Rhino, however unlike with DWF, the client or person viewing the model would need the rhinoceros software installed on their computer.

Another method that I envisioned was to render my watch in 3ds max from many different angles. ie. with the camera target set to 0,0,0 and the position of the camera moved around a sphere.



- 9. EVALUATION -

These separate images could be combined and software could be developed that allows for real-time rotation of the camera. This would result in a high quality image and still be responsive in real-time, because all the images have been pre-rendered. However I could not find any software that currently incorporated this, so if I were to do it I would have to write the code that automated the moving of the camera in 3ds max and then render the image automatically. This could be done with MAXScript. I would then need to write software that took these rendered images and somehow process the data so that it could be converted into one flat image. I would then need to write software that took this big image and allowed panning of it in a way that made it look the same as if the object is being rotated. This is a difficult task with lots of mathematical knowledge necessary. I did not have time to implement this. However this still has the limitations of the 3D DWF where the image is static and assemblies and explosions and mechanical part movements and workings and the simulation of the band and the joints of the band could not be shown.

If I did get this idea into reality then I could have made the user interaction easier, better and more realistic by using a spaceball. This would allow the user to rotate the object via the rotation of a real object where the rotation of the ball results in direct rotation of the watch on the computer.



Figure 9.3 - Spaceball

DYNAMIC SIMULATION OF BAND

When I created the band I just got the general shape then I got the computer to generate all the links. I never did any dynamic simulation of the band. That means that I never tested my design to see how flexible it was and how much and where could it move. As there was only a small gap between each link I suspect there is not much room for movement. However the links were thin and this means more room for flexibility. Another factor to consider here is the stress and strain in the pins. This is determined by how much force is applied to the band and where it is applied it is also affected by the shape and placement of the pins. This stress and strain is too difficult to work out manually but a computer can do it no sweat. However I did not have the software resources to actually analyse this. And therefore I do not know what type of material, heat treatment processes and what diameter to make the pins. For that reason I have not included any pins for the band in my working drawings.

RENDERING IN LESS COMMON PROJECTIONS

Every single commercial renderer that I know of cannot render 3D objects into images of oblique or planometric, even axonometric can be difficult in some. This not so much of a problem and not needed commonly because when you think about it, the most realistic drawing projection is perspective and one of the main reasons that rendering is done is to add realism to a drawing. So if you are trying to make a drawing look more real it seems logical to render the most realistic projection, which is perspective. None the less, in theory all drawing types can be rendered. And since I could not do it I thought that it would be a good place to investigate to help develop software that fills this gap and to show extensive knowledge and skills for this major project.

From the development of ViewIt, I had already derived the projection matrices that converted three



dimensional points into the projected two dimensional coordinates using a range of projections, including oblique and planometric. Now to render images in these projections I could do several things. One solution is to obtain the source code for an existing rendering engine (there many out there that are open source), and change the projection matrix used. However I am not an expert in rendering engines or Java, C#, C++ or any other higher level programming language that many of these renderers use. So for me it would be extremely difficult to change the projection matrix and then compile the whole rendering library again and use it. Also some other functionality of the renderer may rely on the fact that perspective projection is used and hence the rendering engine may not produce the required results if I just change this matrix one place of the code. However this method would not be able to be applied to the industry standard rendering engines such as mental ray or PRMan as they are proprietary software and do not provide their source code.

The second solution, which is probably the simplest for me to implement given my current knowledge and time limits, would be to use an isometric rendering and a Z-Depth map as well to work backwards. When you render an image each x and y, 2D point in the rendering region is assigned a colour. This rendering is only a 2D image and it is impossible to work backwards and get the 3D coordinates for each 2D pixel with just the final 2D rendered image. However if I render an image and also render a Z-Depth map, for each pixel on the screen I have its 3D coordinates and pixel colour information. So in theory I could voxelise my 3D space with coloured voxels. I could then apply the oblique or planometric (or any other projection) matrix, and generate a 2D rendered image in oblique, planometric or any other projection.

However it is not as simple as this. The first problem with this is that when the projection method is changed some of the voxels that were hidden from the camera before are now visible and hence there is no colour data for that voxel/pixel. This could be resolved by taking rendering and Z-Depth maps from different views, just as when 3D laser scanning an object you scan it from different angles to get the whole objects detail.

However this method relies heavily on the fact that a voxel will appear the same rendered colour in its associated pixel regardless of the viewing position. Now this may hold true for a simple rendered image. But for complex rendering techniques it may fail. I think that it should be true for global illumination, shadows and pixel shading, but I know that it is not true for rendering techniques such as ray tracing of environment reflection mapping, which is necessary for realistic rendering of metals and glass.

In practice the latter method described should produce a result that would look good, but be technically incorrect. I never implemented this solution due to the time restraints that I was under.

EVALUATION OF PRODUCTION COMPONENTS

This has been spread out throughout this folio, some of the other places that evaluation of the production components can be found is:

- Time management evaluation based on each production component can be found at Timeline plan → Evaluation of Components, page 75.
- Evaluation as to why I chose to complete certain components as part of my production can be found at Selection and Justification → Components, page 51.
- Evaluation of the design can be found throughout Development of Ideas, page 22.

However the evaluation of some of my production components can be found below. Drawing numbers are used for reference to the drawings. They can be found at the bottom right of each sheet of paper. If the item does not have this number then it is referenced here with words. This number is different to the item number on the small green label stuck to the front of my folio's and other items. Also I have not evaluated all the items, as many of them share the same evaluation. I have evaluated at least one of each type of production item.



DRAWING No. 54



This drawing was produced to fulfil the part of my statement of intent which describes how I intend to produce drawings to allow the watch to be marketed to the public. It is not technical and allows the public to see the watch in a way that hopefully will convince them to purchase it. It consists of an appealing background which proved a good backing that does not draw too much attention, the APOLLO product name and web site, my DD logo and a rendered perspective view of my final watch design. The rendered image came from my Rhinoceros model, I then imported it into 3ds Max 9 where I applied materials, lighted the scene and rendered the image. I am happy with the final composite image and I believe that it has, along with my other images of the same nature, fulfilled my statement of intent.

I made this image, which was compiled in Photoshop, to suit A2 paper at 150dpi or A1 at 72dpi or A3 at 300dpi. This item shows my skills related to the syllabus dot point *"students learn to: produce well composed and balanced drawings using colour and rendering appropriate to the needs of the projects produced"*. Technical details of the rendered watch are listed below.

Output File Format:	.tga (RGBA)
Rendering Application:	3ds Max 9.0
Rendering Engine:	mental ray 3.5
Rendering Duration:	10hrs 31min 14.28sec
Rendering Job Started:	28/02/2007 12:31:52 PM
Rendering Job Finished:	1/03/2007 1:03:40 AM
Rendering Resolution:	2480 x 3508

This final presentation rendering can be compared to one of my early presentation renderings shown in figure 9.4.



Figure 9.4 - Early presentation rendering



Figure 9.4 shows one of my early renderings. It is a lot different to my final renderings. This watch looks like plastic, which should not be the case, it should look metallic as it is made out of stainless steel. In my final drawings I have fixed this. Also I fixed the watch face in my final renderings and I made other changes as well.

DRAWING No. 75



This drawing was not created to act as a promotional tool for the watch. It lacks materials, however it is a different style to my other renderings and I think it looked good enough to be worthy of getting a logo and text and inclusion as a production item. Rendering details are below and it was composed in Adobe Photoshop CS2.

Output File Format:	.tga (RGBA)
Rendering Application:	3ds Max 6.0
Rendering Engine:	3ds max 6 Default Scanline
Rendering Duration:	41min 36sec
Rendering Job Started:	28/04/2007
Rendering Resolution:	3508 x 4961

PRODUCTION ITEM 76



This production item is more related to multimedia industries however the requirements state that we "may incorporate materials, processes and components drawn from other focus areas where appropriate", and this production item has allowed me to showcase certain components of my graphics project easier and in a visual way.

Output File Format:	.mov (Quicktime)
Rendering Duration:	59min 5sec
Output File Size:	145MB
Resolution:	1280 x 720 (HD)
Frame Rate:	30fps

I produced this multimedia presentation for me to show all my video together. This mean that all my animations could be shown in the one file instead of having many different ones. I also produced it to show evidence of a range of skills, evidence of a range of communication techniques and evidence of a range of computer applications.

I used the .mov Quicktime file format for my final video because it allowed for a small file size, great viewing quality and could run without lagging on a standard computer.

I added background music to the multimedia production to make the overall viewing experience better. Although due to the nature of the multimedia production I could not add many sound effects to increase the realism. I added a simple music track which I generated using WolframTones, <http://tones.wolfram.com/>, the home page of this web site is shown in figure 9.5.



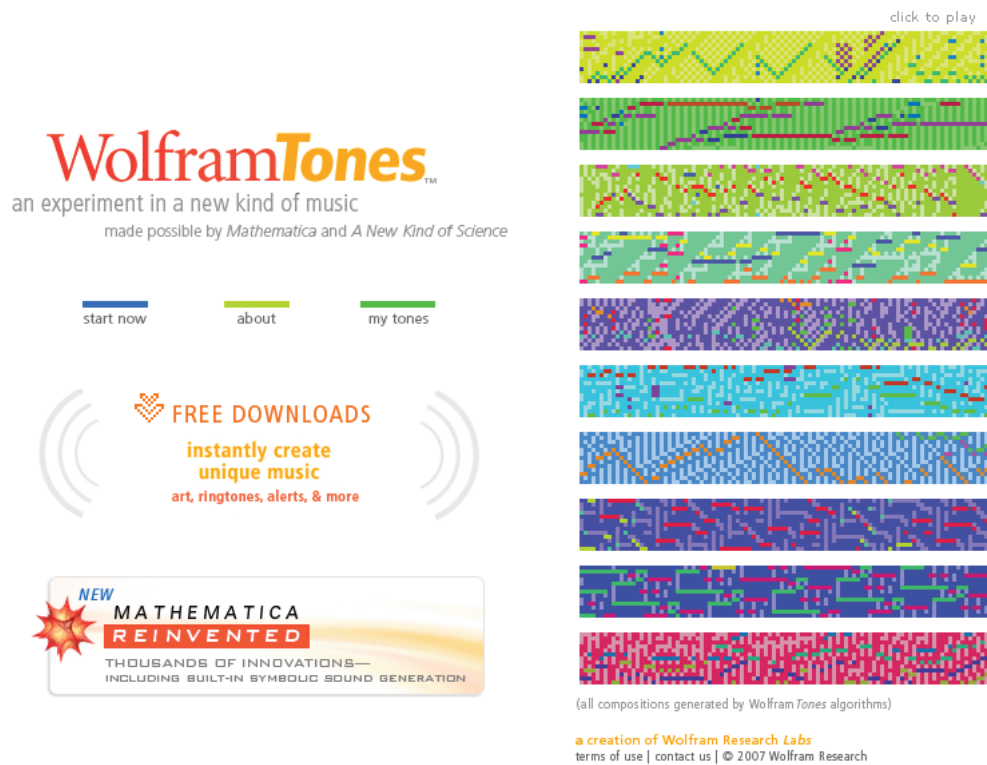


Figure 9.5 - WolframTones, used to generate the music for the multimedia production
<http://tones.wolfram.com/>

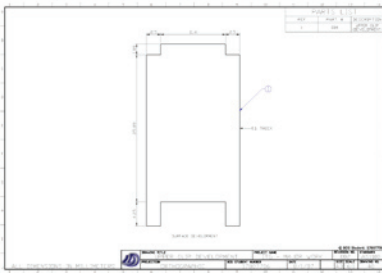
The basic storyboard of the animation is shown below in figure 9.6. The duration of each scene is shown below the scene picture.



Figure 9.6 - Multimedia Production Storyboard

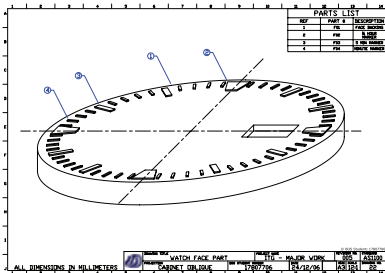


DRAWING No. 40



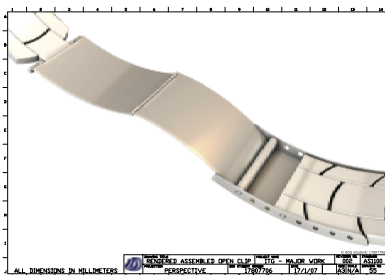
This drawing shows a development of a curved surface. As this part would be manufactured most likely by laser cutting the shape on a 2D sheet of metal and then bending it to shape, the developed surface is needed to be know and the dimensions that go with it. I modelled this part in its undeveloped state in Rhino, then taking advantage of the computing power and software capabilities I got the computer to calculate the developed surface.

DRAWING No. 22



This drawing was produced purely to show my range of skills, and use the oblique circles and oblique projection utilities that I had developed (I did this drawing before I developed ViewIt). The use of oblique in the drawings and of this orientation is not conventional, it is highly distorted and a combination of orthogonal, isometric and perspective views would have produced better results.

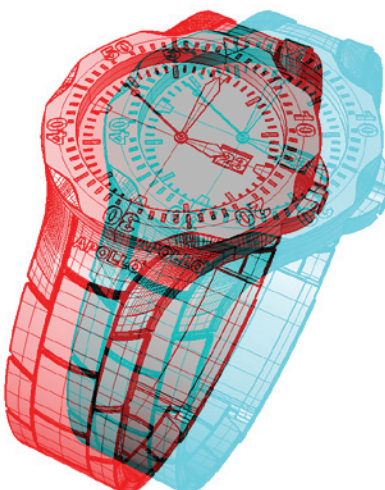
DRAWING No. 55



This drawings shows the clip rendered in an open position. As it was done to allow the manufacturer to easily see what the clip looked like open and as it is meant to show technical detail to the manufacturer I put the rendering on a title block and not with an artistic background.

Output File Format: .tga (RGBA)
 Rendering Application: 3ds Max 9.0
 Rendering Engine: mental ray 3.5
 Rendering Duration: 3h 45min 50sec
 Rendering Job Started: 11/01/2007 4:31:45 PM
 Rendering Job Finished: 11/01/2007 8:17:35 PM
 Rendering Resolution: 4961 x 3508

DRAWING No. 57, 58, 59...



These drawings, or more correctly visualisations, were produced to allow people to see the watch in real 3D. These drawings only took a few minutes to produce, compared with other items in which production times ranged from hours to weeks. I would have liked to have done these drawings using polarizing glasses and a monitor which polarizes the light. This way clear glasses are used and full colour can be accurately reproduced. It can also be viewed for hours without eye strain. However this technology requires expensive equipment that I did not have available.

Because a printer can only print a limited gamut of colour, the 3D images look better on computer than paper.



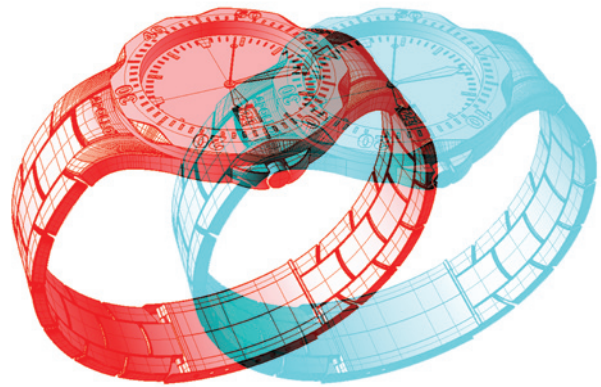
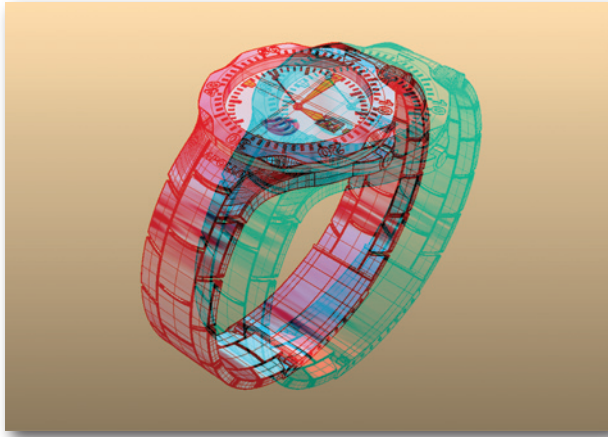
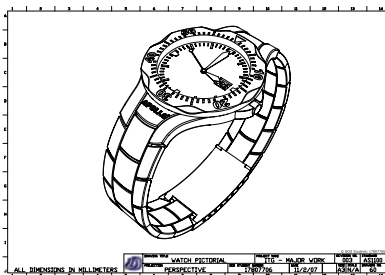


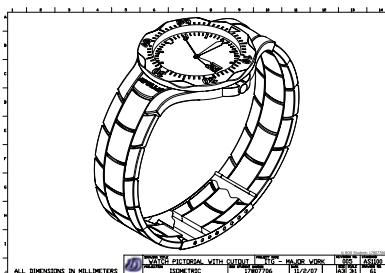
Figure 9.7 - Some of the 3D images

DRAWING No. 60



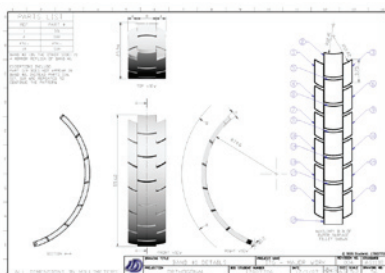
This drawing allows the watch to be seen as a whole in a realistic way, by using perspective distortion, but also in a technical way, by the use of lines, ie. not rendering it. One disadvantage is that the perspective distortion makes it hard for the viewer to determine relative sizes of certain parts, as the size changes depending upon where it is located.

DRAWING No. 61



In comparison with drawing no. 60, this drawing is useful as it shows accurately the relative and absolute sizes of the different parts. However as there is no perspective distortion it looks unrealistic. I removed part of the clip to show internal detail and to provide evidence that I can do removed part drawings and apply relevant standards.

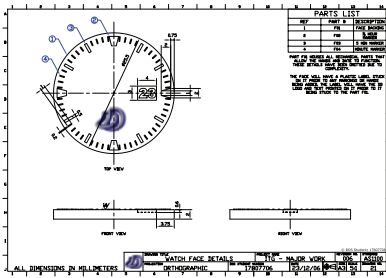
DRAWING No. 56



Producing useful technical drawings of the band proved difficult. I was unsure how to represent the band so I created an orthogonal drawing of the band in its default position and structure. I then produced an auxiliary view of the band. However it is not sufficient to manufacture the band by producing and extruding the band based on the auxiliary drawing. That would result in each link being flat, however as shown by the orthogonal drawing of the band in its default position, each link is slightly curved. I am not an expert in manufacturing processes, however the links in the band would most likely to be CNC milled, and individually then linked together with hidden pins. The data required for CNC milling is mesh data and would come straight from a computer. I have the data, however different CNC milling machines require the data in a certain format. My mesh data could be converted into the appropriate format and sent to a CNC milling machine, but I did not bother with this due to the range of possible formats. Drawing no. 56 would still be required to help show the general shape and to show assembly details.



DRAWING No. 21



One thing to mention about this drawing is the way the logo is placed on the watch face. As specified in the drawing notes the face will have a plastic sticker with the DD logo printed on the sticker placed on the face.

Also as the surface must be smooth to ensure the logo goes on best I have shown a surface finish on the drawing.

DRAWING No. 77



This drawing is a freehand sketch done on computer. It was done with the aid of a graphics tablet, Alias SketchBook Pro and Adobe Photoshop CS2. Using the computer allowed me to zoom in and work with tiny areas easily, it also allowed for multiple undo's and more sketching tools such as smudging, layers and many more. This was produced as a freehand presentation rendering which fits the time it took to produce. It took longer to produce this than all my other freehand sketches, but this can be considered my final design concept sketch and therefore should take more time. The pictorial development of this item is shown below.

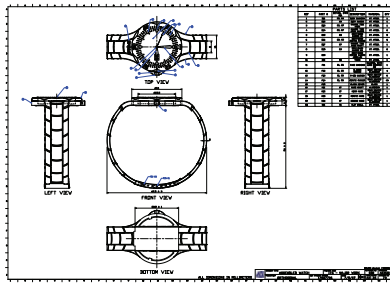




Figure 9.8 - Development of Drawing No. 77



DRAWING No. 73



This drawings was done for several reasons. One was to show that I can produce drawings for larger sheets of paper. The fact that I was only working on a computer screen made this hard. I found it hard to visualise how the final drawing would look like at full size on A1 paper. One of the advantages I notices about working on A1 paper is that you can put a lot of detail into a drawing, without it being so tiny on the final drawing that you cannot see the detail. I also applied relevant standards to the larger drawing size, such as different line weights, etc. Figure 9.9 shows how I aligned the views.

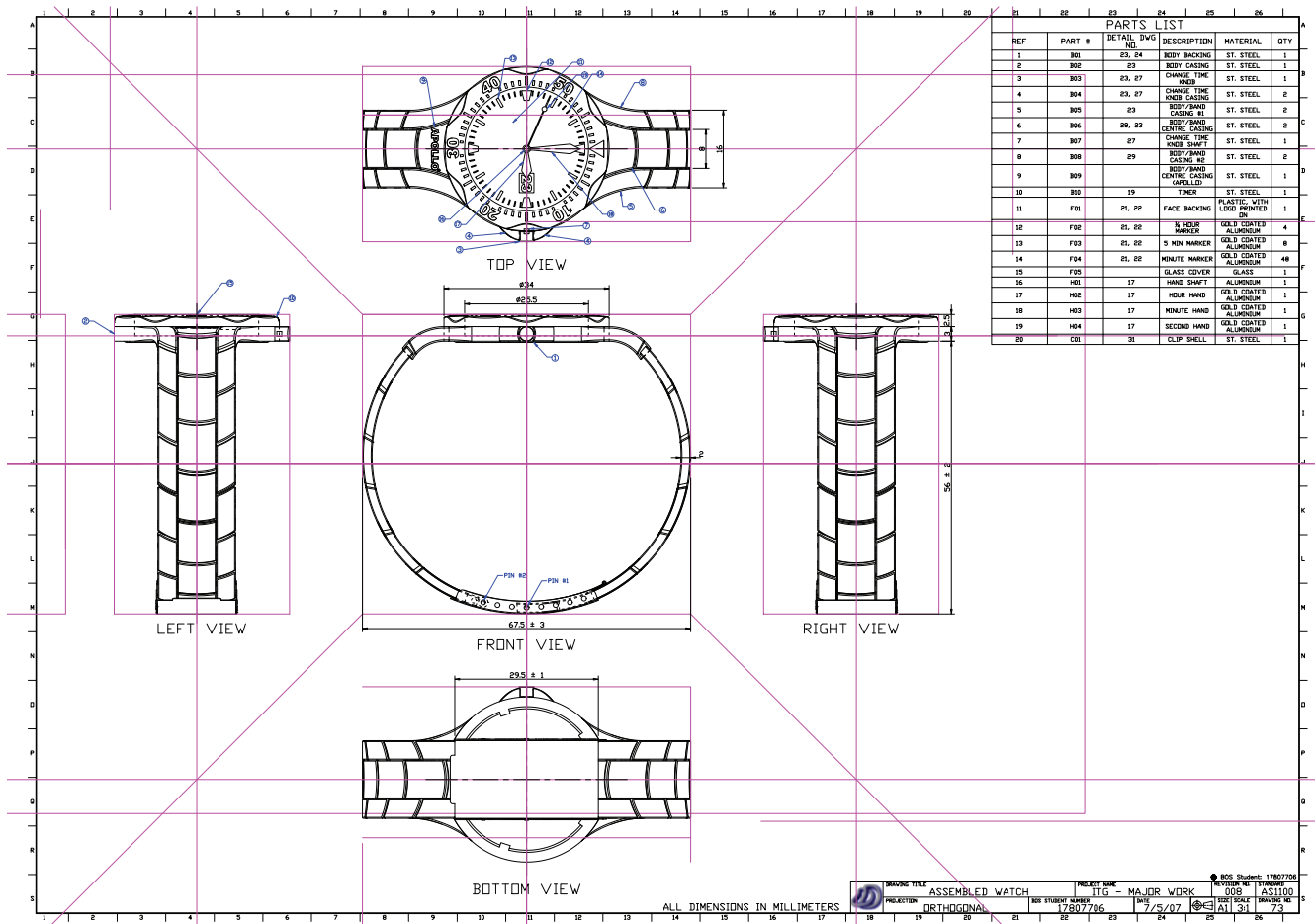
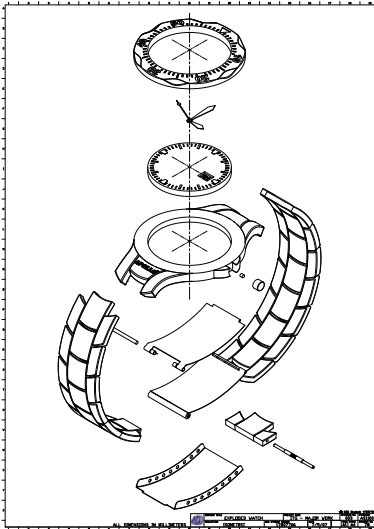


Figure 9.9 - Drawing No. 73 with construction lines shown

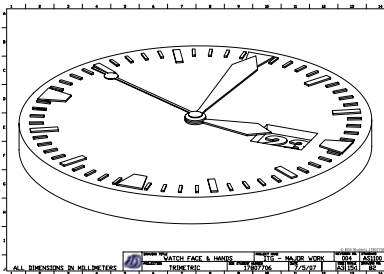


DRAWING No. 78



This drawing was produced on A1 paper in portrait orientation. It shows all the parts of the watch exploded. It was produced to help show how all the parts fitted together, and in order to get the most small geometric detail across clearly a large A1 size paper was needed.

DRAWING No. 82



This trimetric of the watch face and hands, was done primarily to show evidence of a range of skills (trimetric drawings) and to show examples of what kinds of things can be achieved with the use of ViewIt (see Appendix A - Developed Software → ViewIt, page 141). This drawing was based on raw data generated from my 3D model using ViewIt, however finishing touches and manual hidden line removal was necessary. Figure 9.10 below shows the raw data that came from ViewIt. As I had already created a drawing template, this cleaning and finalising process was made considerably easier. I choose to do the trimetric drawing of the watch face and hands because removing hidden lines and other touching up would take too much time manually so I just did part of the watch to show the principal that it can be done. I choose the face and hands also because they contain curves and still enough detail to show proof of concept.

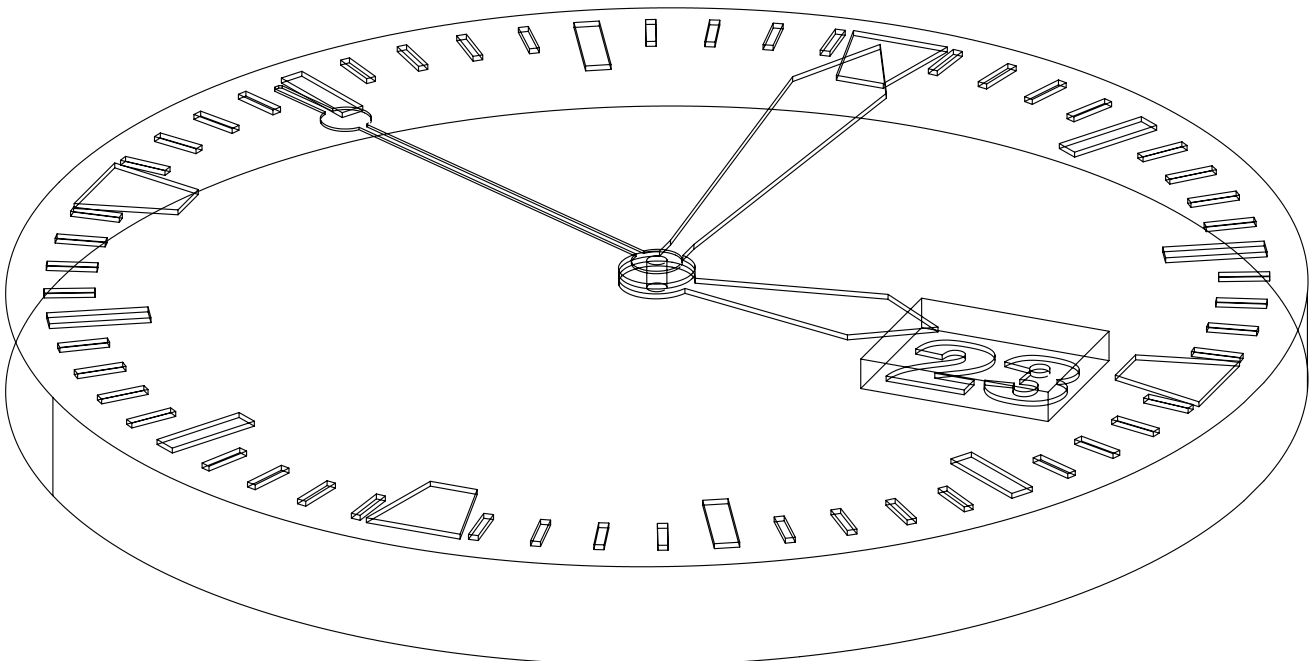
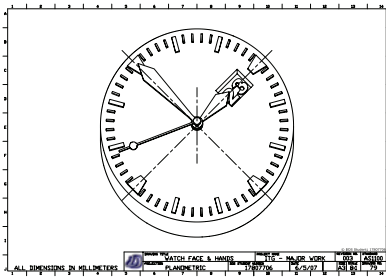


Figure 9.10 - Raw trimetric drawing data from ViewIt



DRAWING No. 79



This planometric drawing of the watch face and hands was produced for the same reasons as drawing no. 82 (see previous). The raw ViewIt data from which it was constructed is shown in figure 9.11 below.

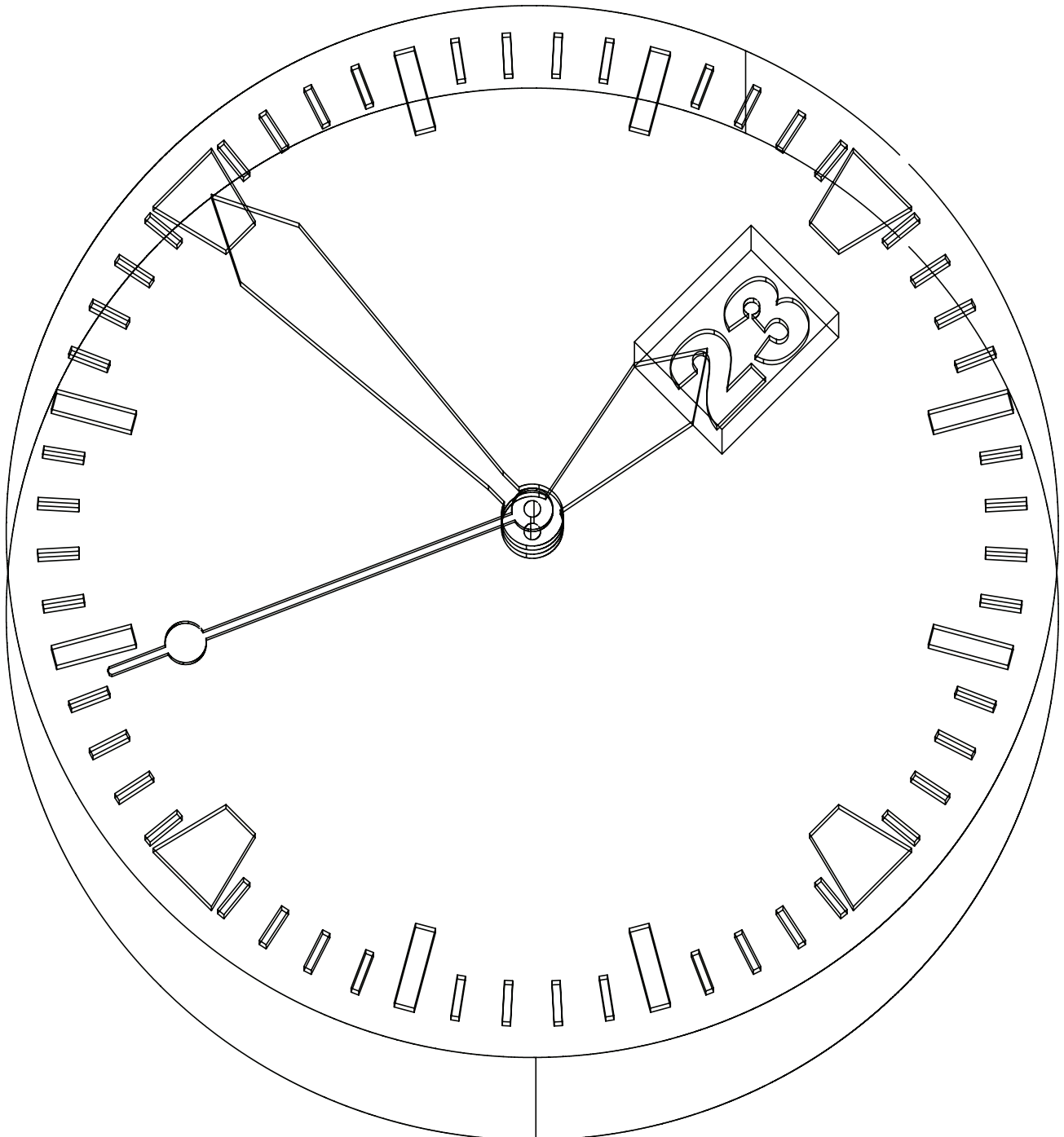


Figure 9.11 - Raw planometric drawing data from ViewIt



DRAWING No. 100



I never planned on doing this production item in my original timeline plan, however early on I did think that this type of thing would be really good to do. Unfortunately this task kept being delayed as I was working on other parts of the project. So when I was finally wrapping up my project instead of just scrapping this idea, I did

it in a very short time frame, an hour or two. Just to show the principal and that I had the idea and the knowledge to do it. I wish I had had more time so that I could have done a better job, and I am sure that with more time I could have done this better, however that just shows that I could have managed my time a bit better, but it also shows the huge amounts of time I have spent on this project.

The idea behind this type of drawing is to take a rendering of my watch and place that watch on a real life photo of a wrist in an environment. This will increase the realism of the rendering and give the watch context. It also allows people to see how big the watch is as they can compare it to the size of the wrist.

To get the image to look correct firstly the position of the camera in the computer rendering must match the position of the camera in the real life photograph. One way to do this is to, using the watch as the coordinate system orientation, measure the cartesian coordinates of the real life camera and use those coordinates to set the position of the camera in the computer model. Another is to use camera matching utilities in 3ds Max 9. This can be done with some reference points. As long as these points are the same position in the virtual world and physical world, then the computer can match the camera in 3ds Max to the photograph. The marker template I used is shown below in figure 9.12.

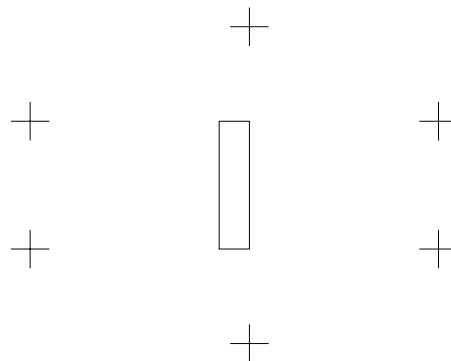


Figure 9.12 - Paper place underneath wrist to allow photo to be match in 3ds max

This is how I should have done it, however as I was running extremely short of time I didn't do this. I just used a rendering (figure 9.13) that I had already done and tried to visually take the photo (figure 9.14) so that the photo would match with the rendering. Another photo could be taken from the same position with no paper or hand (a tripod would assist this) to get the background information that could be used to erase the paper.





Figure 9.13 - Computer Rendering

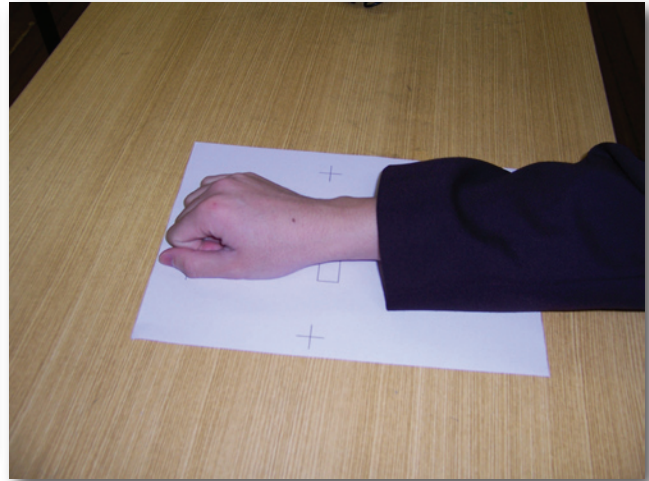


Figure 9.14 - Photograph

These two images were combined in Adobe Photoshop to give the image shown below in figure 9.15.

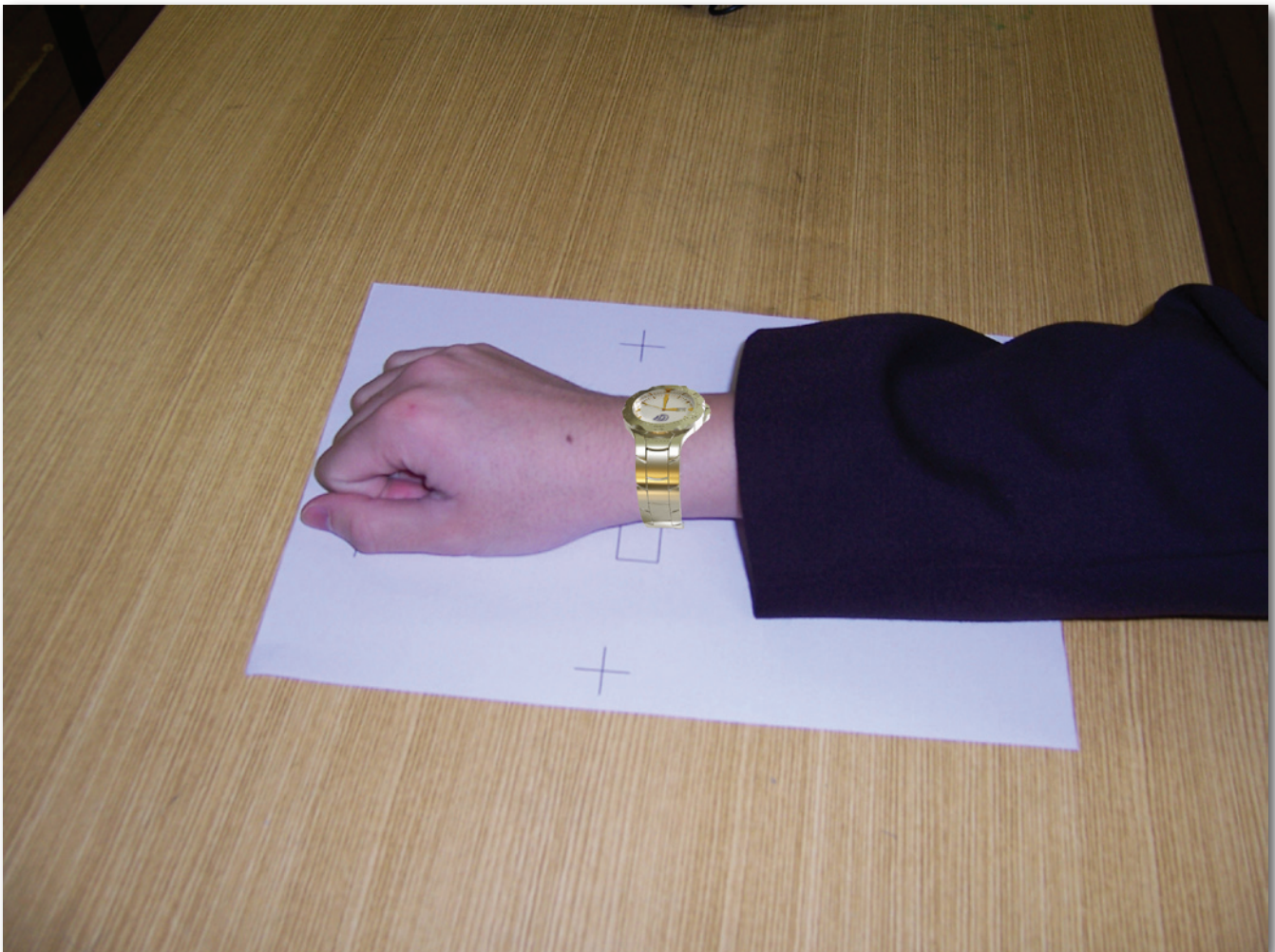


Figure 9.15 - Rendering on Photograph

However as I was rushing, the views are not exactly the same so it does look a little incorrect, also I had to manually go around the computer rendering and delete the background. I should have deleted this background before rendering and used the alpha channel to get just the watch.

If I had more time, then when using the digital camera I would have had the camera save the files in the cameras native raw format. This would allow for me to have more control over the image. In



- 9. EVALUATION -

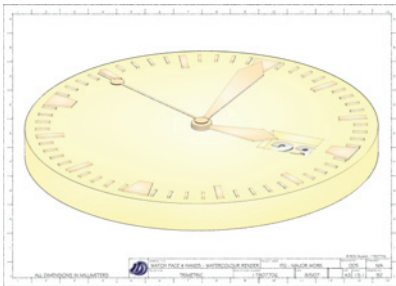
particular I could edit the exposure and other settings.

For better and more accurate and physically correct results I could take a light probe of same environment. This could be done by using an expensive specially build device or I could simply take a photo of a highly reflective spherical ball, such as a mirror ball. I could then create a HDRI from several of these environment photographs by taking several photographs with the camera set to different exposures and then I could combine the images to create a high dynamic range environment map with software such as HDRShop. This would mean that the environment of the photo would match that of the rendering.

Another thing that I could have done is use a hand and wrist on the computer and render that hand on the computer with the watch, just like what is done for CG movies.

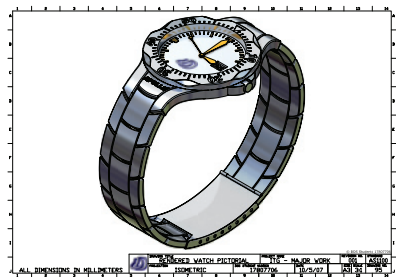
Again, I could have made this item much better if I had more time to work on it.

DRAWING No. 92



This drawing was produced using the line drawing from drawing no. 82, and processing it and adding colour in Autodesk Impression. Drawing No. 94 and Drawing No. 93 were also made using the same process.

DRAWING No. 95



This isometric rendered drawing shows that my renderings are not just limited to perspective or orthogonal. It shows my ability to render images in isometric. I could not however, easily render images in drawing projections such as oblique or planometric. See Evaluation → General Project Evaluation → Ideas That Never Made Reality → Rendering in Less Common Projections, page 90 for reasons and documentation of this idea/problem.

Creating this rendered image with the lines on it was made easier since it is in isometric. Meaning all I had to do was to create a 2D isometric drawing from my model in Rhino and then take the lines into AutoCAD, and then render the image in 3ds Max using parallel lines of sight and the camera at 50,50,50 with the target at 0,0,0. Then in AutoCAD I imported the rendered raster image, picked a point and aligned the two images and then I scaled the rendered image to match the line drawing and that's how I got the rendered isometric with the technical drawing overlaid.

If I was doing the same type of drawing (rendered and lines overlaid) but in perspective then I would need to ensure that the camera in AutoCAD and 3ds Max was looking and positioned at the same place and that the lens length and units of drawing were the same.



DRAWING No. 96



In my opinion this rendered drawing looks more realistic than my previous ones. This rendering shows the floor that the watch is sitting on, rather than an abstract background. It also shows the glass, which is not obvious on my previous rendered drawings. This rendered drawing shows the same watch and materials as the bulk of my previous renderings however a different environment map has been used.

Output File Format:	.tga (RGBA)
Raw Output File Size:	57.9 MB
Rendering Application:	3ds Max 9.0
Rendering Engine:	mental ray 3.5
Net Rendering Duration:	48h 15min 5.53sec
Rendering Job Started:	11/05/2007
Rendering Resolution:	4961 x 3508

I feel that this rendered drawing was an important part of my production, as for some reason that I do not know, prior to this rendered drawing I had been creating my renderings based upon a low resolution and low static range image. I had been using an environment map which was saved as PNG. A rendering created with this environment map is shown in figure 9.16.



Figure 9.16 - Rendering based on PNG environment map

I guess the reason that I used this environment map is because it creating an image that looked good. However I didn't realise that I could achieve much better. I had tried some high dynamic range images (HDRI) as environment maps, however I didn't like the results from that one environment map. So I ceased further exploration into other HDRI environment maps. Until this drawing, drawing no. 96, when I found that by using a HDRI as an environment map and adjusting the brightness, contrast and particularly exposure and gamma I could obtain much more photorealistic renderings.

I also realised, it was stupid of me not to realise the great importance and significance earlier, that the way a metallic material looks depends greatly upon the surrounding environment that it reflects, thus the environment map can change how good the rendered image looks. I also realised that this high dynamic range is vital.

The HDRI images I used were 32bit, this allowed for real life environments to be better recreated



- 9. EVALUATION -

in the computer.



Figure 9.17 - Same HDRI however different exposure shown

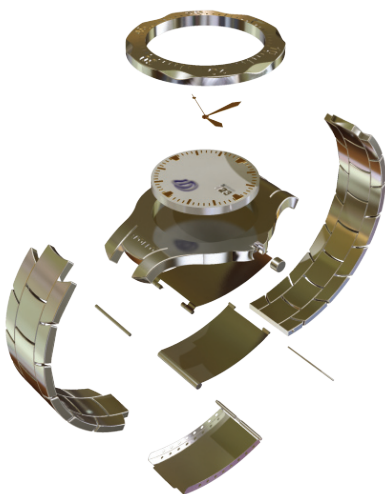
A normal 16bit image cannot hold all the information that is in a photo. Figure 9.17 shows two 16bit snapshots of a single 32bit HDRI. As you can see, particularly by the clouds in the window on the left image, a HDRI stores much greater detail in light and intensity, which is vital for producing photorealistic rendered images.

In addition to using HDR environment maps, I also rendered some of my images in HDR. Drawing No. 96 was one of them and the same rendering is shown in figure 9.18 below with different exposures.



Figure 9.18 - Drawing No. 96 at various exposures
(low quality due to time constraints)

DRAWING No. 97

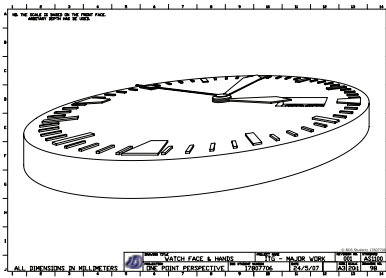


This is an animation showing the parts of the watch being exploded. The purpose of creating something like this is to show how all the parts fit together, as this can be difficult to see from an assembled orthogonal drawing. To speed up the rendering of this animation, I lowered the indirect illumination settings and the resolution. I also took advantage of network rendering. This animation was included in Production Item 76.

Output File Format:	.tga (RGBA)
Total Output File Size:	239 MB
Application:	3ds Max 9.0
Engine:	mental ray 3.5
Net Rendering Duration:	39h 59min 48.53sec
Job Started:	11/05/2007
Resolution:	800 x 600
Number of Frames:	330



DRAWING No. 98



This drawing shows my ability to create complex one point perspective drawings. Rhino could not generate one point perspective drawings, so I developed ViewIt which allowed me to generate 2D one point perspective drawings from my 3D model.

Some of the difficulties that arose are I didn't know where to place the vanishing point to achieve the best results. So I simply choose a few until I got a view that I liked. Some of the generated views are shown below in figure 9.19, 9.20 & 9.21. The raw output from ViewIt from which this drawing was created is shown in figure 9.22.

ViewIt from which this drawing was created is shown in figure 9.22.

Also this drawing has a scale of 20:1, due to the nature of one point perspective this scale can only be directly applied the the frontal face. Other dimensions can be calculated from my scale given enough information, however even with details of the VP and other it is still quite complicated to calculated the true size from the drawing measurements.

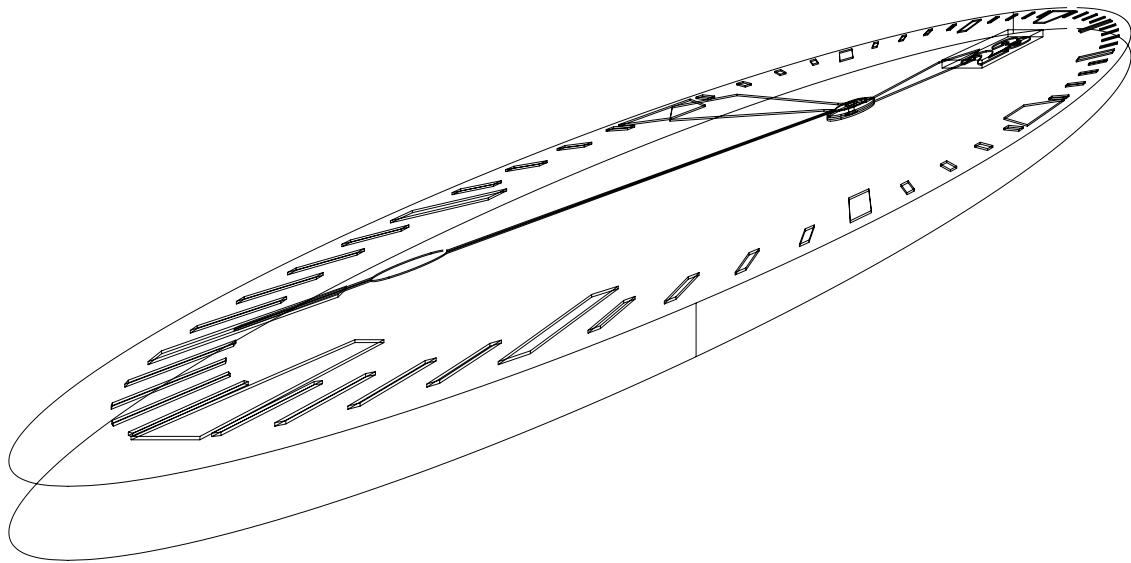


Figure 9.19 - Test 1, perspective effect was too great

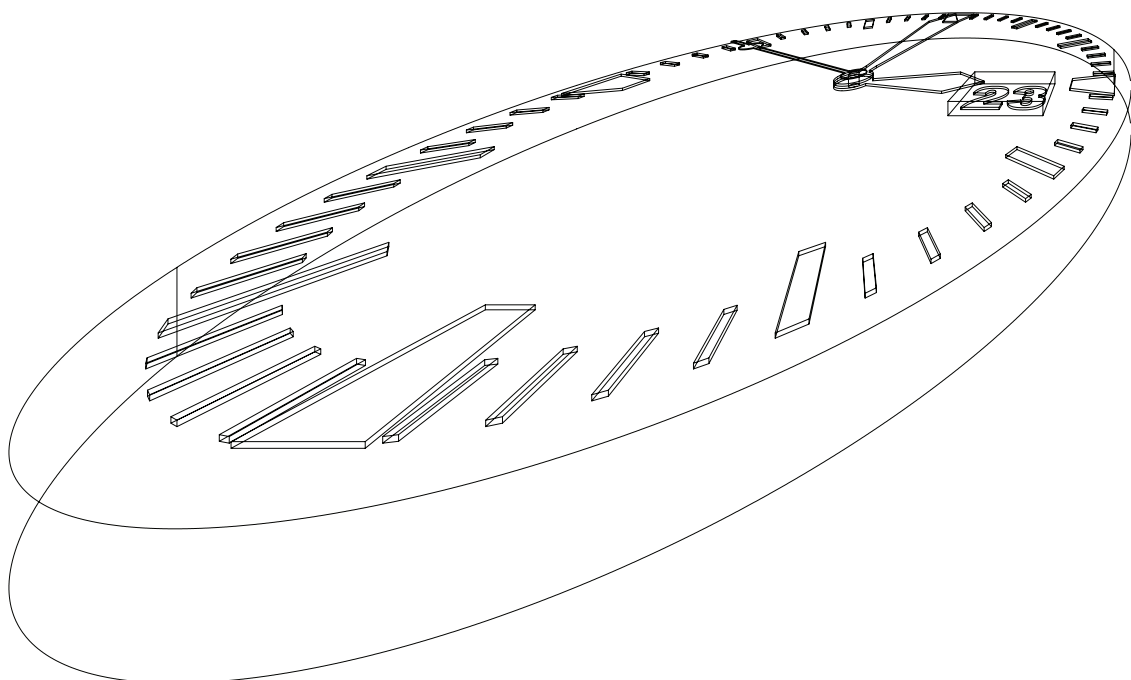


Figure 9.20 - Test 2, perspective effect was still too great



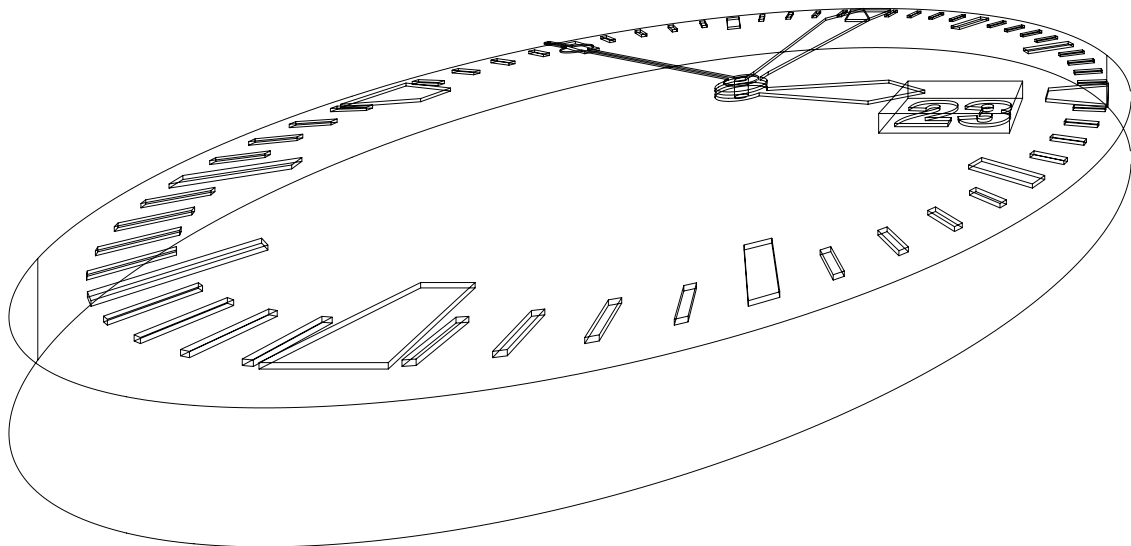


Figure 9.21 - Test 3, better but the persepective effect still looked too great

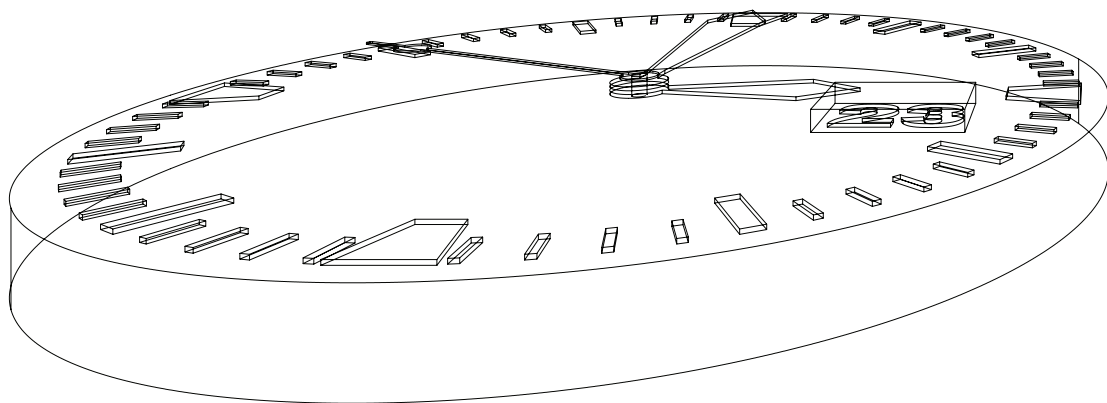
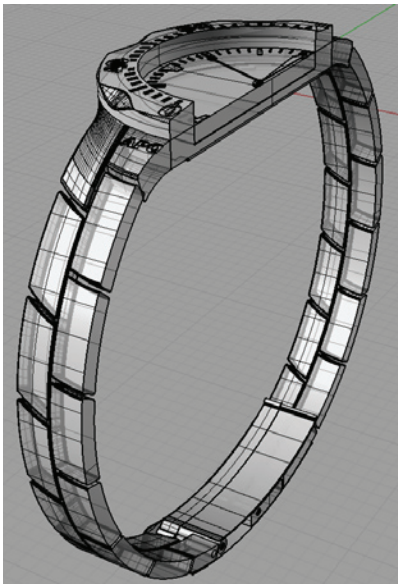


Figure 9.22 - Raw output from ViewIt, from which Drawing No. 98 is based

DRAWING No. 101



I got the idea for this drawing from the drawing shown in figure 9.23. However as I have not modelled much hidden detail, this drawing is not as important in terms of information conveyed.

If I had more time I could have made it look better by rendering it. Furthermore I could have done two renderings from the same view. One sectioned and one not. This way I could overlay the two in Photoshop and make the non-sectioned rendering a bit transparent. This would give the effect shown in figure 9.23. However I simply ran out of time to complete this.



(image unavailable)

Figure 9.23 - Cutaway Rendering
Source: Unknown (Not produced by 17807706)

PRODUCTION ITEM No. 102 (PHYSICAL MODEL)



This 3D physical model was produced as it was the only way for me to see exactly (without actual materials) what the watch will look like. It was very useful and an important production item for me mainly because I could also see the real size of all the parts. This proved important because it wasn't until I saw and could hold the physical model that I realised that many parts of my watch are too small, eg. the change time knob. The next best way to visualise the product is to use renderings pasted on real life photographs, which is shown in Drawing No. 100, page 102. I used 3D printing for the rapid prototype because it was the most accessible method to me. Other methods are explored in Selection and Justification → Processes → Production of Project → Rapid Prototyping, page 60. This item also shows evidence of modelling, a syllabus dot point.



PLEASE NOTE: THE PRODUCTION OF THIS ITEM WAS PARTIALLY OUTSOURCED TO A COMMERCIAL COMPANY. I PRODUCED THE .STL FILE OF THE MODEL WHICH WAS SENT TO THE COMPANY. THE COMPANY THEN PRINTED THE .STL FILE USING THEIR ZCORP 3D PRINTER. DETAILS OF THE COMPANY CAN BE FOUND IN THE REFERENCES SECTION, PAGE 220.

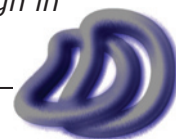
FLAWS IN PRODUCTION COMPONENTS/PRODUCT DESIGN

This is a list of all the things that you may not have noticed, but are wrong and should have been done differently. However as the watch will never actually be produced and due to time constraints, these flaws were never fixed. However to show that I know about them and to acknowledge them, they are listed below.

- Part B01, the back cover, should have a screw thread on the inside to allow it to screw on and off part B02, body casing. Also the body casing should have a matching thread to allow this.
- Part T01, the timer, and part B02, the body casing should have the joints to allow the timer to spin, however in my drawings the timer just sits on top of B02.
- Part F05, the glass cover is glued in place, this is probably not the best mechanical design, and it should have a better method of keeping it in place, also this results in it spinning with the timer, but it should stay still irrespective of the timer motion.
- The hands only have one shaft, this will result in them all rotation together. Three shafts should have been made inside each other to allow the hands to rotate independently of each other.

APPROPRIATENESS OF DESIGN AND/OR DESIGN MODIFICATION

Appropriateness of design and/or design modification and the evaluation of the final design in terms of design can be found in the Development of Ideas section, page 22.



COMMUNICATION TECHNIQUES

Communication techniques were an integral part of this management folio. Communication allows me to tell others my ideas and thoughts. Different communication techniques were used for different parts of the project and were used to communicate different things. They are described below.

REPORT STYLE TEXT

Report style text is seen in this folio. I used text, in specific report text type, because it allowed me to communicate what I did and why. Report text allowed me to inform the reader of any relevant information in a clear and formal style.

DESKTOP PUBLISHING/Non-FICTION PUBLICATION

By the use of titles, sub-titles, referenced figures, captions, footnotes, page numbers, contents, chapters, a title page/cover, references and an introduction I could present my folio in an acceptable and easy to understand format that is conventionally used in the world of desktop publishing. This technique was used to compile various other communication techniques that I used.

TECHNICAL DRAWINGS

Technical drawings were used to communicate technical details of my watch design and its manufacture. Use of these are evident in some parts of the production component.

FLOWCHARTS

Flowcharts were used in this folio. I used them to show the path I took in the production component. Programming flowcharts were used mildly in Appendix A.

TABLES

Tables were used to present data in a format that lists the properties and allows me to compare the data. I used tables to list information about the parts in my watch, both on a separate sheet and integrated in with my technical drawings. I also used tables in this management folio.

SPREADSHEETS

Spreadsheets were used in a similar way to tables, but to present the data in a more formatted and logical way. Using spreadsheets I could perform calculations on the data and analyse the data in a more mathematical way.

GANNT CHARTS

Gantt charts are a subset of spreadsheets. I combined graphics along with the data and data processing that is made available by using spreadsheets. Gantt charts were used to plan my time on a weekly scale.

PHOTOGRAPHS

Photographs were used to show environments and objects in the real world, in particular to show me working in safe environments which I used for the OH&S section.

FREEHAND SKETCHES

Freehand sketches were used to communicate my design ideas, to others and myself. I used annotations in these sketches to communicate design ideas more clearly.

MULTIMEDIA VIDEO

This is shown by the video in the production component. I was hoping to make an interactive multimedia presentation of the watch which allowed users to interactively select parts of the watch and get information (technical and non-technical) about that part. I was hoping to do this in Flash, however I ran out of time to complete this.



MISCELLANEOUS RENDERINGS

These renderings were not included in the production component as they do not show the actual materials that the watch is made from. However they do illustrate the power of rendering and rendering in different forms.

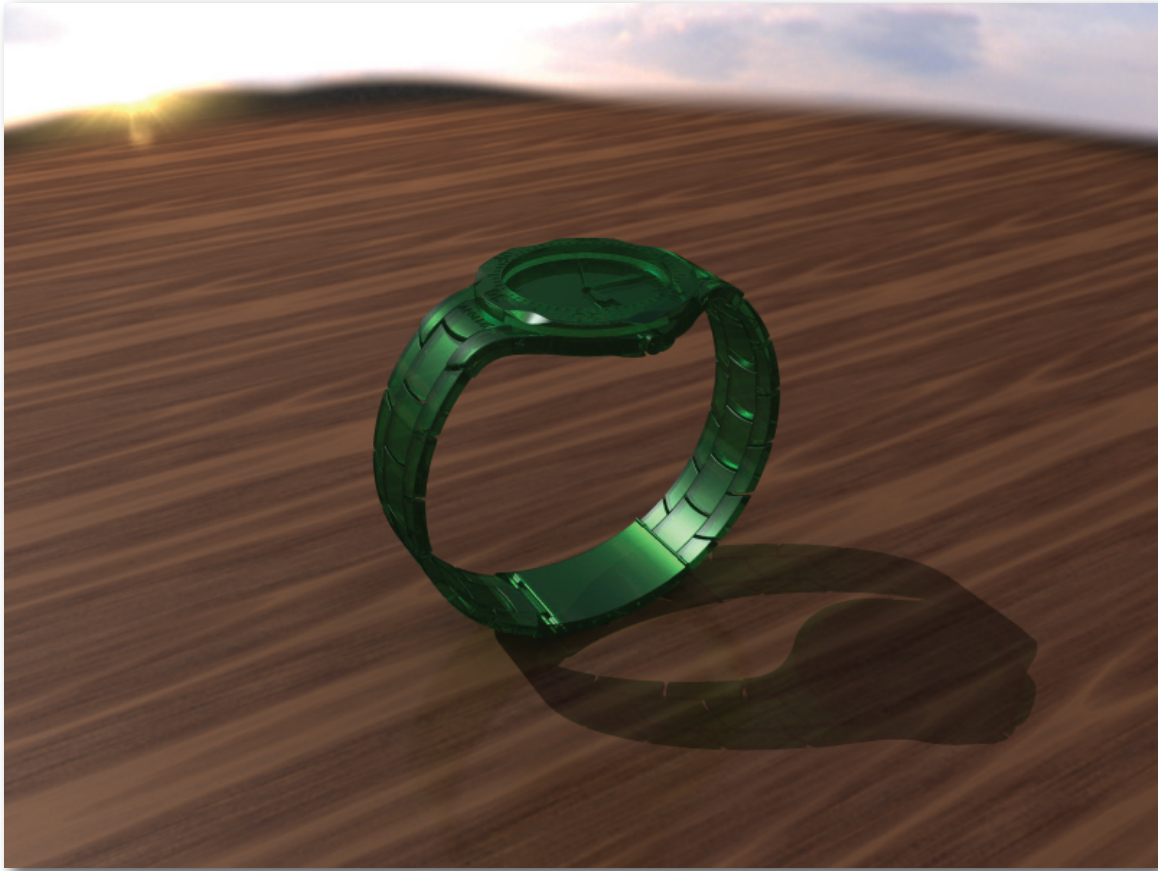


Figure 9.24 - Rendering showing enviroment, rendered in Image Studio 3.0





Figure 9.25 - Rendering showing watch carved from wood, rendered in Image Studio 3.0



Figure 9.26 - Rendering showing watch as clear red plastic, rendered in Image Studio 3.0



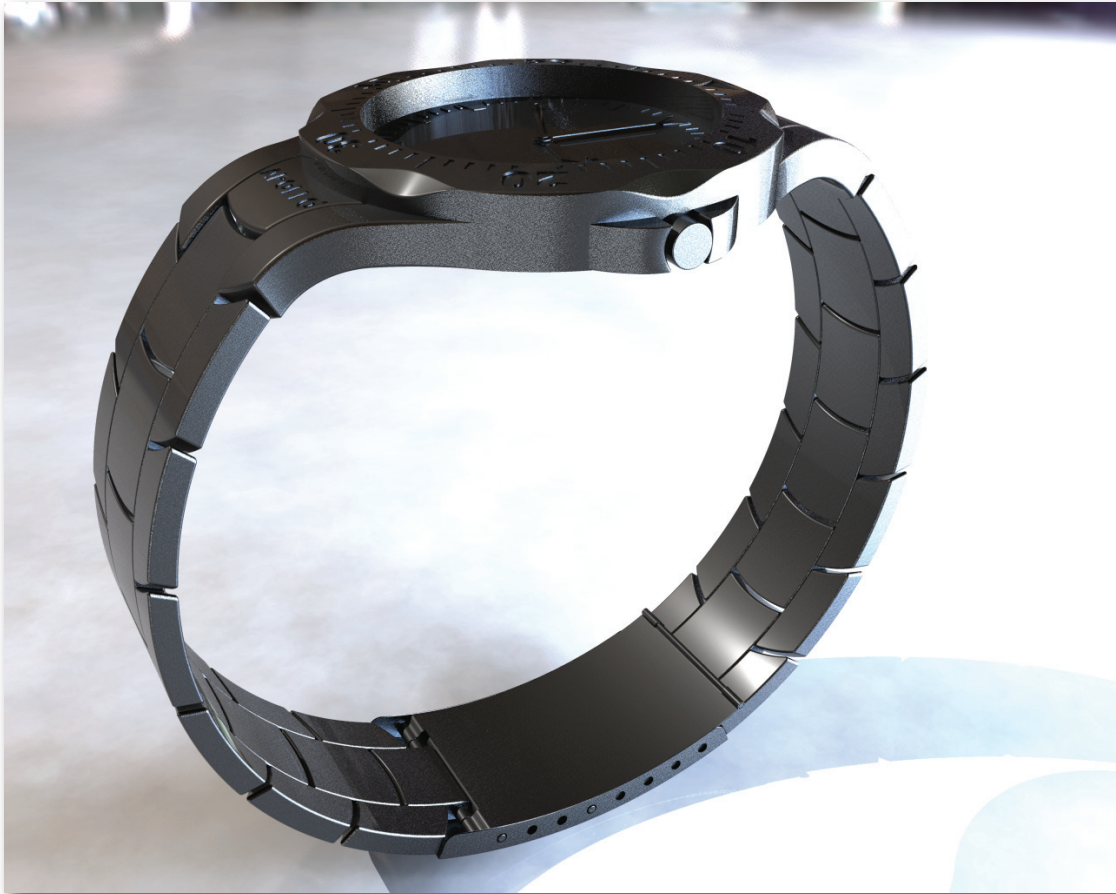


Figure 9.27 - Rendering shows a more realistic background and floor but not realistic metal, rendered in Image Studio 3.0



Figure 9.28 - Rendering shows how adding a real photo background and image based lighting can increase the realism of a rendering, rendered in Image Studio 3.0





Figure 9.29 - Rendering, rendered in Image Studio 3.0





Figure 9.30 - Photorealistic Rendering (3ds Max 9: metal ray, using 'room day' environment map)

EVALUATION OF MANAGEMENT FOLIO

The management folio proved an integral part of my project. It allowed me to document the project, evaluate it and give an insight into how I did things and why, and showcase the communication skills needed that were not addressed in the production component.

ONGOING EVALUATION

As ongoing evaluation is ongoing it can be found in almost every other part of this folio. However I basically have summarised all that information here, to give a diary/record of procedures that is not specified in the marking guidelines. The exact time when I completed the tasks mentioned below can be found in the timeline plan.



- 9. EVALUATION -

Term 4 - 2006

Week 1:

- I began by drafting my statement of intent, and then refining it.
- I also designed the DD logo, which I needed to start the rest of the project as it would appear on everything.

Week 2:

- Next I started working on my timeline plan. I devised a proposed scheme, and throughout the project I checked back with my original proposed plan to make sure I was processing as planned.

Week 3:

- Next I mapped out my proposed finance plan.
- I now started my research. I collected ideas and information as to the design of the wrist watch that I am producing drawings of. I also collected information that would help me produce my folio and production component (not related to the design of the watch).

Week 4-8:

- I took a break for a couple of weeks. I was not behind my proposed timeline plan so I didn't need to work for a few weeks.

Week 9-10:

- When the school term got close to ending I found more time to work on my project. Using my design research I started to develop my design ideas. I made annotated sketches of my ideas.
- I now had a better idea of what extent my production would be. So I documented and justified the materials, processes, components, and other resources that I used and would be using.

Summer Holidays

Week 1-3:

- Next I refined my watch design, and began modelling the design in a 3D CAD program (Rhino). During the modelling stage I encountered many problems, these were resolved and documented (see Solutions to Problems in Production, page 130). I also began development of my solution to one of my problems, this led to the development of ViewIt (see Appendix A - Developed Software → ViewIt, page 141)
- Once I finished certain components of my design in 3D in Rhino, I developed working drawings, of these components.

Week 4-5:

- Once I had done most of my working drawings, and I had finalised my 3D model design. I exported the model into 3ds Max and began applying materials and doing renderings to see what the different materials made the watch look like. Once I had finalised the materials, I created high quality presentation renderings.

Term 1 - 2007

Week 1-5:

- Once I done the majority of working drawings, rendered drawings, I worked on other production components that are less important. I also worked on my folio, to make it contain more information, and in a more correct format.



Week 6-10:

- I did no work on my project. I spent this time studying for my half yearly exams.

Term 2 - 2007

Week H1-7:

- As the due date of the project became closer, and as I started finishing the bulk of my project I self evaluated it and checked it for errors of possible improvements. I also started bringing it all together, I was trying to get as much finished as possible. After doing that I got others to read it and view it and give me advice and suggestions.

EVALUATION FROM INDUSTRY

(image unavailable)

Figure 9.31 - Evaluation of Drawings from Ms Graphics Industry, IND-TECH



EVIDENCE OF INDUSTRIAL PROCESSES/TECHNOLOGIES

Evidence of industrial processes and technologies can also be seen in the production component, however it is not always obvious what process or technology I use, so a brief explanation is included here. Industrial processes and technologies can also be seen in Selection and Justification → Processes, page 55 and Selection and Justification → Resources, page 62, however that has few references and examples of where that process and/or technology was used, so again evidence is here. However it would be impossible for me to document all the processes and technologies I used, so I have tried to incorporate many without listing all the several hundred.

The following screenshots show interim phases of documentation, planing and production. As they showcase the interim phases of production, many of the things that can be seen in the screenshots were never included as part of the final production as I improved on them at a later stage. A short description is also below each image.

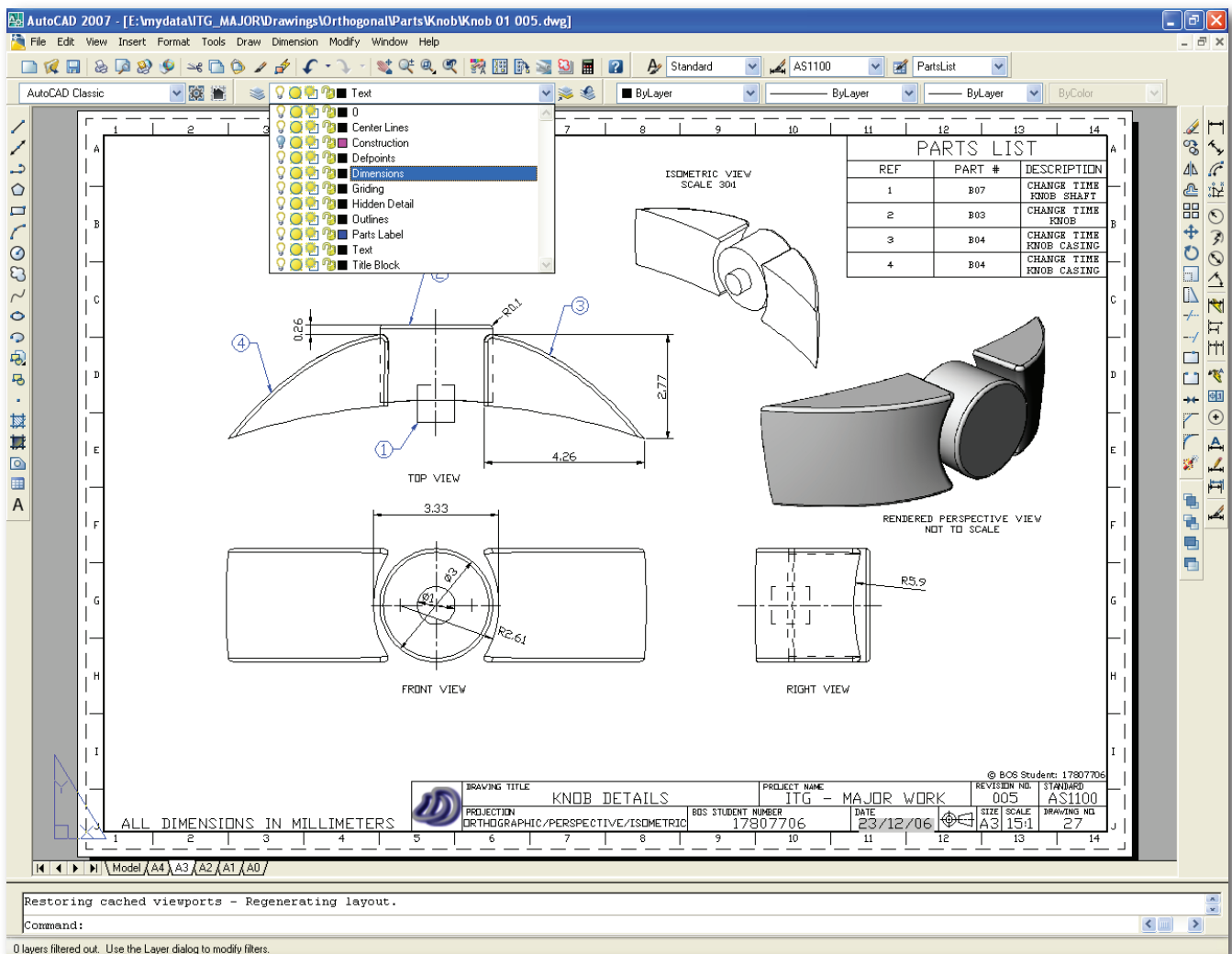


Figure 10.1 - Development of CAD Drawings

From raw 2D data generated from my 3D model (some of which was generated using ViewIt, see Appendix A → ViewIt, page 141), in AutoCAD I fixed any artifacts or things that were wrong, checked hidden detail, added centre lines, added dimensions, added parts list and combined multiple drawings on the one layout.



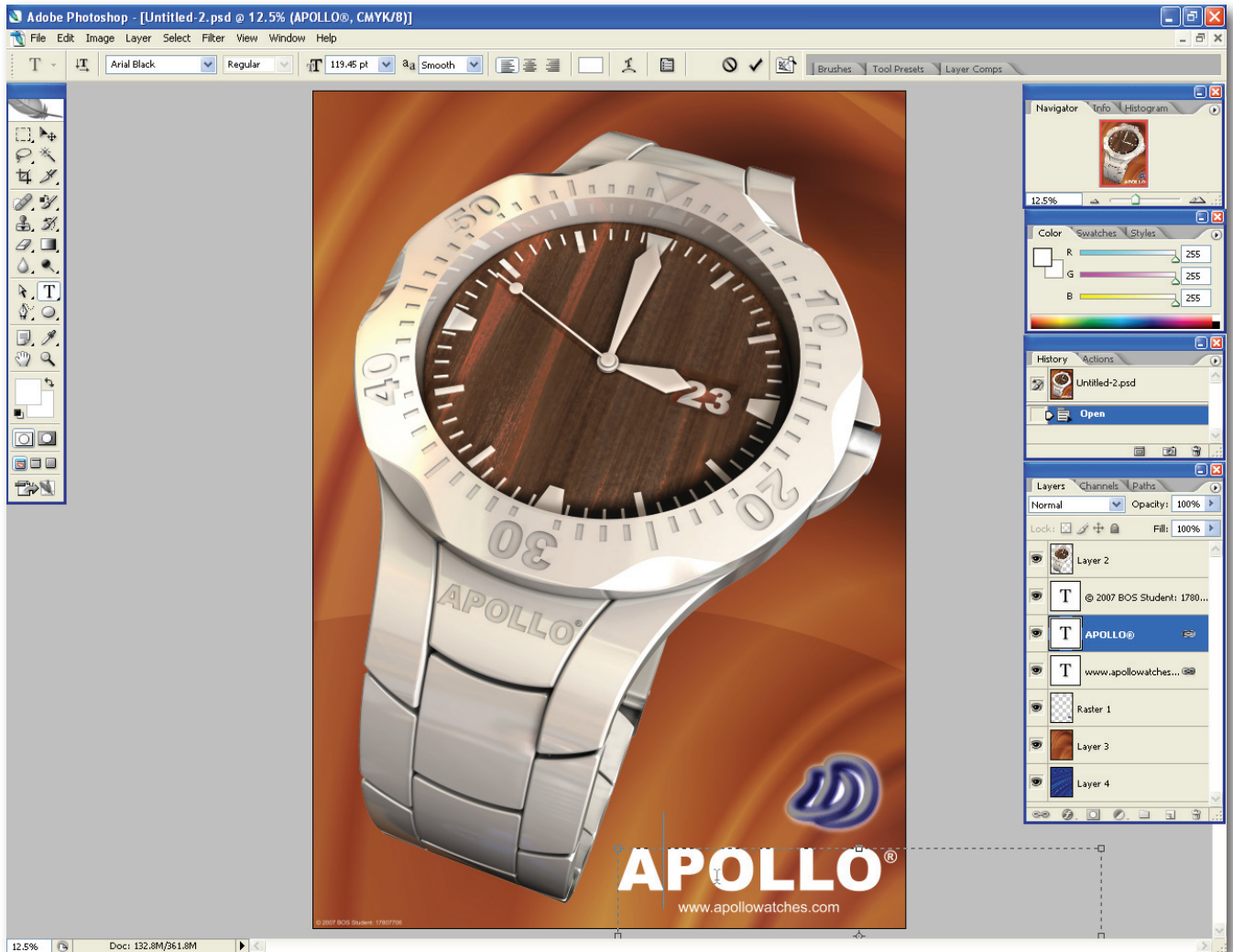


Figure 10.2 - Development of Marketing Images and Presentation Rendering Composites

Once I had rendered some images of my watch, I compiled them with a background, text and logo to make a presentation rendering. I used layers and transparency along with modifying colours to get a well presented poster which showcased composition, balance and colour. These type of drawings fulfilled the marketing part of my statement of intent.

I had to be aware of and utilise the different colour systems including RGB and CMYK. I also used exposure controls along with gamma, brightness and contrast functions. I also used 32bit images for transparency or high dynamic range images.



- 10. EVIDENCE OF INDUSTRIAL PROCESSES/TECHNOLOGIES -

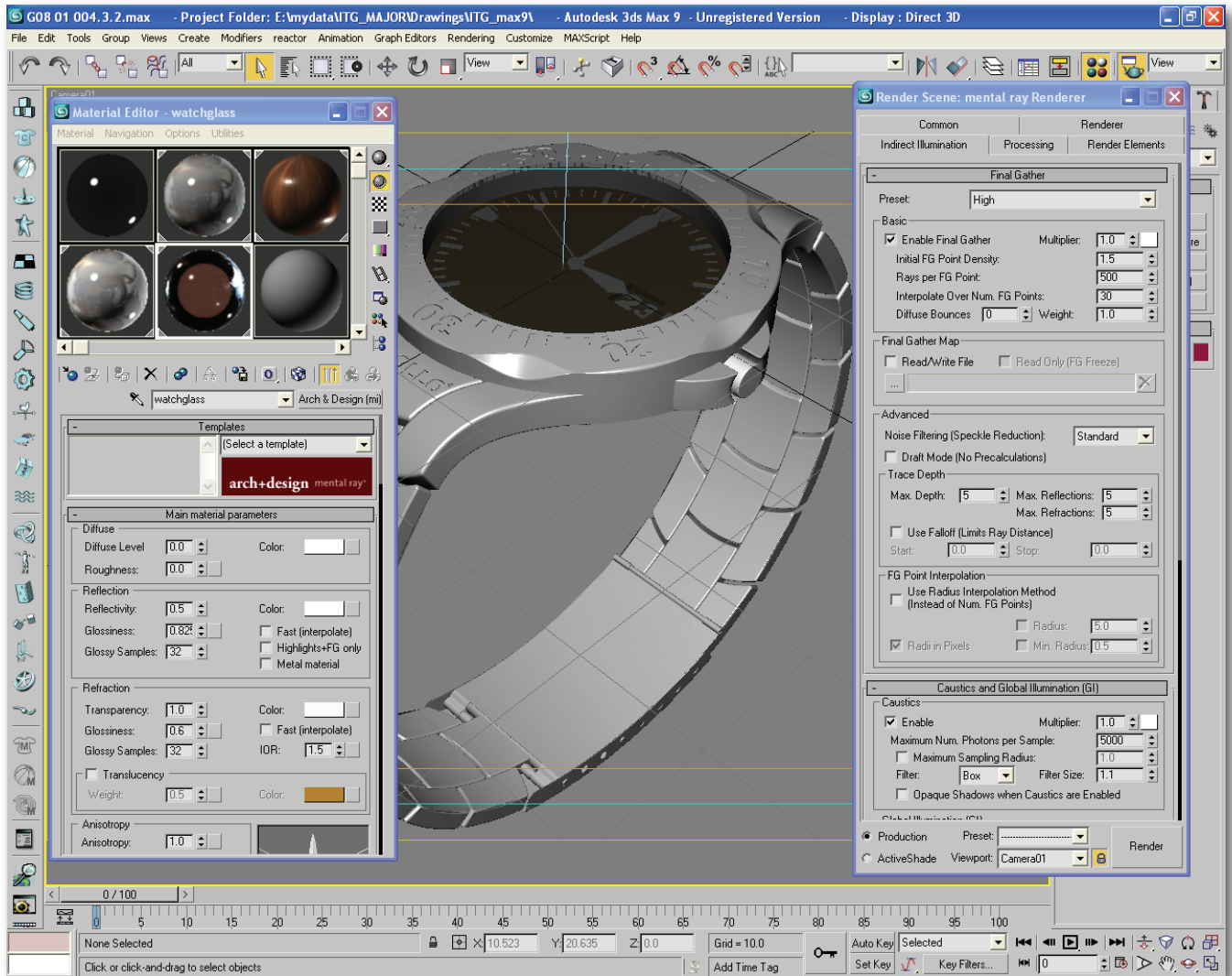


Figure 10.3 - Development of Photorealistic Renderings

I took my 3D model from Rhinoceros and imported it into Autodesk 3ds Max 9. Here I applied materials, as seen in the material editor (left side) in which I edited things like texture maps, diffuse level and colour (colour from object), reflectivity and reflection (for metals, plastics, glass, etc.), and environment maps (these made the metals look real, without the environment maps the metal would have very little to show as a reflection, and it is these reflections which make the object look realistic). I could also render the images, the settings can be seen in the mental ray Rendered (right side), where I could change the settings depending on what I was going to use the images for. For test images I set most settings at low levels, including the final image resolution. For high quality final images, I had to make these settings slightly higher, however the higher the levels the longer the image took to render. I set things such as final gather, which gives bleeding and reflection of light effects, I also edited caustic settings for glass, lastly sampling quality and image resolution was notched up for my final renders to remove artifacts and make the image look better.

I made use of the animation capabilities of the software for generating exploded animations and animations of the watch rotating.



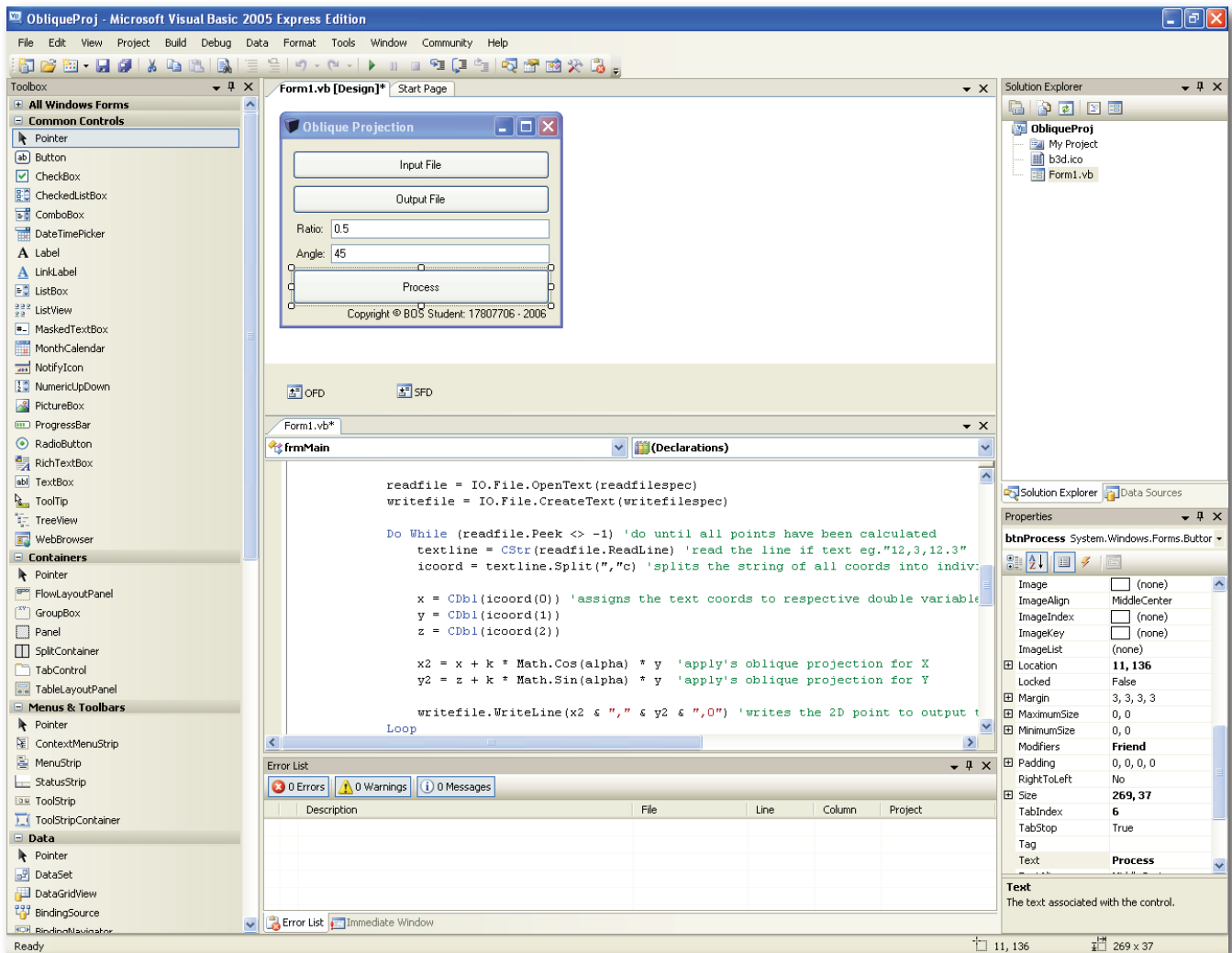


Figure 10.4 - Development of Oblique Projection Tool

Here you can see how I developed the Oblique Projection Tool which was later expanded into *ViewIt*. Basically, I started with a bank form, and then added controls to it such as buttons, text box's, etc. I then coded the code to make the program work. I took advantage of debugging tools when I had bug's in my program that I couldn't readily understand or fix.



- 10. EVIDENCE OF INDUSTRIAL PROCESSES/TECHNOLOGIES -

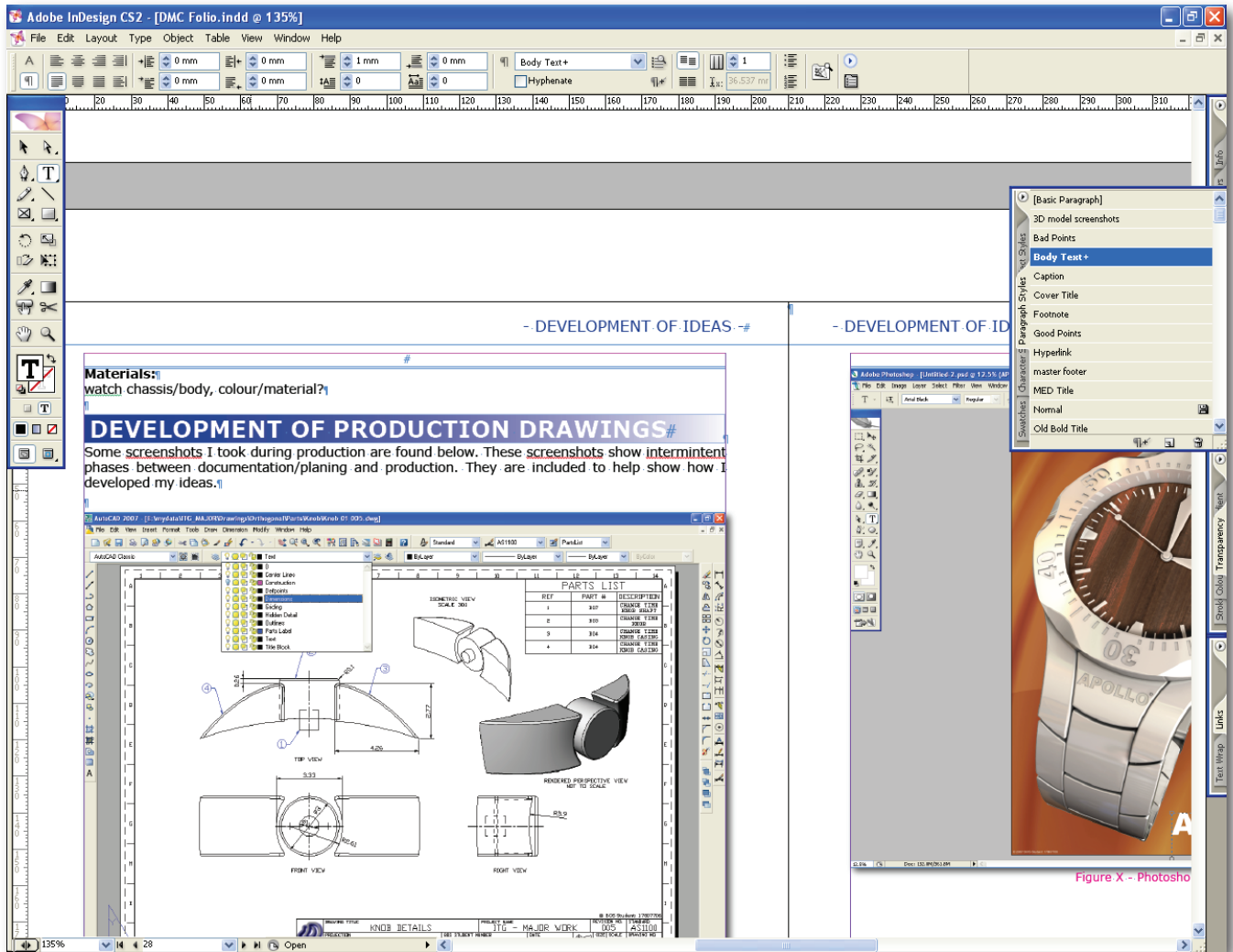


Figure 10.5 - Development of Management Folio

Here is the interface of the documentation component. I added images, formatted text and graphics to make a document that looked professional and was easy to follow.



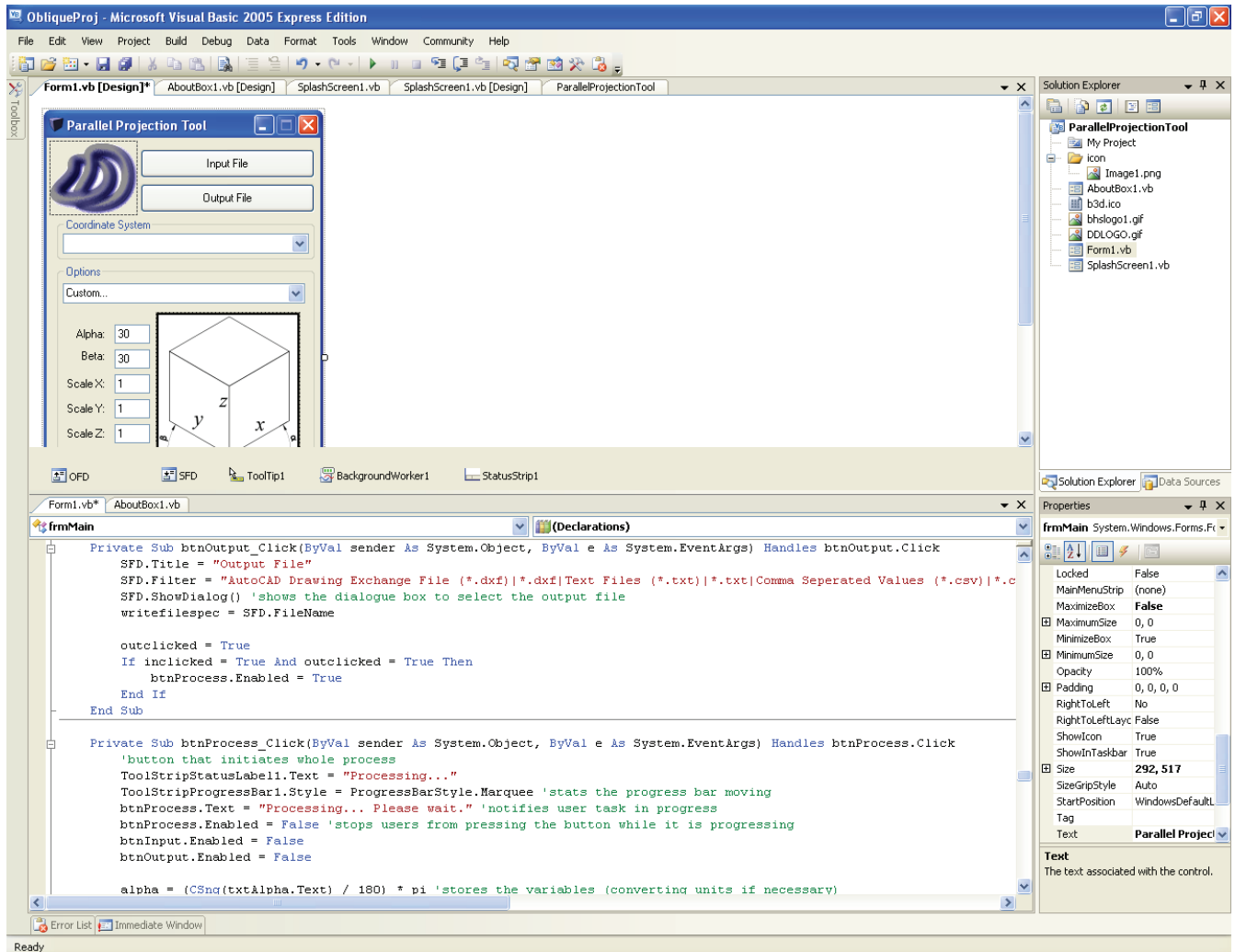


Figure 10.6 - Development of ViewIt

Here you can see the development of *ViewIt* which was based on the Oblique Projection Tool. More details about *ViewIt* can be found in Appendix A → *ViewIt*, page 141.



- 10. EVIDENCE OF INDUSTRIAL PROCESSES/TECHNOLOGIES -

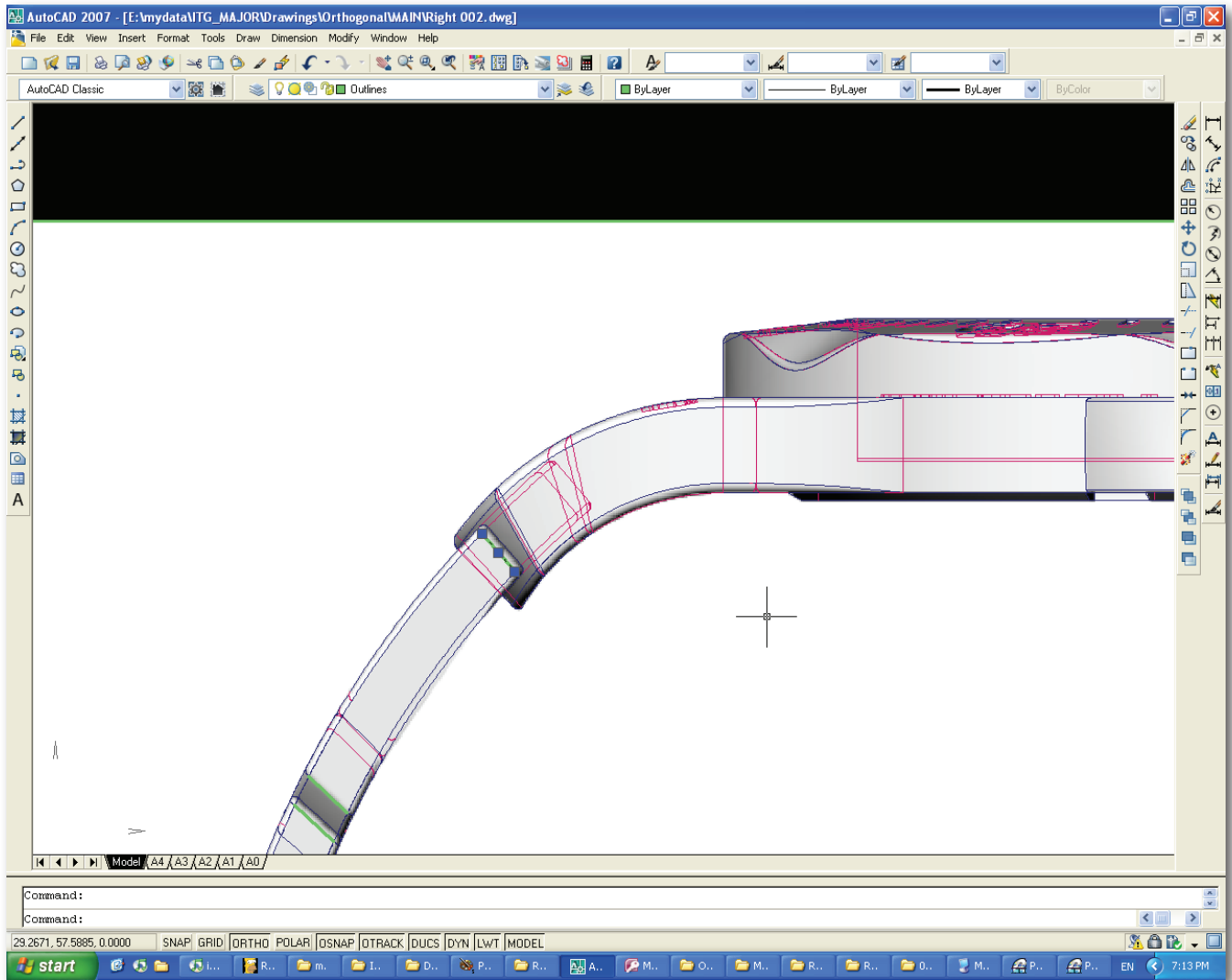


Figure 10.7 - Creating 2D drawings using a background image

When creating 2D line drawings, I used a rendered image placed behind the drawing to see exactly where lines should go, and if they are hidden lines or not. Although the software did generate the lines for me, it was not 100% reliable. This is one way I managed quality control.



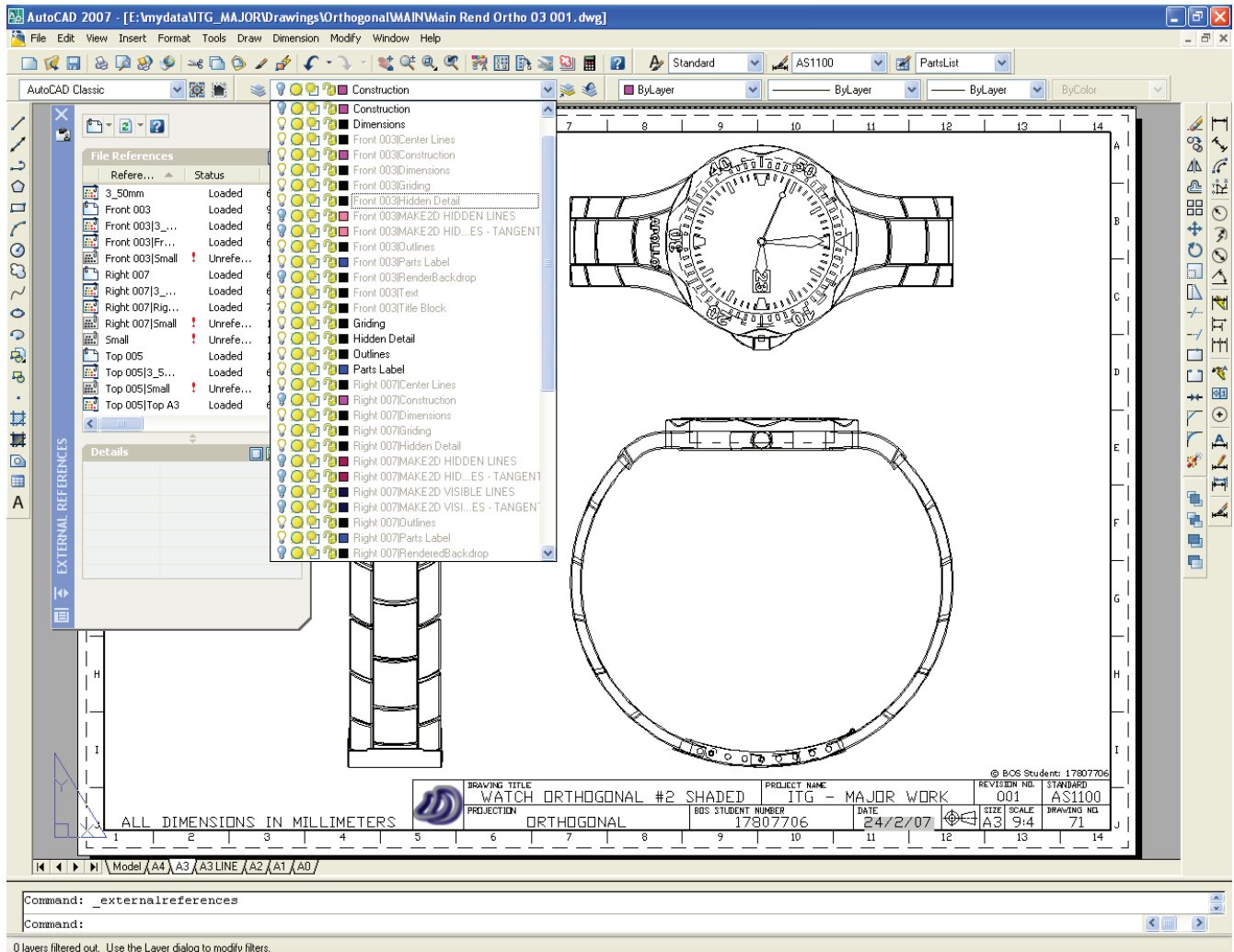


Figure 10.8 - Use of Xrefs in AutoCAD®

I used Xrefs in AutoCAD because it meant that if I made a change to one drawing it would be changed in subsequent drawings that were based upon that first drawing. This was especially made use of in the orthographic drawings of the whole watch. As I made a range of orthographic drawings each slightly different, but all still had the same raw model data. Using this process (as part of quality control) ensured quality of the product as it meant that no two drawings would conflict with each other. It also made it easier for me, as if I found that a change was needed I only had to make the change once on one drawing instead of change all the drawings that had that part in it.



- 10. EVIDENCE OF INDUSTRIAL PROCESSES/TECHNOLOGIES -

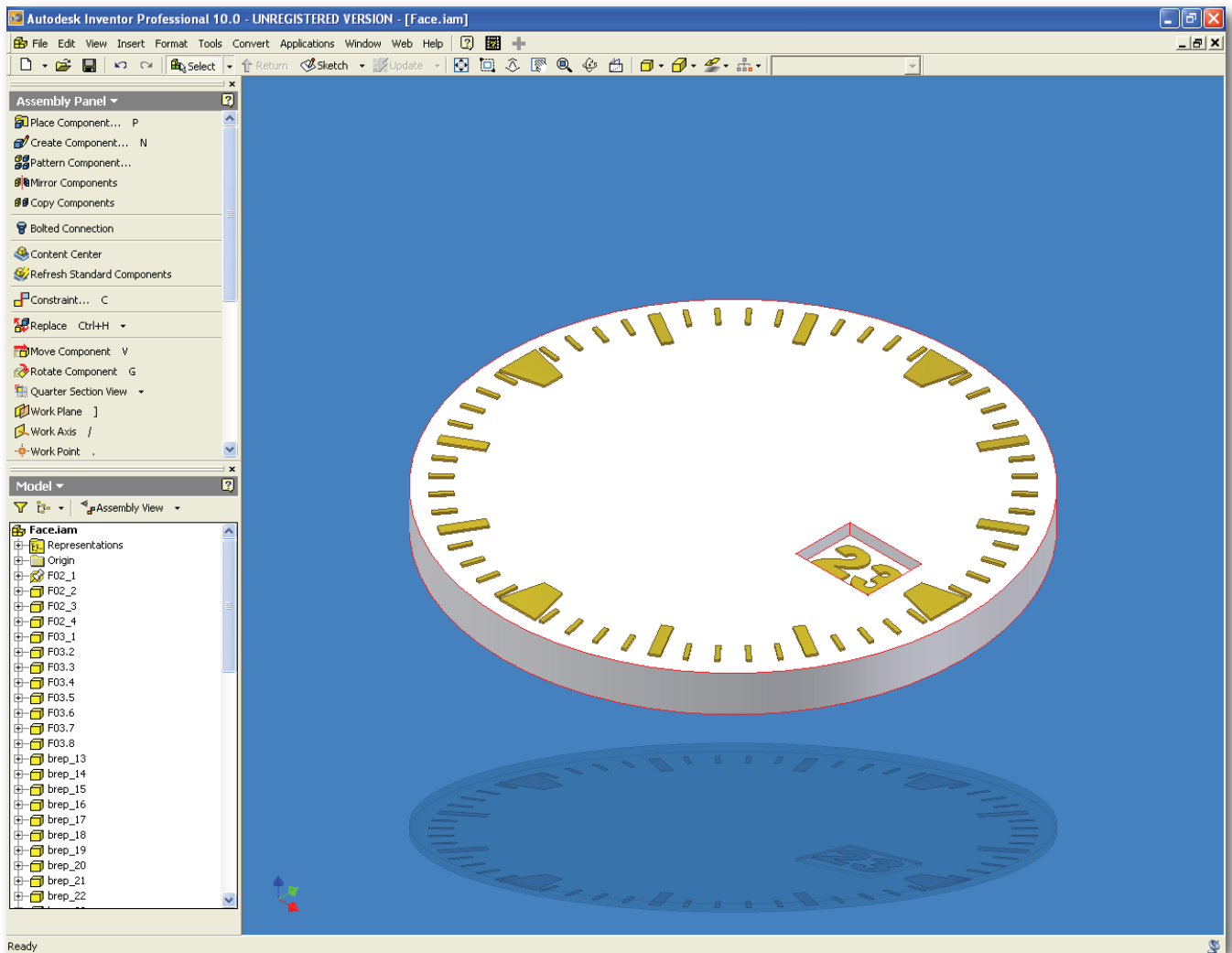


Figure 10.9 - Use of Autodesk® Inventor® Professional 10

I explored Autodesk Inventor, however I did not use it as my primary tool. Although it was better at parts and part assemblies than Rhino, in my opinion Rhino's modelling tools and interface was much easier to use. Because my watch only had a relatively low number of parts, I could get away with doing the whole assembly in Rhino. The screenshot above shows a part of my watch (the face) imported into Inventor. I then applied materials. Inventor did not provide any 2D drawing creation features that could not be done using Rhino, so that is why I used Rhino and not Inventor.

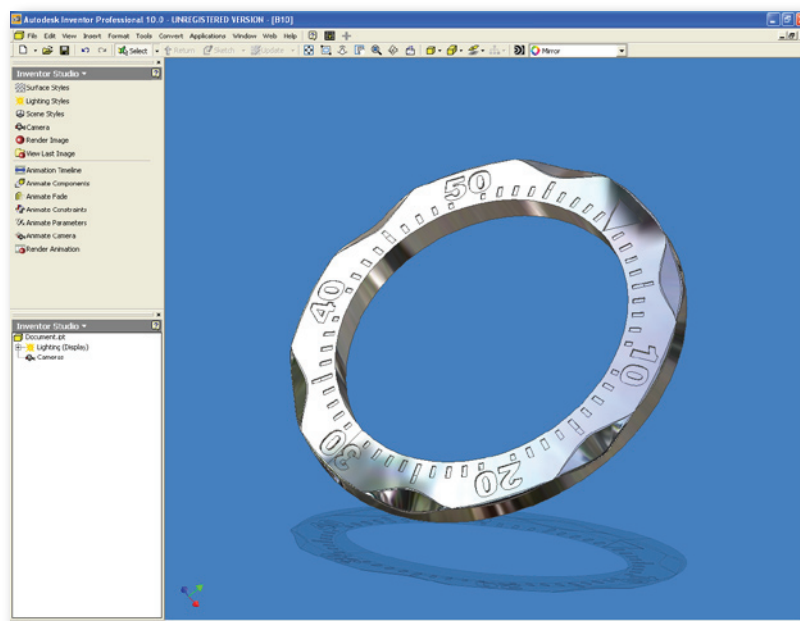


Figure 10.10 - Timer in Inventor with materials applied





- 10. EVIDENCE OF INDUSTRIAL PROCESSES/TECHNOLOGIES -

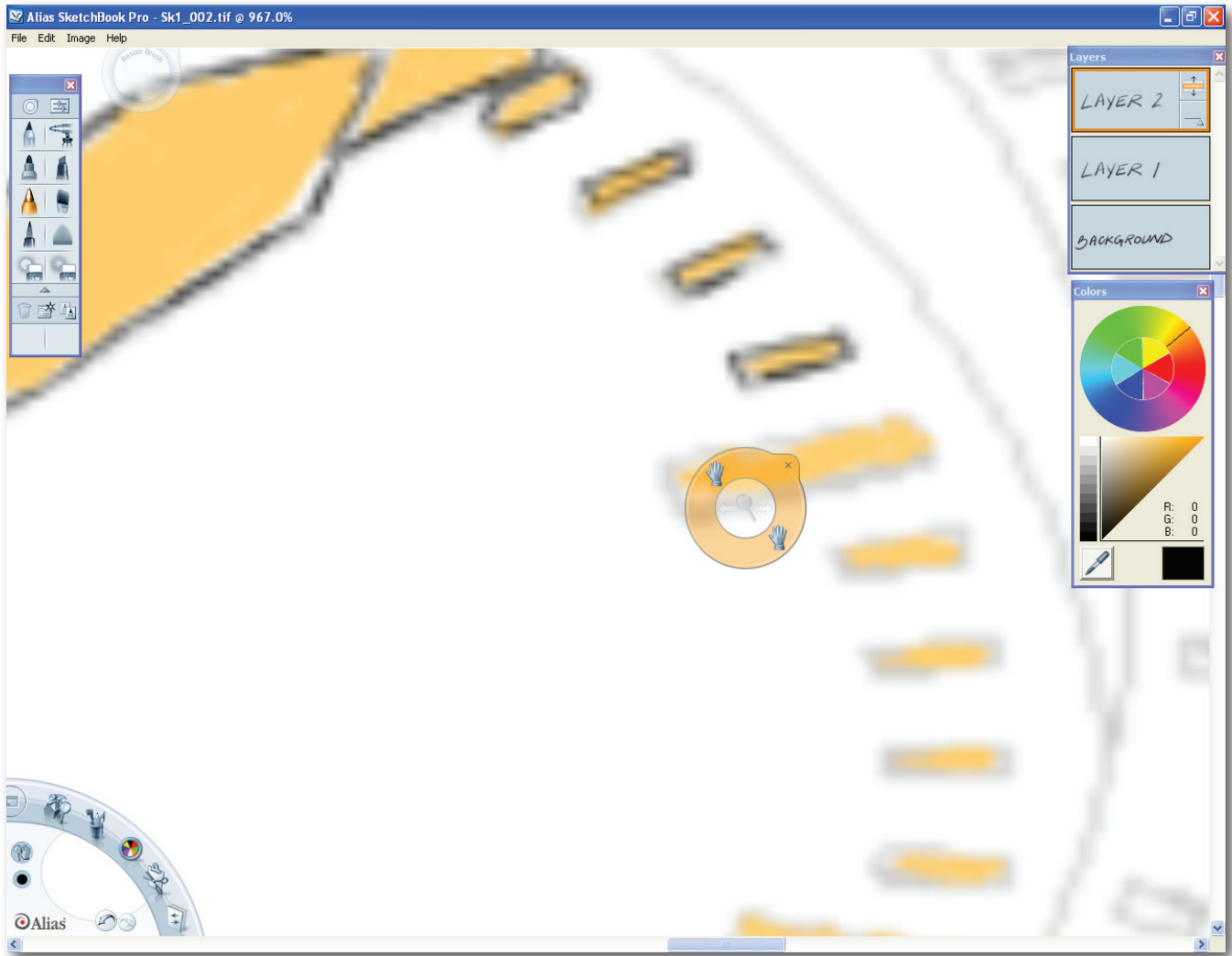


Figure 10.12 - Freehand Sketching and Painting on Computer

I used this painting software in conjunction with a graphics tablet to produced computer freehand sketches.



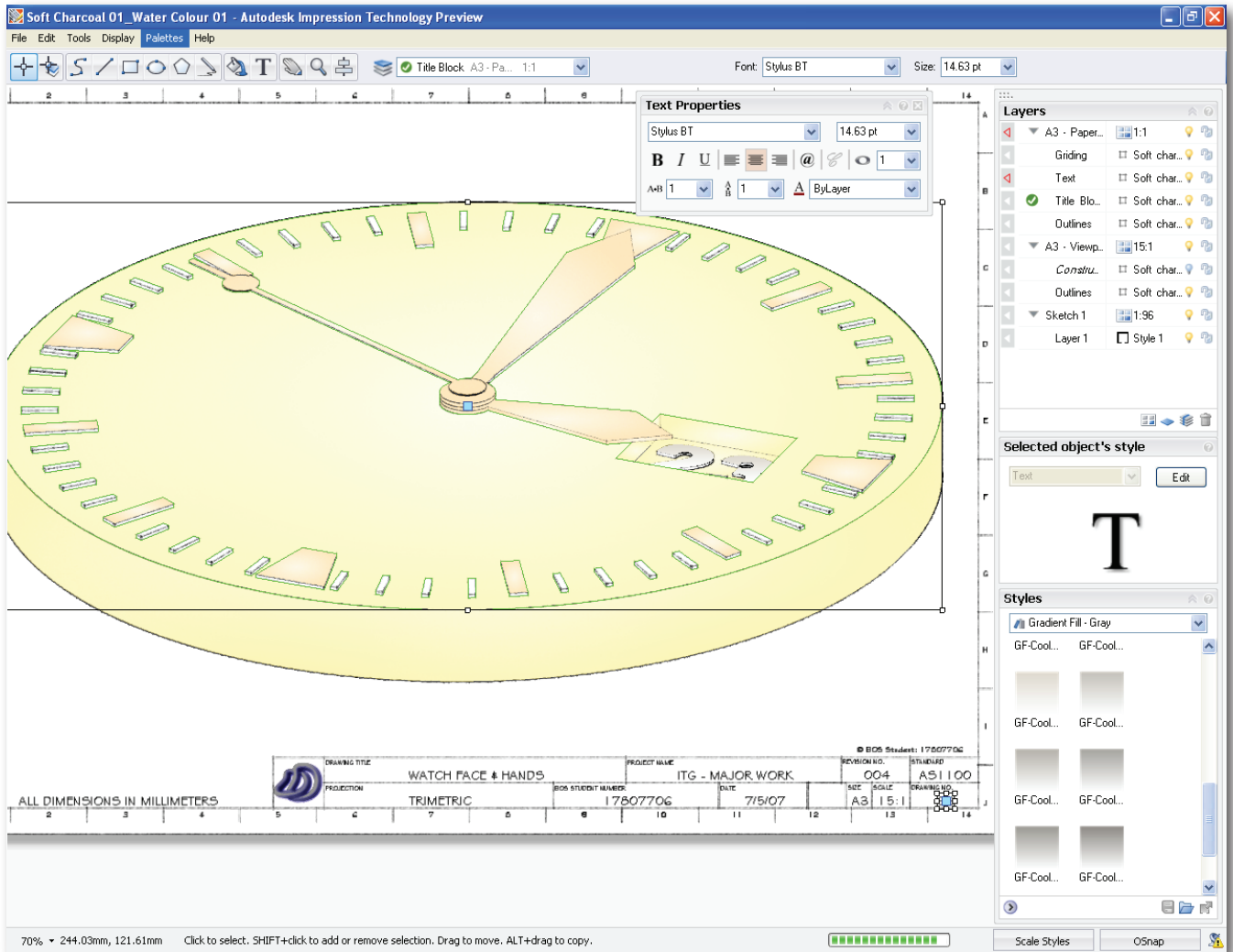


Figure 10.13 - Use of Autodesk Impression to get freehand looking renders and drawings

Impression is a piece of software that takes 2D line drawings and allows you to make them look like hand sketched drawings, and you can also make them look like hand rendered drawings. I used this program so that I could make good looking presentation drawings, that looked less technical due to the fact that they looked hand sketched and rendered.



SOLUTIONS TO PROBLEMS IN PRODUCTION

In the production of my major project I encountered many problems. The nature of these problems was diverse and included problems such as computer/software problems, design problems, finance problems, time management problems and graphics problems. I found ways to overcome these problems, but I also did things to minimise the risk of problems occurring. Documentation of design problems can be found in the Development of Ideas section, page 22. Problems which arose during the production of developed software are partially documented in Appendix A, page 138. Please note also that ViewIt (see Appendix A → ViewIt, page 141) was created as a solution to a computer/software/graphics problem that only arose during production.

COMPUTER/SOFTWARE PROBLEMS

COMPUTER CRASHES

To minimise the risk of computer crashes happening to me I used Windows XP Pro SP2. By using SP2, lots of the bugs in the operating system that would cause my computer to crash were fixed. Also I tried to limit the number of programs I had open, this freed up RAM which meant less lagging and less likely hood that the system would crash. This seemed to work because my computer didn't crash at all, however programs did. I found that programs I was using crashed all the time. This was annoying and a waste of time at the most, but I didn't really loose much work because of certain measures that I had taken which are explained below.

SOFTWARE CRASHES

Software crashes were frequent (see figure 11.1). They were commonly encountered when using Rhinoceros. Some of the reasons for this is the program was still being BETA tested, and because of the nature of the software, you can easily preform operations that use more RAM than my computer has and require huge amounts of processing. Normally this would just mean that it takes time for certain operations, however sometimes the software just crashed. Apart from this being a nuisance, it also could potentially cause me to loose data. So I turn on the *autosave* option, and set it to frequent. This meant that if the program did crash, I would still have a fairly recent copy of the file I was working on. (See also, Selection and Justification → Processes → Production of Project → Layers in Rhinoceros, page 56)

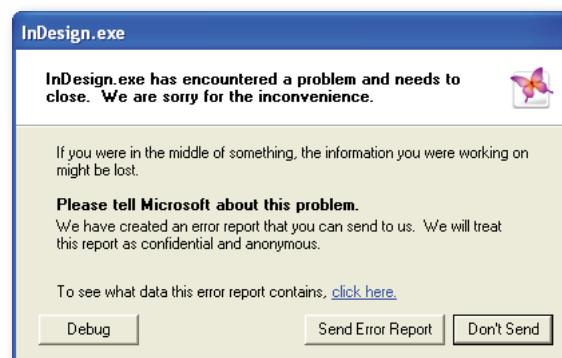


Figure 11.1 - Software Crash Dialogue Box

COMPUTER VIRUSES

It is possible for my computer to get a virus that could potentiality cause total data loss. So to make sure that this wouldn't happen I used antivirus software, and also made back-up's of my project at stages. I made back-up's onto a Re-Writable CD, however I rarely needed to do this, because due



to the nature of the work, I was constantly working at home and at school. To make this process easier I used software to synchronise the data, so I had three copies, one on my home computer, one on my flash drive and one on the school network. So if something happened to one set of data, I still had it on the other two mediums. Back up became more difficult during the later stages of production when the total file size of the project grew to greater than 12GB. This synchronisation process is explained in the next paragraph.

MANAGEMENT OF DATA

When I was working on my major project, I did some at school and some at home. Sometimes I worked in other places. To make sure that I was working efficiently I had to synchronise my data. Copying files from my home computer to my flash drive to work on them at school, and then copying them back to my computer at home is an extremely inefficient and error prone method. To work on my project (the computer parts), I had to work out some way to synchronise my data. At first I used Windows built in *Briefcase* folder this provided low level synchronisation, but I didn't use it, instead I chose to use Microsoft SyncToy. This was a much better synchronisation tool. Reasons include it could deal with renamed files, and it didn't only allow one application to open the folder at any one time.

CREATING MESHES

When I made my 3D model in Rhinoceros, the final model was made up of a collection closed polysurfaces. In order to export this model to certain applications and mediums such as 3ds Max for applying materials and rendering or to a rapid prototyping machine I needed a mesh. Firstly I'll explain the terminology and core basics of Rhinoceros objects. Rhinoceros is NURBS oriented, that is you make a model out of surfaces. If these surfaces join without gaps such as the surfaces on a cube, you have a closed polysurface. Every time you render this model inside rhinoceros or when it is displayed as shaded in the viewport as mesh is created, and it is that mesh that is rendered and displayed. Where NURBS can define a curve or surface with infinite precision, a mesh on the other hand cannot. A mesh basically is made up of vertices and edges. Meshes can only approximate NURBS curves and surfaces. That explains the terminology and basics, so the problem that I faced with this is when my polysurface was converted to a mesh. I set the mesh detail option to the highest possible to get as smooth a model as possible, however some parts of my newly created mesh had holes and other technical problems. The holes in the mesh were so small that you could not see them and some of them could be fixed. These holes didn't cause a problem with the first use of my mesh, rendering because they were small. However these holes could potentially cause problems if I were to send this mesh to a rapid prototyping machine. But also the other technical problems could also cause rapid prototyping problems.

After fixing what I could the only mesh problems came from the band mesh. I could not fix these, thus I did not rapid prototype the band. These mesh problems occurred due to the way I modelled the band and I did not have time to find and use a different method to make the band, so I was left with a mesh with problems.

GRAPHICS PROBLEMS

OBLIQUE CIRCLES

In my final design of my wristwatch, many features are circular, and as Rhinoceros could not generate a 2D oblique drawing from the 3D model I would have to do it manually on AutoCAD. This posed a great problem for me, as I was unaware of any shortcut method for producing oblique circles. I would have to use the method of plotting, ie. measure distances off my orthogonal drawing and use those distances to put dots on the oblique drawings, then join the dots. However this was a very repetitive, boring, inaccurate and error prone method. I didn't like it, and as I couldn't find any other solution in any technical drawing text book, I invented my own. Actually it is kind of the same as the plotting method but it is automated. Basically I derived the cartesian equation of a top plane circle, projected into 2D by oblique projection. (See figure 11.2)



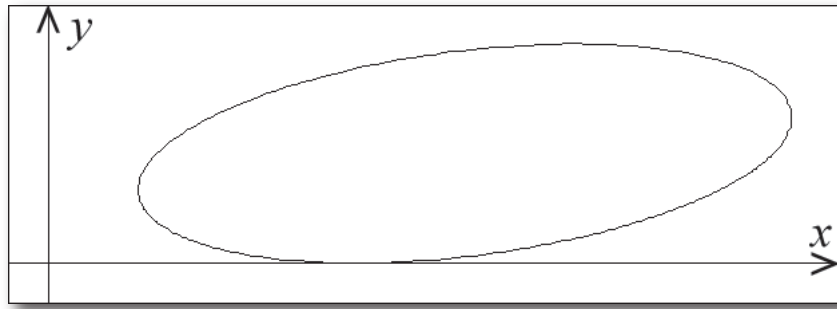


Figure 11.2 - Top plane circle in cabinet oblique projection, in x,y coordinate system.

The equations I came up with were,

Cabinet	Cavalier
$x = r + y \pm 2\sqrt{(r\sqrt{2} - 2y)y}$	$x = r + y \pm \sqrt{(r\sqrt{2} - y)2y}$
Centre of circle: $\left(\frac{r(\sqrt{2} + 2)}{2}, \frac{r}{2\sqrt{2}}\right)$	Centre of circle: $\left(\frac{r(\sqrt{2} + 1)}{\sqrt{2}}, \frac{r}{\sqrt{2}}\right)$
Area of Circle: $\frac{\pi r^2}{2\sqrt{2}}$	Area of Circle: $\frac{\pi r^2}{\sqrt{2}}$
Cabinet	
$y = \frac{1}{9} \left(-r + 2r\sqrt{2} + x \pm 2\sqrt{-r^2\sqrt{2} + 4rx + rx\sqrt{2} - 2x^2} \right)$	
Where r is the radius of the circle.	

Now I could generate to whatever accuracy I wanted the points on the circle. As I could just scale it in AutoCAD to get circles with different radius, I only needed to generate it for a radius of one. Using Microsoft Visual Basic 6.0¹ I generated a set of points that made up the circle (I generated 1416 points), then I made these into a set of lines in AutoCAD and grouped them, and placing a *node*² at the centre. I could then manipulate that group of lines in AutoCAD to get the other side face, and scale it to gain other radii. Note that some of the mathematics was done with the aid of Mathematica 5.2³.

This is evidence that I have learnt the syllabus dot point, "Students learn to: use planes and coordinates to plot computer generated or mechanically drawn outlines of intended designs wherever relevant to projects undertaken".

Also the wide range of skills that I used to produce the oblique circle, shows *evidence of a range of skills*, and it shows that my skills are not just limited directly to graphics skills, but I can use other skills to solve graphics problems. This was devised early in production, prior to the development of *ViewIt* (Appendix A - Developed Software → ViewIt, page 141). After the development of *ViewIt* this solution became redundant. The early method described here was used for the production of drawing item 22.

ISOMETRIC PROJECTION VS ISOMETRIC DRAWING

To get isometric drawings from my 3D model I changed the Rhinoceros viewport to parallel (this eliminated the perspective distortion) and then to get isometric rather than any axonometric, I set the camera target to 0,0,0 and the camera location to x,y,z. Where the x, y and the z were of equal

1 Visual Basic® is a programming language and interface.

2 AutoCAD® terminology for a point.

3 I used a 15 day free trial version of Mathematica. Therefore it is not on my finance plan. It was not included in the Selection and Justification section because it was not a major resource used. As with footnote 1.



magnitude. (eg. 100,100,100 OR 100,-100,100)

Isometric drawings should have the three axis at full scale, however the above method used resulted in an equal and flat foreshortening (perspective distortion would result in non flat foreshortening) of the three axis. To fix this I applied a 2D scale transformation to the 2D isometric projection drawing at a scale of $1.2247448727831780997784461020033 \approx [\text{Sqrt}(6)/2]$ which resulted in the correct axis scaling required for an isometric drawing.

REPRESENTATION OF FILLETED EDGES

This was a problem for me because many parts of my watch had filleted edges. This was done because it increased the aesthetics and safety of the product. The following is illustrated in figure 11.3. By standard, tangent edges are not shown on 2D drawings, however this can make a product hard to visualise and make the drawings unclear and in some instances impossible to define a shape even when using an orthogonal drawing. So I included filleted tangent edges on my drawings to make them clearer and easier to understand.

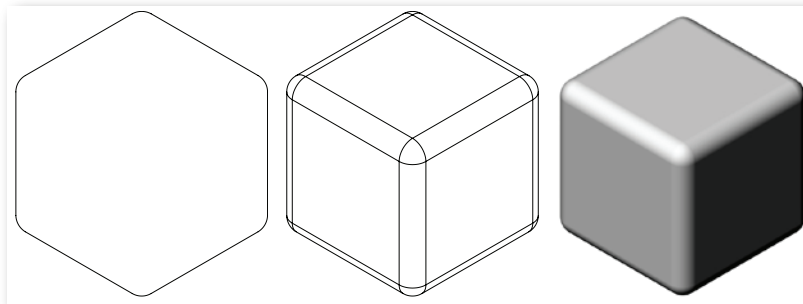


Figure 11.3 - Left: Tangent edges omitted, Centre: Tangent edges shown, Right: Rendered

In some circumstances I could use the method shown in figure 11.4 (drawing no. 32). However this was not always possible.

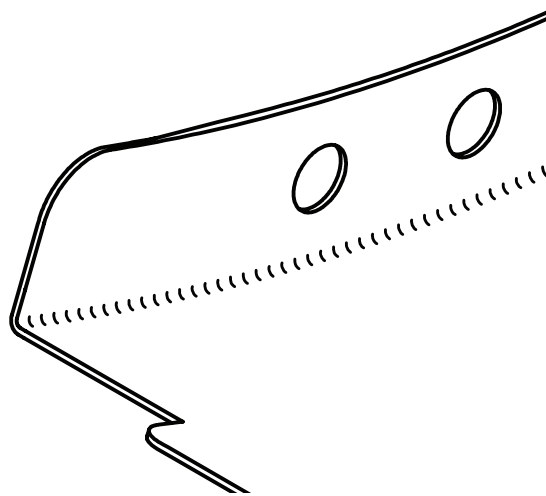


Figure 11.4 - Representation of Filleted Edges (from drawing no. 32)

SCALES & AS1100

This posed a problem for me because I had to decide whether or not to follow AS1100 in regards to standards. The standards "recommended" enlargement scales of 2:1, 5:1, 10:1, 20:1, 50:1. Herein lies the problem, these are big jumps and sometimes say 5:1 made the drawing too large and bigger than the page, but 2:1 left too much space surrounding the drawing. A scale in between would be much better but the standards I am following don't "recommend..." that. So I decided to try to follow the standard where I could and when following left too much space I used a different scale but still tried to keep it reasonable.



COLOUR SPACE

This was a problem for me mainly because I was not an expert in colour spaces and since graphics is all about colour, knowledge and understanding would allow me to produce the best results. Yes even if I had no idea about the colour space and different colour models I could still produce a folio and production component that look good, but I can get much better results and knowledge if I know something about it. As a result of not knowing much and needed to know the knowledge, I researched and learnt about colour models during production. I did this through software manuals and text books. My main problem was converting all my rendered images, graphics and folio from the RGB colour model which is used to display images on a computer model to the CMYK colour model which is used to print colours onto paper, film and other media.

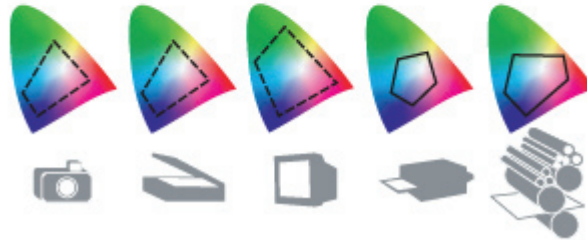


Figure 11.5 - Colours possible with different devices

Source: Adobe Help Centre

COMPUTER RENDERING PROBLEMS

TOO SLOW

This was a big problem to start off with. I was not a professional at using and doing materials, lighting and rendering in 3ds Max, and therefore I did a lot of trial and error in making adjustments to rendering, lighting and material options. This resulted in having to make an adjustment and then render to see exactly what effect it had. But after I got the settings right I notched up the quality and the rendering times increased dramatically. All the extra features that made my renderings photo realistic such as global illumination (bleeding of colour), caustics (effects of light passing through glass), ambient occlusion (more realistic lighting effects), HDRI environment maps (shows reflections from the environment (these made metals look real, without the HDRI map the stainless steel would look fake)), high resolution shadows, high resolution and high quality reflections, including glossy reflections and the resolution of the output file (large enough to print on A3 size paper and not look pixelated) drastically increased the rendering times. To ensure the quality of my product lowering the settings was not an option. Using a Pentium 4 with 2.3GHz it could take an average of 10 hours (some even took 48hrs) for some scenes to render just one frame and if I wanted to render an animation then it would take much longer, although an animation would not require the same output resolution as it would not be printed. So I needed a way to get the renderings rendered faster, so I considered the options. Firstly I could have bought a super computer but I soon found out that I could not afford one. So I thought about getting a faster processor, such as an Intel 1.4GHz Itanium 2 (see figure 11.6) or an Intel Core 2 Duo, however that would be really expensive and would only increase efficiency by 33%, which is good but compared to the other options available it was not the best solution. Also I cannot just change the processor, I would need to but a whole new computer.



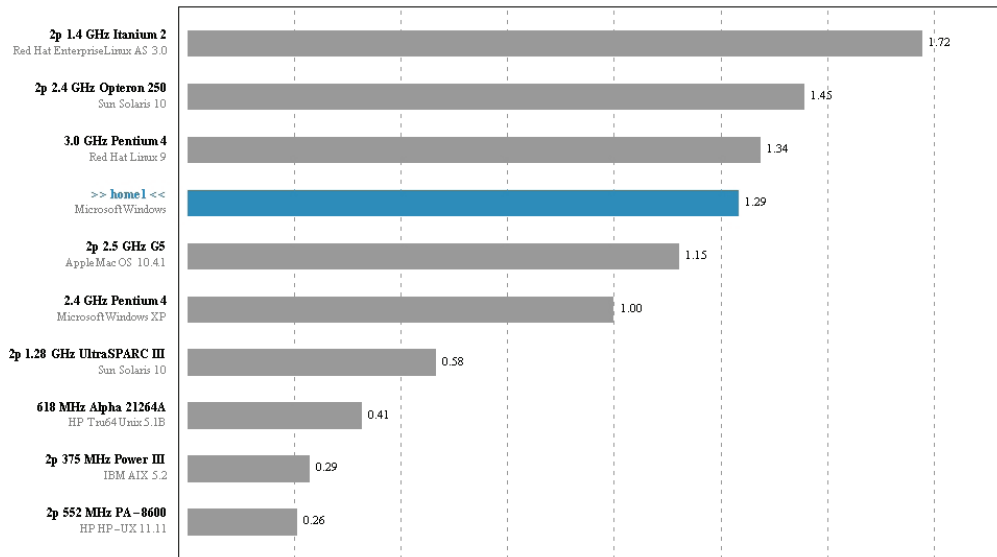


Figure 11.6 - System Performance
(Graphic generated from Mathematica)

After investigation my options I decided to take advantage of the schools network and utilise all the computers just sitting idle at night time and over weekends. The school leaves all computers turned on overnight, so instead of them doing nothing I could get them to render my images a lot faster. Theoretically given the school has over 110 Pentium 4 computers, my images would take 1/110th of the time to render. Unfortunately this was not the case. As firstly the network link speed on the manager (sends out jobs) computer would be too high to send jobs out to all computers at once, so at the start it might be slow but as the slave (rendering) computers demands become more spread out the manger computer would cope with the load. Secondly, although the slave computer does not need a licence to run, it still needs the software installed. Unfortunately I was not allowed to do this kind of mass installation by the network administrator. However I was able to use some computers in the graphics lab. The use of these computers at night and over weekends meant that I got my renderings done faster.

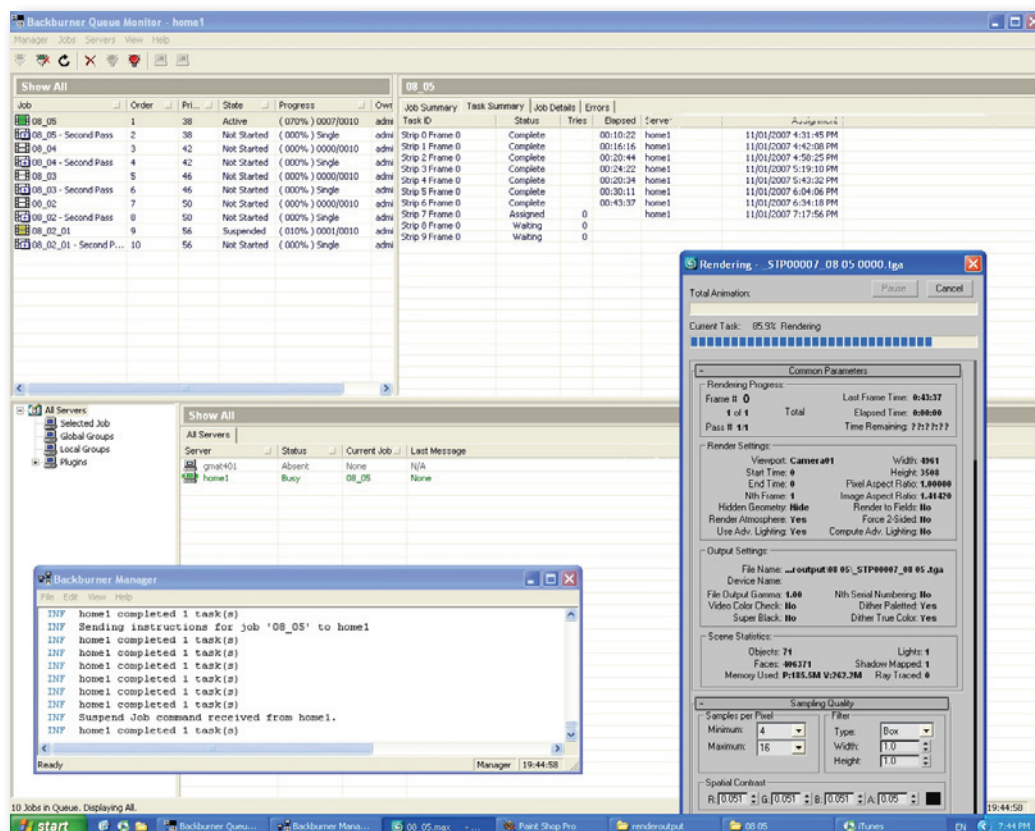


Figure 11.7 - Network rendering screenshot



- 11. SOLUTIONS TO PROBLEMS IN PRODUCTION -

I also sometimes used the mental ray satellite. Which is simple and exe that runs in the background of a computer. If another computer on the network uses distributed bucket rendering and choose that computer with the satellite exe running to help rendered the image. Then with out any other settings of things needing to be done or turned on, when the render button is pressed the image will render just as if only one computer was contributing, however it will presumably do it half as fast and you will most likely have two blocks of your image rendering at the same time.

This mental ray satellite only works with the mental ray rendered however. But as most of my renderings were done with mental ray this was not a problem. Advantage of using the satellite system vs. the backburner system include I do not need to worry about maps and if they are available at the referenced path on all the computers; I do not need to worry about where all the output files go and if they will go to different computers; and I don't need to install the 3ds Max software, I just need to install the satellite, which is much easier and faster once I worked out how to set it up through internet research and reference to the user manuals.

If I needed my renderings done faster and if I had much longer animations and a much greater budget then I could have utilized commercially available render farms or supercomputers. The computational power of different mediums that I could have utilised are shown in the table below.

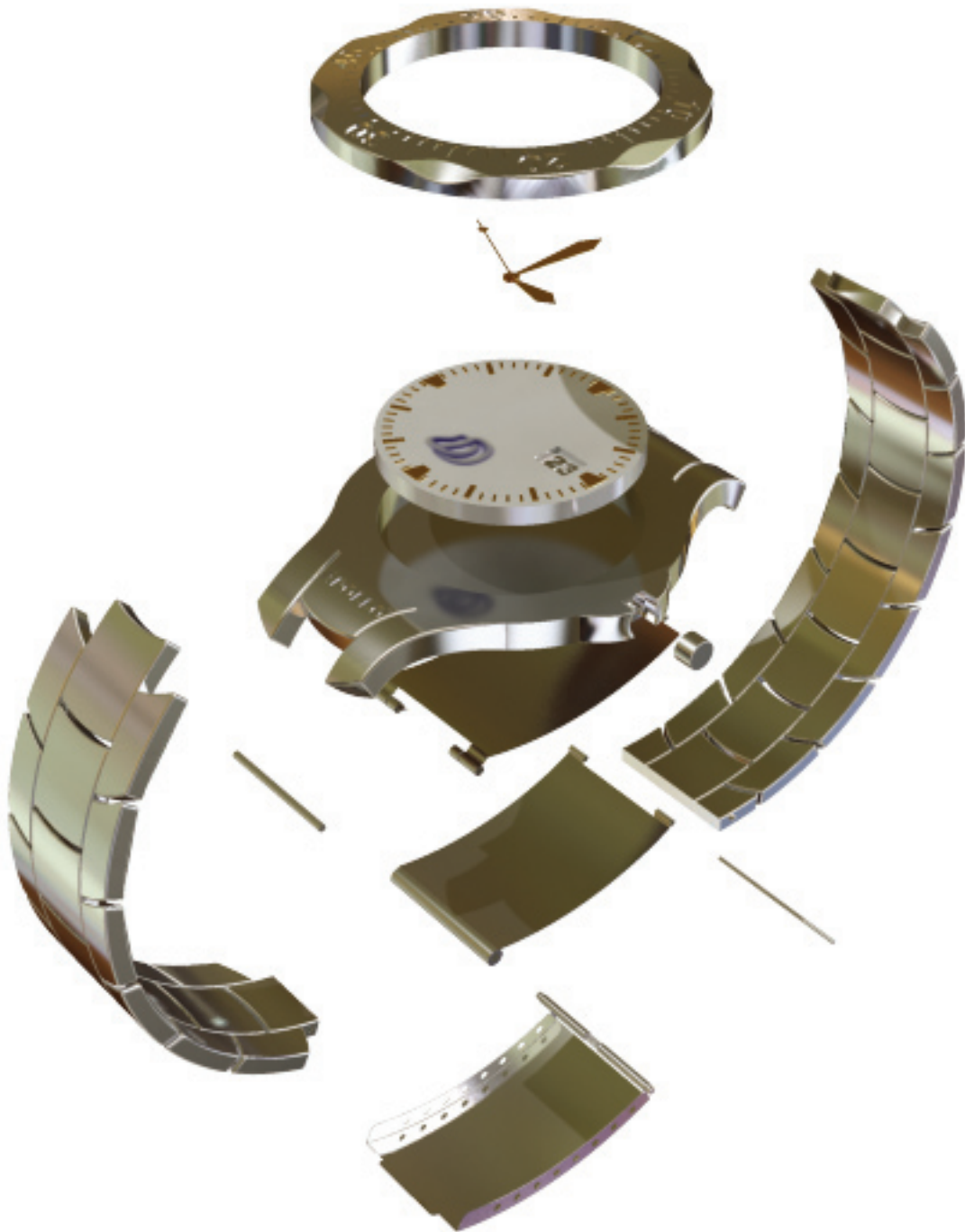
Device	FLOPS
Intel Pentium 4 @ 3.2GHz	8 GFLOPS
XBox 360 @ 3.2GHz	118 GFLOPS
PS3 Cell @ 3.2GHz	218 GFLOPS
PS3 System	2 TFLOPS (2000 GFLOPS)
BlueGene/L	36 TFLOPS (36 000 GFLOPS)

Table 11.1 - Computing Power of various devices

PATH NOT FOUND ERROR

As mentioned in the section above, I used network rendering to solve the problem of renderings being too slow. But this also came with its own problems. One of them I uncounted the first time I used it, and was solved simply. The problem was when the slave computer rendered the image it tried to save it to my removable disk drive. This was a problem when one of the other computers couldn't find my drive attached to that computer. One obvious solution would be to save the images to the C drive, but then I would have to manually collect the images from all the computers. But even worse, if I was only rendering one image then the image would not save at all. In the end I used UNC naming conventions. An example is, \\GL005\\e\$\\folder\\file.jpg where a file would be directed to a specific computer and its specific drive.





APPENDIX A - DEVELOPED SOFTWARE

This section documents the computer software I created to help produce my Major Project. All aspects of the development including mathematical derivation, interface design, programming (coding) and documentation of problems and solutions were completed by me, 17807706. I did use secondary information and advice and I have documented all sources of information and assistance in the appropriate location, being either footnotes, references, bibliography or acknowledgements.

OBLIQUE PROJECTION TOOL

This is a small tool I created early in production. It was later expanded into *ViewIt*, which appears after this tool. In simple terms it takes 3D points and converts them into 2D points in oblique projection.

VISUAL BASIC® 6 USER INTERFACE

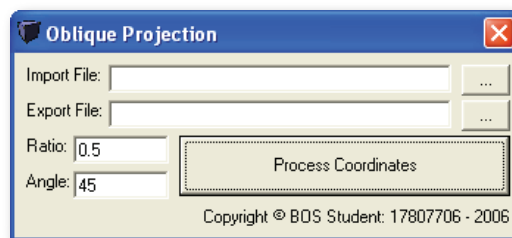


Figure 12.1 - User Interface

VISUAL BASIC® 6 CODE

```

'/////////////////////////////////////////////////////////////////
'// This tool takes 3D Coordinates and converts //
'// them into 2D coordinates using oblique //
'// projection. //
'// //
'// VB Version: VB6 //
'// Developed by BOS Student: 17807706 //
'// Produced for HSC IT-Graphics Major Project //
'// Copyright © BOS Student: 17807706 - 2006 //
'/////////////////////////////////////////////////////////////////

Dim compos1, compos2 As Integer 'comma positions in input line of coords (eg. input line look like
1,2,5 it needs to be converted into the x,y and z part separately
Dim x As Double 'x,y,z are the 3D coordinates of the 3D point
Dim y As Double
Dim z As Double
Dim x2 As Double 'x2,y2 are the 2D coordinates of the projected point
Dim y2 As Double
Dim k As Double 'k is the oblique ratio (cabinet = 0.5, cavalier = 1)
Dim alpha As Double 'alpha is the angle, usually 45°
Dim textline As String 'textline is the string of text for a line in the input file.
Const pi As Double = 3.14159265358979

Private Sub cmdGo_Click()
If txtImport.Text = "" Or txtExport.Text = "" Then
MsgBox "Please specify a file for import and export.", vbInformation, "File Missing" 'if the file
path is not specified
Else
k = CDbl(txtRatio.Text) 'stores ratio in memory
alpha = (CDbl(txtAngle.Text) / 180) * pi 'converts the angle to radians and stores in memory

Open txtImport.Text For Input As #1 'opens import file

```



```

Open txtExport.Text For Output As #2 'opens export file

Do Until EOF(1)
    Line Input #1, textline
    compos1 = InStr(1, textline, ",", 0) 'calculates the position of first comma in string
    compos2 = InStr(compos1 + 1, textline, ",", 0) 'calculates the position of second comma in
string

    x = CDBl(Mid(textline, 1, compos1 - 1)) 'input X coord
    y = CDBl(Mid(textline, compos1 + 1, compos2 - compos1 - 1)) 'input Y coord
    z = CDBl(Mid(textline, compos2 + 1, Len(textline) - compos2)) 'input Z

    x2 = x + k * Cos(alpha) * y 'apply's oblique projection for X
    y2 = z + k * Sin(alpha) * y 'apply's oblique projection for Y
    Print #2, x2 & "," & y2 & ",0" 'writes the 2D point to text file
Loop

Close #1 'close import file
Close #2 'close export file
MsgBox "Done.", vbInformation 'display done message box
End If
End Sub

Private Sub cmdExport_Click() 'gets file path
CD.Filter = "Text Files (*.txt)|*.txt|All files (*.*)|*.*"
CD.DialogTitle = "Export 2D Points"
CD.ShowSave
If CD.FileName <> "" Then txtExport.Text = CD.FileName
End Sub

Private Sub cmdImport_Click() 'gets file path
CD.Filter = "Text Files (*.txt)|*.txt|All files (*.*)|*.*"
CD.DialogTitle = "Import 3D Points"
CD.ShowOpen
If CD.FileName <> "" Then txtImport.Text = CD.FileName
End Sub

```

DERIVATION OF OBLIQUE TRANSFORMATION EQUATION¹

The two lines which do the actually oblique transformation are:

```

x2 = x + k * Math.Cos(alpha) * y
y2 = z + k * Math.Sin(alpha) * y

```

Mathematically represented as:

$$x' = x + k \times \cos \theta \times y$$

$$y' = z + k \times \sin \theta \times y$$

These two lines take a 3D point as input and give a 2D point as output. The derivation of these two lines is as follows.

Firstly I used the z-up left hand coordinate system (see figure 12.2). Although this is not the most commonly used system, it is the system used by Rhinoceros 4.0.

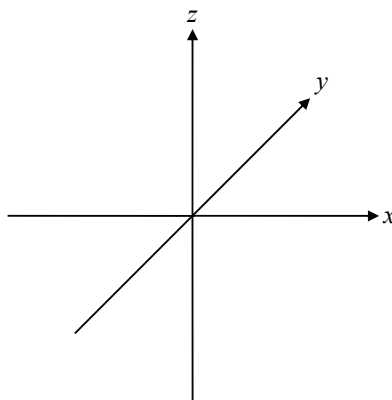


Figure 12.2 - z-up, left hand coordinate system

¹ The equations were obtained through research at http://en.wikipedia.org/wiki/Cavalier_projection, however I did the derivation myself.



The 2D coordinate system that I used is the standard xy plane that is used in maths and AutoCAD, shown in figure 12.3.

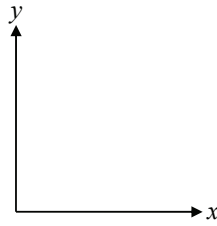


Figure 12.3 - 2D coordinate system

Now I split into x and y . Working with each individually. From here on the 2D coordinates will be x' and y' to avoid conflict with the 3D coordinates. Logically x' will be equal to x plus 'something'. That 'something' will depend upon how far into the y axis the point is. From elementary maths I we get the following relationships.

$$\begin{aligned}x &= r \cos \theta \\ y &= r \sin \theta\end{aligned}$$

where r and θ are the coordinates in polar form and x and y are the coordinates in rectangular form.

The r and θ will be the ratio and angle respectively (for cabinet oblique, $r = 0.5$ and $\theta = 45^\circ$). So that 'something' extra will be, for x' , $r \cos \theta$. And as the further back this point is on the y axis, this value will get larger. So $r \cos \theta$ is multiplied by y . The same process is done for the y' value except \sin is used. Put this all together and you get those two lines of code.

Note that the oblique projection matrix is:

$$\begin{bmatrix} 1 & k \cos \theta & 0 & 0 \\ 0 & k \sin \theta & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

EVALUATION

This self developed tool proved useful, however it could have been extended. Currently the tool only took points as input. However my 3D model contained NURBS¹ surfaces, it was not made up of point and sharp corners. I could have converted my *polysurfaces*² into a mesh and use the points on the mesh as input data, however to get a smooth enough mesh thousands of points are needed and as my software did not join the exported points this option was impracticable. (Mesh problems can be found at Solutions to Problems in Production → Computer/Software Problems → Creating Meshes, page 131.) So I used this utility in conjunction with my oblique circles as described in Solutions to Problems in Production → Graphics Problems → Oblique Circles, page 131. So this utility only helped me to make oblique drawings that were only made up of straight lines and circles. This utility also requires a great deal of human work after and before the data is processed, that is why I made many changes when developing *ViewIt*. The details and improvements of made in *ViewIt* are found in that section.

The oblique projection tool was made in 2006 and only took CSV as input and CSV as output. Also I did it in Visual Basic 6.0 which I was accustom to using, however it is a very old version and big changes were made in the latest release Visual Basic 2005 (I used the free Express Edition). So in 2007 I got a book on VB 2005 and learnt the differences between the two languages. Now armed with the knowledge I could convert my application into VB 2005, when converting it I realised that I needed a much more powerful and better tool, so I turned it into a better application. More details are found in the next section (next page).



- 1 Non-Uniform Rational B-Splines. A method of defining curves. Can also be used to define surfaces.
- 2 Rhinoceros terminology for a collection of surfaces.

VIEWIT

I began production of ViewIt in January 2007. I finalised release 4.8 in May 2007, which was the version I used to develop parts of my major project. However its use is not limited to this specific project. Modern architects, engineers and industrial designers no longer base their projects entirely in 2D. Modern software allows for the designing of products and structures in 3D. 3D CAD has many advantages, however we still need 2D drawings. However modern methods involve the modelling of a product or structure in 3D and then 2D drawings are created based on that 3D model. Many current 3D CAD/CAM software products incorporate features that automatically create 2D drawings from 3D data. Unfortunately in the software I have found, this is limited to orthogonal, isometric, and three point perspective views. Although these drawing projections are the most commonly used, the others are still important, and they appear in graphical communication and technical drawing text books. For instance if I want to create an oblique or a planometric drawing I should be able to have the computer software to generate it automatically, in the same way that orthogonal drawings are automated. Especially as the detail and the extent of curves used in modern products and structures is so great that manual methods would be long tedious and sometimes almost impossible.

ViewIt fills this gap that other software products have left. ViewIt takes 3D data from existing 3D CAD/CAM software, and generates 2D drawings from that 3D data. Although there are still many improvements that could be made to the product, it is proving promising so far. ViewIt can generate 2D drawings in cabinet oblique, cavalier oblique, planometric, dimetric, trimetric, isometric, one point perspective and orthographic projections. ViewIt can also generate custom projections (this is due to the way it was designed). In future releases of ViewIt I hope to extend the support to many CAD/CAM applications (currently only Rhinoceros and AutoCAD are fully supported), and I also hope to implement hidden detail features, which were not addressed in the current version due to time constraints and the importing architecture.

Examples of drawings created using ViewIt and raw output from ViewIt can be found at the end of this section at Drawings Created Using ViewIt, page 214.

VISUAL BASIC® 2005 USER INTERFACE

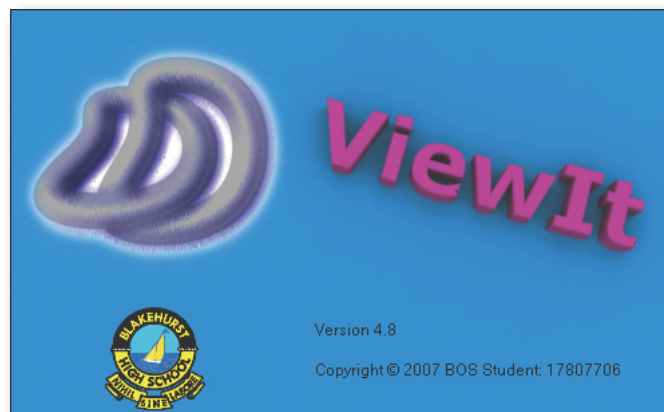


Figure 12.4 - Splash Screen (Is transparent)

Figure 12.5 shows the default user interface of the program. I designed the UI myself from built in Visual Basic.



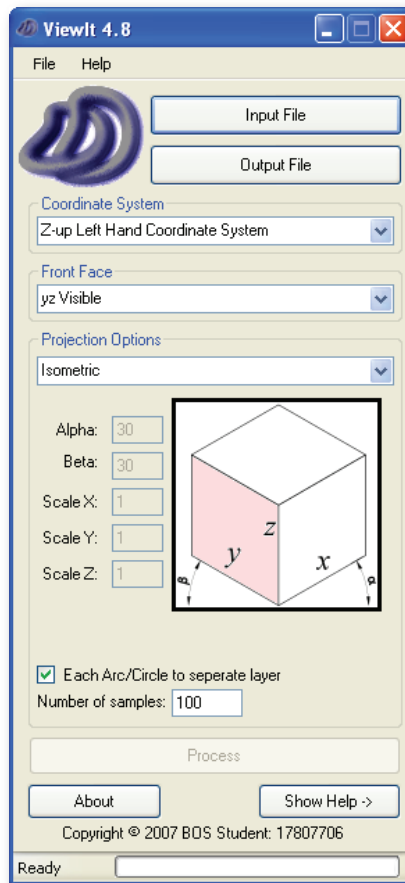


Figure 12.5 - User Interface

Figure 12.6 shows the UI with the help shown, which can be toggled on/off.

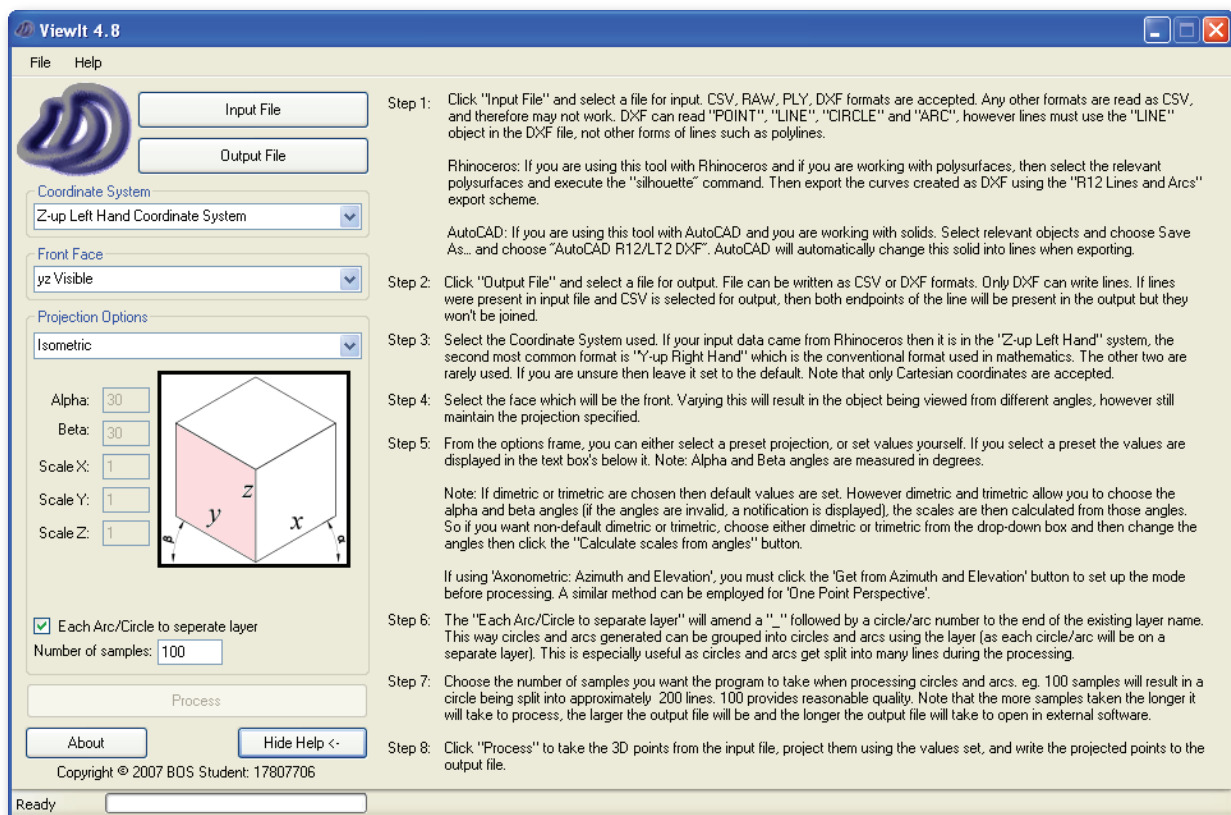
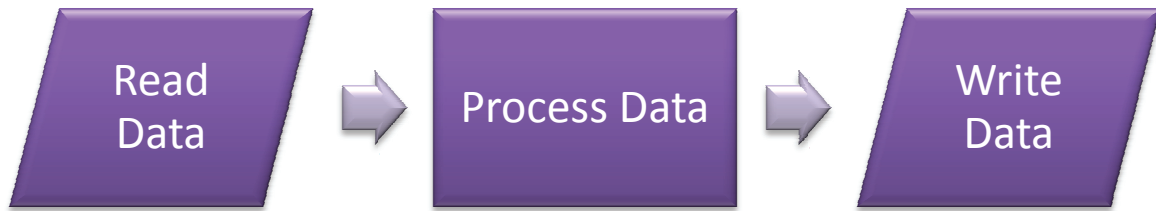


Figure 12.6 - User Interface with help shown



BASIC PROGRAM STRUCTURE



1

This involves opening the file, sorting through it to find the relevant data and then storing the relevant data in the respective variable. The sorting through process to find the data proved difficult in DXF. I had to learn how DXF worked first before I could start using it.

It also involves converting any non line/point data into lines so that it could be processed. This involved converting circles and arcs into many smaller lines, which provided an approximation that could be processed.

Reading and converting circles and arcs involved deep mathematical knowledge; involving converting coordinates from object coordinate system to world coordinate system and applying rotational matrices to match the normal vector given. Also creating the small lines for an arc given a start and finish angle proved difficult and required mathematical ability and the mental visualisation to match.

This was easy as all the hard work and problems had been dealt with in the previous step. All that had to be done here was project a point from a given Cartesian coordinate to a projected Cartesian coordinate on a 2D plane. I developed the mathematical formula to do this for a general rule, involving scaling on each axis and the resulting 2D angle of two of the projected axis. After I developed the formula, I derived the matrix to match it.

$$\begin{bmatrix} S_x \cos \alpha & -S_y \cos \beta & 0 & 0 \\ S_x \sin \alpha & S_y \sin \beta & S_z & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This involved writing the output data from the previous step to a file. This wasn't too difficult and I already had a thorough understanding of DXF from learning how to read from it.

LIST OF SUBS AND CUSTOM FUNCTIONS

In rough order of appearance in code. Not all listed.

PROCESS()

This routine contains is the whole process done. It runs the read file routine, the change coordinate system routine, the *proj* routine and then the write to file routine.

GETVP()

This routine calculates the 3D coordinates in the 3D model coordinate system of the vanishing point. This is based upon the defaults chosen that are relative to the extents of the object.

DXFREAD()

This reads a DXF file and assigns the relevant data to the respective variables. This is the main input file format, and had the most functionally, being able to correctly read and successively write layer data, points, lines, circles and arcs. This made it the superior format used and developed.

CIRCLTOLINES(...)

This subdivides a circle (given the centre in OCS, radius and normal vector), into many small lines

1 Flowchart symbols used in accordance with ANSI standards.



- 12. APPENDIX A - DEVELOPED SOFTWARE -

which act as a finite representation of the circle. This conversion to small lines is essential for circle to work, as when the final 2D drawing is created the original circles will no longer be circles, and often not even ellipses. This routine was the most difficult part to program as I had to write code that converted from OCS to WCS.

ARCTOLINES(...)

This is the same as the circletolines(...) sub routine however it omits an arc of the circle. In other words when the circle is split into many connected lines, the lines do not form a closed curve. As the originally object was an arc not a circle, parts must be omitted. This proved more difficult than originally planned, by the fact that I was only given a start and end angle in the DXF file, this had to be converted to x component parts that would be omitted.

TESTQ(...) (FUNCTION)

This was called in the arctocircle(...) sub routine. The function will return a true/false upon whether or not a given angle is in a given quadrant. This was written as a separate function as it was a few lines long and it was frequently called during the arctocircle(...) sub routine.

CSVREAD()

This routine opened a CSV (comma separated values) file and collected the relevant data and stored it in the respective variables. csvread() was one of the first read routines to be coded as it is the simplest. Also Rhino could export points in that file format easily. The limitation however is that it can only contain points, not lines, circles and arcs. So the csvread() routine became redundant over the much more powerful dxfread() routine, which was developed after. However CSV files are still supported by the software.

RAWREAD()

Again this routine was developed early and left in the software to offer more input file format options. It was initially useful as it provided a way to project 3D meshes, not just points. However due to the fact that meshes are usually extremely complex with many faces this input format was not the predominant one.

PLYREAD()

Similarly this routine read PLY mesh files, it was included as it was easy and fast to program into the program, however it had the same draw backs as RAW and is not the predominant file format.

DXFWRITE()

This wrote the 2D data to DXF format file. It also wrote the the lines and points to the same layer that it was imported from.

CSVWRITE()

Wrote the output data to CSV file format. This was easy to program, but had little functionality as it only saved points not lines.

COORDSYS()

This was included to allow the program to successfully take input from other coordinate systems. As this is a very mathematical topic, I will not comment on it, but I think that I should have left it out as I never needed to use it.

FRONTFACE()

This sub routine is done before proj(). It simply rotates the object to make the given face appear as the front. This proves useful as there is not default front face given in 3D programs, so this allows the user to choose which face they want to be the front face.



ROTATEAZEL()

This sub routine performs 3D rotations on the object so that it is oriented in a way that was specified by the projection mode chosen for 'Axonometric: Azimuth and Elevation'.

PROJ()

This routine did the actual projection. This is the backbone of the program. Everything else was written around and to support this. It was however very easy to program, it was the reading and converting the data that was difficult but without that data being read and processed and written, this routine would be useless. Once I had done the mathematics to find the general projection matrix that I needed, this routine was complete.

PERSPPROJ()

This routine does the same thing as proj() however it is for perspective projection. Perspective is one of the few types of projection that could not be obtained through the flexible projection matrix used in proj().

BTNDEFAULTAXONSCALES_CLICK()

This calculated the value to scale each axis given the alpha and beta values. This works to generate correct dimetric and trimetric drawings. This was needed because the alpha and beta angles can be vary for the drawing, however the scales that go with the given angles cannot vary, otherwise you do not have a dimetric or trimetric drawing. As the calculation of these scales involves a complex formula I built it into the program to make it easier to use and easier to get correct results.

BACKGROUNDWORKER1_DoWork()

This started the process on a separate thread to allow the user interface to remain responsive.

BACKGROUNDWORKER1_RUNWORKERCOMPLETED()

Once the whole process was complete the clean up of the "processing..." etc. was preformed. Also variables were reset so that the didn't interfere if the program ran again.

PICTORIAL DEVELOPMENT OF VIEWIT

A pictorial representation of the development of ViewIt is shown below with the main forms. The main form reflects the features and how the programmed changed through development.

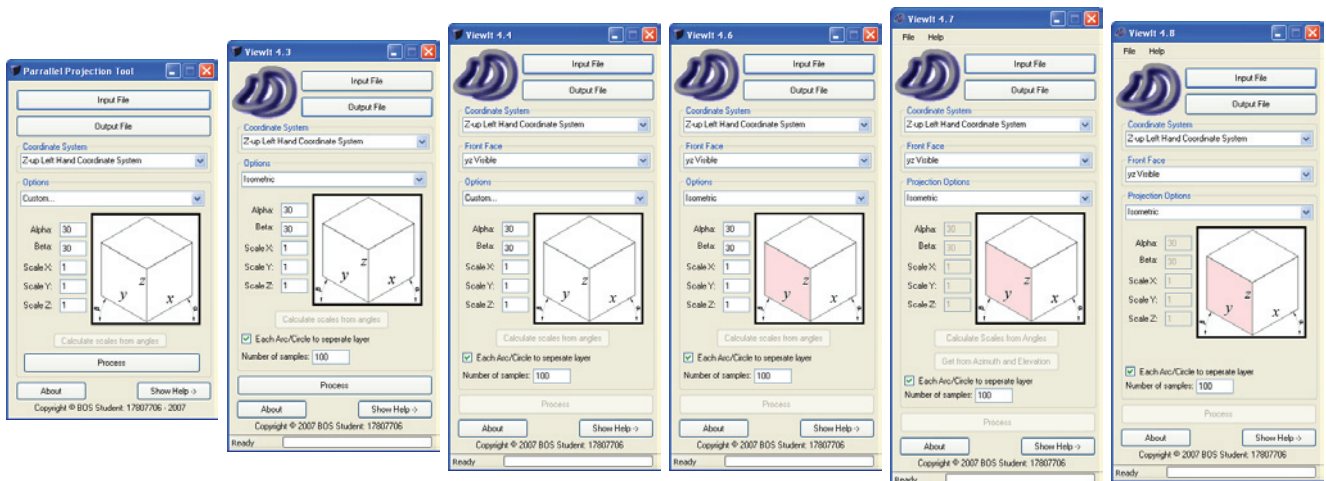


Figure 12.7- Pictorial Development of ViewIt

USER REFERENCE

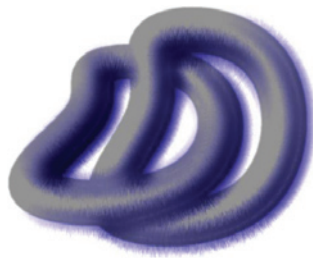


ViewIt 4.8 – User Reference

Version 4.8

BOS Student: 17807706

April 2007



Copyright © 2007 BOS Student: 17807706



Contents

Contents.....	2
1. Introduction	3
2. User Interface.....	4
3. Extended Step by Step Help.....	6
4. Drawing Projections	10
5. Version History.....	15
6. Technical Specifications.....	17

ViewIt 4.8 – User Reference | BOS Student: 17807706



2



1. Introduction

I began production of *ViewIt* in February 2007. I finalised release 4.8 in May 2007, which was the version I used to develop parts of my major project. However its use is not limited to this specific project. Modern architects, engineers and industrial designers no longer base their projects entirely in 2D. Modern software allows for the designing of products and structures in 3D. 3D CAD has many advantages, however we still need 2D drawings. However modern methods involve the modelling of a product or structure in 3D and then 2D drawings are created based on that 3D model. Many current 3D CAD/CAM software products incorporate features that automatically create 2D drawings from 3D data. Unfortunately in the software I have found, this is limited to orthogonal, isometric, and perspective views. Although these drawing projections are the most commonly used, the others are still important, and they appear in graphical communication and technical drawing text books. For instance if I want to create an oblique or a planometric drawing I should be able to have the computer software to generate it automatically, in the same way that orthogonal drawings are automated. Especially as the detail and the extent of curves used in modern products and structures is so great that manual methods would be long tedious and sometimes almost impossible.

ViewIt fills this gap that other software products have left. *ViewIt* takes 3D data from existing 3D CAD/CAM software, and generates 2D drawings from that 3D data. Although there are still many improvements that could be made to the product, it is proving promising so far. *ViewIt* can generate 2D drawings in cabinet oblique, cavalier oblique, planometric, dimetric, trimetric, isometric, one point perspective and orthographic projections. *ViewIt* can also generate custom projections (this is due to the way it was designed). In future releases of *ViewIt* I hope to extend the support to many CAD/CAM applications (currently only Rhinoceros® and AutoCAD® are fully supported), and I also hope to implement hidden detail features, which were not addressed in the current version due to time constraints and the importing architecture.



2. User Interface

Figure 1 - Splash Screen

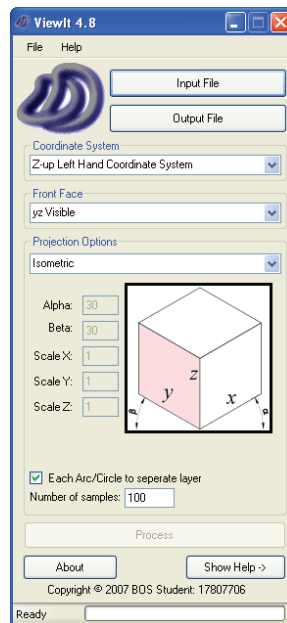


Figure 2 - Main User Interface



Figure 3 - About Box

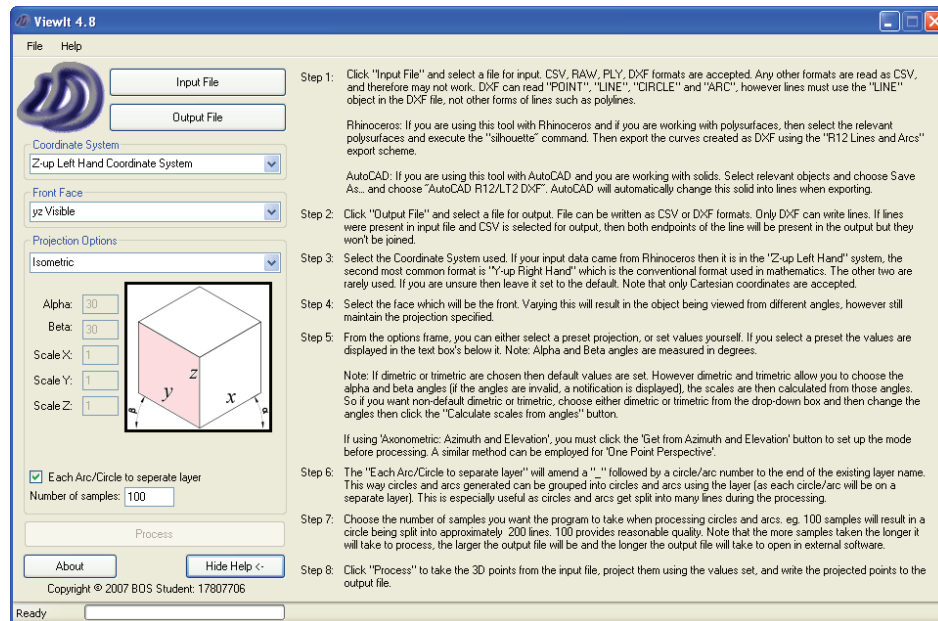


Figure 4 - User Interface with Help

ViewIt 4.8 - User Reference | BOS Student: 17807706



3. Extended Step by Step Help

Step 1:

Click "Input File" and select a file for input. CSV, RAW, PLY, DXF formats are accepted. Any other formats are read as CSV, and therefore may not work. DXF can read "POINT", "LINE", "CIRCLE" and "ARC", however lines must use the "LINE" object in the DXF file, not other forms of lines such as polylines.

Rhinoceros: If you are using this tool with Rhinoceros and if you are working with polysurfaces, then select the relevant polysurfaces and execute the "silhouette" command. Then export the curves created as DXF using the "R12 Lines and Arcs" export scheme.

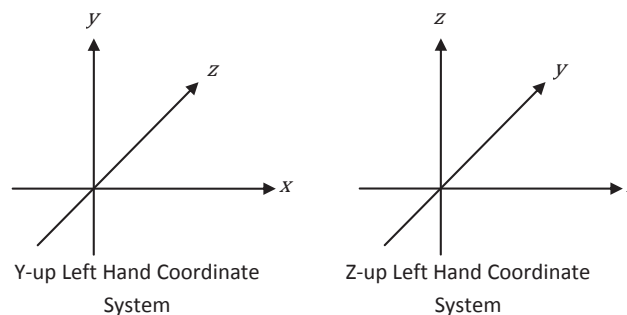
AutoCAD: If you are using this tool with AutoCAD and you are working with solids. Select relevant objects and choose Save As... and choose "AutoCAD R12/LT2 DXF". AutoCAD will automatically change this solid into lines when exporting.

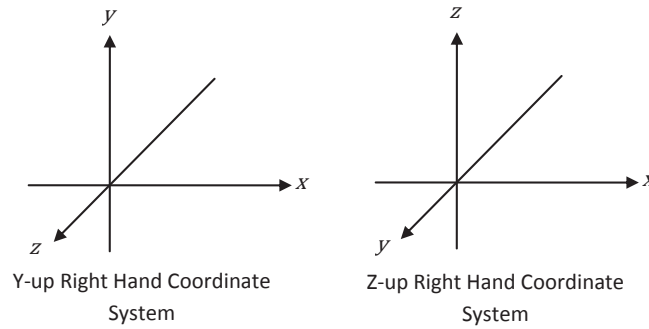
Step 2:

Click "Output File" and select a file for output. File can be written as CSV or DXF formats. Only DXF can write lines. If lines were present in input file and CSV is selected for output, then both endpoints of the line will be present in the output but they won't be joined.

Step 3:

Select the Coordinate System used. If your input data came from Rhinoceros then it is in the "Z-up Left Hand" system, the second most common format is "Y-up Right Hand" which is the conventional format used in mathematics. The other two are rarely used. If you are unsure then leave it set to the default. Note that only Cartesian coordinates are accepted.





Step 4:

Select the face which will be the front. Varying this will result in the object being viewed from different angles, however still maintain the projection specified. Altering this setting will rotate the object prior to the main calculations to make the chosen face at the default front.

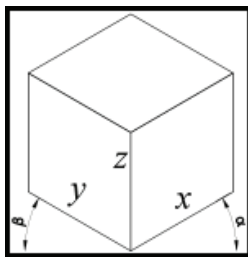


Figure 5 - Cube shown in UI

In the figure below a cube is shown in 3rd angle orthogonal and perspective.

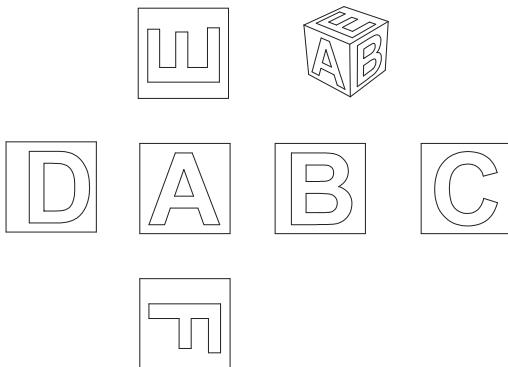


Figure 6 - Cube Orthogonal & Perspective

yz Visible – A is front
 xz Visible – B is front
 xz Hidden – D is front
 yz Hidden – C is front
 xy Visible – E is front (although E will be front, the rotation may appear incorrect)
 xy Hidden – F is front (although F will be front, the rotation may appear incorrect)



The use of "xy Visible" and "xy Hidden" is not recommended.

Step 5:

From the options frame, you can either select a preset projection, or set values yourself. If you select a preset the values are displayed in the text box's below it.

Note: Alpha and Beta angles are measured in degrees.

Note: If dimetric or trimetric are chosen then default values are set. However dimetric and trimetric allow you to choose the alpha and beta angles (if the angles are invalid, a notification is displayed), the scales are then calculated from those angles. So if you want non-default dimetric or trimetric, choose either dimetric or trimetric from the drop-down box and then change the angles then click the "Calculate scales from angles" button.

The scales for custom angles can be calculated with the following formula. However *ViewIt* will do this if you click on the "Calculate scales from angles" button. (Note: The projection must be set to either dimetric or trimetric for this button to be enabled.)

$$x \text{ scale} = \sqrt{\frac{\tan \beta (1 + \tan^2 \alpha)}{\tan \alpha + \tan \beta}} \quad y \text{ scale} = \sqrt{1 - \tan \alpha \times \tan \beta} \quad z \text{ scale} = \sqrt{\frac{\tan \alpha (1 + \tan^2 \beta)}{\tan \alpha + \tan \beta}}$$

There are certain conditions on the angles you can use for dimetric and trimetric drawings, so when using custom angles with dimetric and trimetric it is best practice to choose the angles then click the "Calculate scales from angles" button, which along with calculating the correct scales that go with the angles you choose, it also checks to make sure the angles you have chosen are valid.

If using 'Axonometric: Azimuth and Elevation', you must click the 'Get from Azimuth and Elevation' button to set up the mode before processing.

If using 'One Point Perspective' you can set the location of the vanishing point (VP). Depth = -VPz. The coordinate system used for the vanishing point is the same of that shown in the diagram. The picture plane (PP) is on the yz plane, ie. they are the same plane. To get some 'standard' perspective view use the 'Choose Default Perspective' button. This allows you to choose where the vanishing point is located relative to the extents of the object. However the depth is not set by using 'Choose Default Perspective'. You will need to set it yourself.

Step 6:

The "Each Arc/Circle to separate layer" will amend a "_" followed by a circle/arc number to the end of the existing layer name. This way circles and arcs generated can be grouped into circles and arcs using the layer (as each circle/arc will be on a separate layer). This is especially useful as circles and arcs get split into many lines during the processing.

Step 7:

Choose the number of samples you want the program to take when processing circles and arcs. eg. 100 samples will result in a circle being split into approximately 200 lines (double), however 100 provides reasonable quality. Note that the more samples taken the longer it will take to process, the larger the output file will be and the longer the output file will take to open in external software.



Step 8:

Click "Process" to take the 3D points from the input file, project them using the values set, and write the projected points to the output file.

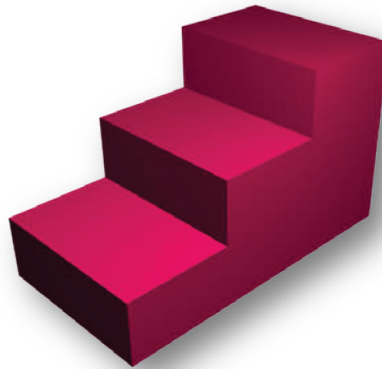
The progress bar at the bottom will show when the software is processing. It should be constantly moving.



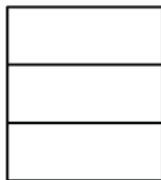
4. Drawing Projections

There is a range of drawing projections available in *ViewIt*. Many are not commonly used in industry and are therefore not supported well in other software solutions. That is partly why I developed this software so that if you wanted these less common drawing projections you could get them. Please note that many of the 3D CAD/CAM packages on the market can generate orthogonal, perspective and isometric drawings. Those functions built into the software are much more powerful than what *ViewIt* can do, use the features from the software you are working with first, only if that shall fail or is not applicable should *ViewIt* be used.

The following diagrams show the views generated from the pre-set projections. They are of the object shown in a rendered perspective view below. Note: The following diagrams are to scale. (Except the rendering)

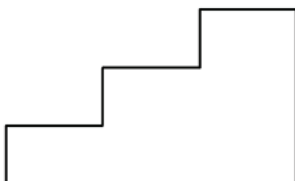


Front



$$\begin{aligned}\alpha &= 0^\circ \\ \beta &= 0^\circ \\ S_x &= 0 \\ S_y &= 1 \\ S_z &= 1\end{aligned}$$

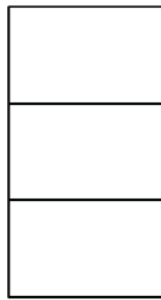
Right



$$\begin{aligned}\alpha &= 0^\circ \\ \beta &= 0^\circ \\ S_x &= 1 \\ S_y &= 0 \\ S_z &= 1\end{aligned}$$

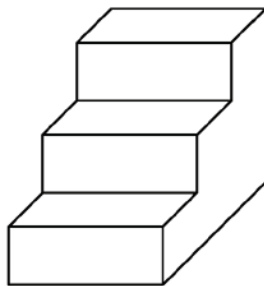


Top



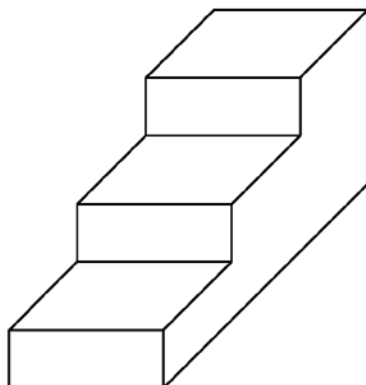
$$\begin{aligned}\alpha &= 90^\circ \\ \beta &= 0^\circ \\ S_x &= 1 \\ S_y &= 1 \\ S_z &= 0\end{aligned}$$

Cabinet Oblique



$$\begin{aligned}\alpha &= 45^\circ \\ \beta &= 0^\circ \\ S_x &= 0.5 \\ S_y &= 1 \\ S_z &= 1\end{aligned}$$

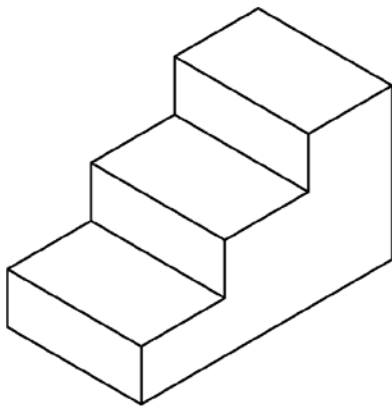
Cavalier Oblique



$$\begin{aligned}\alpha &= 45^\circ \\ \beta &= 0^\circ \\ S_x &= 1 \\ S_y &= 1 \\ S_z &= 1\end{aligned}$$

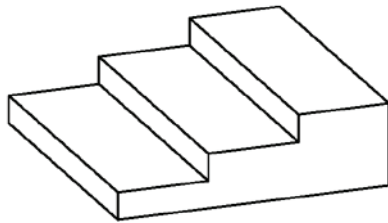


Isometric



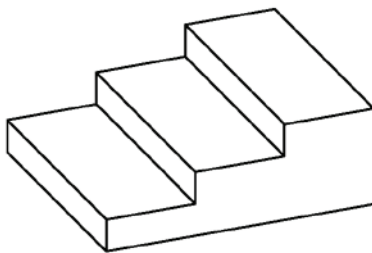
$$\begin{aligned}\alpha &= 30^\circ \\ \beta &= 30^\circ \\ S_x &= 1 \\ S_y &= 1 \\ S_z &= 1\end{aligned}$$

Dimetric



$$\begin{aligned}\alpha &\approx 7.18^\circ \\ \beta &\approx 41.41^\circ \\ S_x &\approx 0.94 \\ S_y &\approx 0.94 \\ S_z &\approx 0.47\end{aligned}$$

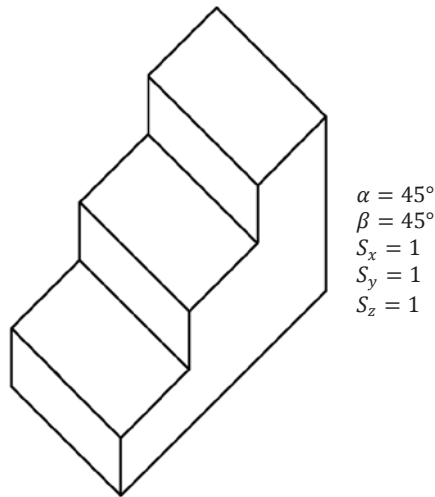
Trimetric



$$\begin{aligned}\alpha &= 10^\circ \\ \beta &= 45^\circ \\ S_x &\approx 0.94 \\ S_y &\approx 0.91 \\ S_z &\approx 0.55\end{aligned}$$

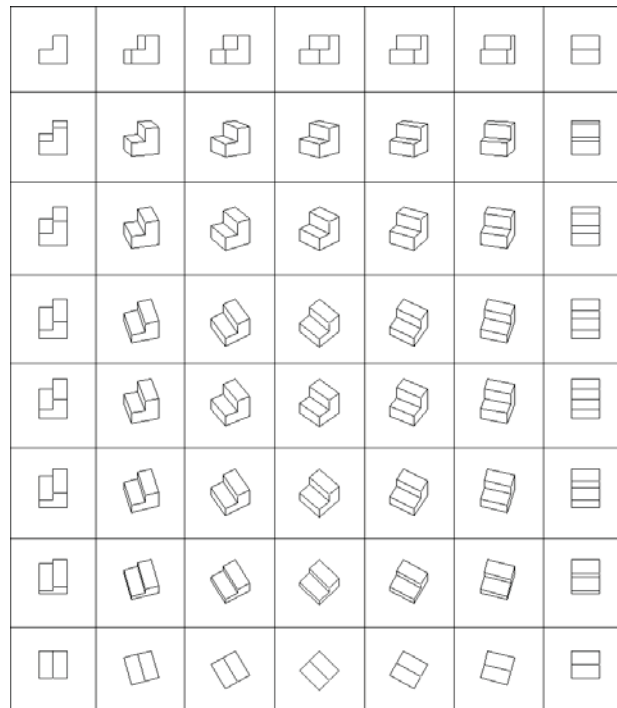


Planometric



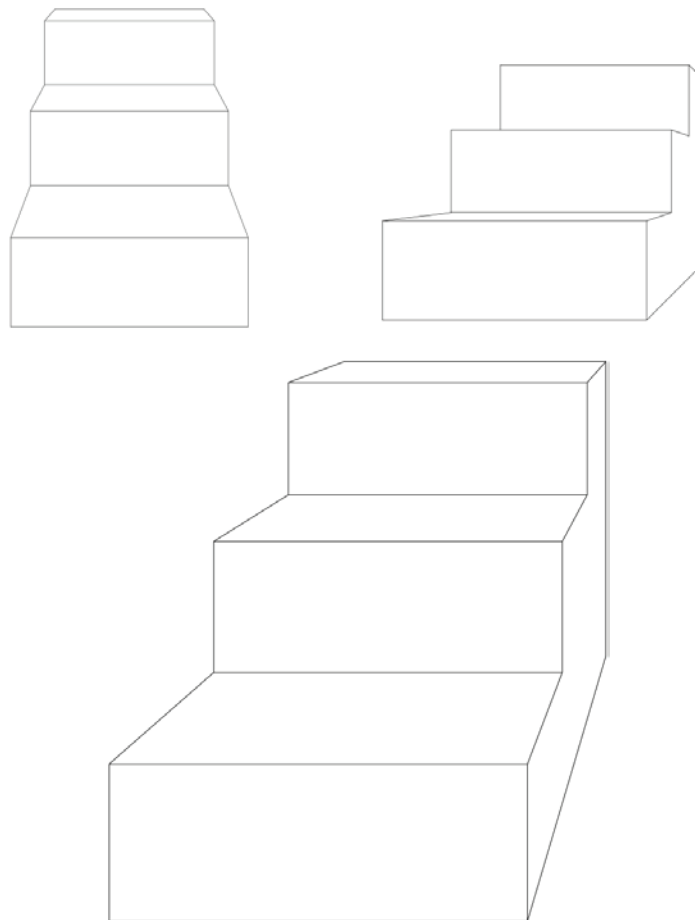
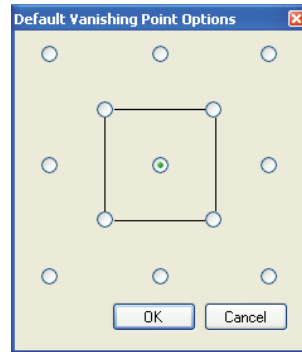
Axonometric: Azimuth and Elevation

This method allows a range of axonometric views to be generated. Unlike the other axonometric options available this method allows the user to choose the view based upon rotating the object and then projected orthogonally onto a projection plane lying on the yz plane. Because of this the alpha, beta and scale angles become obsolete. The azimuth and elevation angles can be specified or chosen from some defaults. The defaults contain some special cases which include isometric, orthogonal and a form of dimetric.



One Point Perspective

Different defaults can be chosen from the dialogue box. Where the option button represents the vanishing point and the square represents the extents of the object. The object is being looked at from the front face.



5. Version History

ViewIt 4.8

Released: May 2007

Updates:

- 'One Point Perspective' projection mode was implemented.

ViewIt 4.7

Released: April 2007

Updates:

- 'Axonometric: Azimuth and Elevation' projection mode was implemented.
- Menu bar was added.
- Small icon for application was changed to DD.

ViewIt 4.6

Released: April 2007

Updates:

- Front face bug was fixed.
- Front face is displayed graphically.
- DXF import routine was cleaned up causing less import errors and artefacts.

ViewIt 4.5

Released: April 2007

Updates:

- Bugs in Arcs were fixed. Extensive documentation was done.
- Circles and Arcs to separate layers functionality was added.

ViewIt 4.4

Released: March 2007

Updates:

- The functionality to select the face you want to appear as the front was added.

ViewIt 4.2

Released: January 2007

Updates:

- Circles could be imported.

ViewIt 4.0

Released: January 2007

Updates:

- Name was changed to ViewIt from Parallel Projection Tool.
- GUI was extended.
- Splash screen was added.

Parallel Projection Tool 3.0

Released: January 2007

Updates:

- Processing was run on a separate thread. Also processing bar was added.

ViewIt 4.8 – User Reference | BOS Student: 17807706



Parallel Projection Tool 2.0

Released: January 2007

Updates:

- Oblique Projection Tool was extended to support other drawing projections.
- Main projection algorithm derived and first implemented. Has not changed at all in future releases.

Oblique Projection Tool 1.0

Released: January 2007

Updates:

- Oblique Projection Tool was extended to support other drawing projections.
- Main projection algorithm derived and first implemented. Has not changed at all in future releases.



6. Technical Specifications

File Formats:

Input:

AutoCAD® Drawing Exchange File (.dxf)

Entities supported include: CIRCLE, ARC, LINE and POINT. AutoCAD R12 DXF format is the best format to use. Later versions may cause problems. If exporting from Rhinoceros® choose “R12 Lines and Arcs” and if exporting from AutoCAD choose “AutoCAD R12/LT2 DXF”. DXF is the best file format to use when using ViewIt.

Comma Separated Values (.csv)

Each line is read as a point. Coordinates are separated by a comma, and no quotes should be around the values. No other data except for the point should be included.

The following line would be read as a point with $x = 1$, $y = 2$ & $z = 3$.

1,2,3

Text Files (.txt)

Reads the same as CSV file. If exporting points from Rhinoceros® choose points file, comma separated, and NO quotes around values.

RAW Triangles (.raw)

Reads meshes, only the vertices are read, and are processed as points. This format has only been tested by exporting RAW from Rhinoceros®. (See comment below.)

Polygon File Format (.ply)

Similar to RAW Triangles, used for importing meshes, however only the vertices are read as points, no edges are read as lines. The reason for this is meshes are usually complex, and do not just have the edges that you would see in a technical drawing. So if you are bringing in data from Rhinoceros® it would be much better to use the method described in the step by step help and use DXF.

Output:

AutoCAD® Drawing Exchange File (.dxf)

This is the best export format, supporting points, lines and layers.

Text Files (.txt)

Only points are exported.

Comma Separated Values (.csv)

Only points are exported. Same as TXT.

Installation:

EXE File Size: 1.49MB (4.8.0.30)

Supported Operating Systems: Microsoft® Windows® XP, Microsoft® Windows® Vista™

Prerequisites: Microsoft .NET Framework 2.0 (If not installed, file will be downloaded from internet. To avoid this, install “dotnetfx.exe” found in installation archive prior to installing ViewIt)



VISUAL BASIC® 2005 MAIN FORM CODE

NOTE: The green text symbolises comments. They tell me and others information that is useful to understanding the program.

```
Public Class frmMain
    '////////////////////////////////////
    '/// This program takes 3D Coordinates and converts //
    '/// them into 2D coordinates using various      //
    '/// projections. Version: 4.8                    //
    '///                                              //
    '/// VB Version: VB2005                          //
    '/// Developed by BOS Student: 17807706           //
    '/// Produced for HSC IT-Graphics Major Project  //
    '/// Copyright © BOS Student: 17807706 - 2007    //
    '////////////////////////////////////

    #Region " Global Declarations"
        Dim icoord() As String 'array containing string input xyz coords. used for csv files

        Dim x(690060) As Single 'point or first line endpoint
        Dim y(690060) As Single
        Dim z(690060) As Single

        Dim x2(690060) As Single 'second line endpoint
        Dim y2(690060) As Single
        Dim z2(690060) As Single

        Dim ptnum As Integer 'number of points or lines read

        Dim xp(690060) As Single 'xp,yp are the 2D coordinates of the projected point
        Dim yp(690060) As Single

        Dim x2p(690060) As Single 'xp,yp are the 2D coordinates of the projected point
        Dim y2p(690060) As Single

        Dim alpha As Single 'projection variables, alpha & beta in DEGREES
        Dim beta As Single
        Dim Sx As Single
        Dim Sy As Single
        Dim Sz As Single

        Dim CoordSysCombo As String 'needed as can't access object due to multithreading
        Dim FrontFaceCombo As String 'needed as can't access object due to multithreading
        Dim projectionpreset As String 'needed as can't access object due to multithreading

        Dim textline As String 'string of text for a line in the input file
        Dim dxfhandle As Integer 'dxf object handle
        Dim layername(690060) As String 'name of layer so input layer is used for output
        Dim q As Integer 'temp for loop number
        Dim lines As Boolean = False 'whether lines or points are being projected. true=lines, false=points, a
        combination will result in lines=true and lines with the same endpoints will appear in output // this has been
        outdated as the program evolved, it is redundant (still functions though) but was never removed, it won't work when
        points come in from the file before lines do.
        Const pi As Single = 3.14159265358979

        Dim samples As Integer 'number of samples to take when processing circles

        Dim startangle As Double 'start angle for arcs, in DEGREES
        Dim endangle As Double 'end angle for arcs, in DEGREES

        Dim circlecount As Integer 'circle number relating to layer num assigned
        Dim layer As String 'layer name for circle, interim phase only, used for layername() array.

        Public azimuth As Single 'angle in DEGREES
        Public elevation As Single 'angle in DEGREES
        Dim azel_notready As Boolean = True

        Dim usedefaultpersp As Boolean = False 'when in perspective mode, whether to use the values set or the ones
        choosen from defaults
        Public defpersp As Integer 'different values mean where the vp is located in regards to the object
        '0 = not defined
        '1 = centre object
        '2 = top left
        '3 = top right
        '4 = bottom left
        '5 = bottom right
    End Region
End Class
```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```
'6 = top left extend half object
'7 = top right extend half object
'8 = bottom left extend half object
'9 = bottom right extend half object
'10 = top centre extend half object
'11 = bottom centre extend half object
'12 = left centre extend half object
'13 = right centre extend half object

Dim VPy As Single 'vanishing point y
Dim VPz As Single 'vanishing point z
Dim VPdepth As Single '-(vanishing point x)

Dim outclicked As Boolean = False 'so that button can be enabled when both input and output file are first
clicked
Dim inclicked As Boolean = False

Dim writefile As IO.StreamWriter 'write file object
Dim readfile As IO.StreamReader 'read file object
Dim readfilespec As String 'write file path
Dim writefilespec As String 'read file path

#End Region

Private Sub process()

    If CInt(txtSamples.Text) = 0 Then
        MsgBox("Please specify a non-zero number of samples.", MsgBoxStyle.Exclamation)
        BackgroundWorker1.CancelAsync()
        Exit Sub
    Else
        samples = CInt(txtSamples.Text)
    End If

    Try 'some files cannot be opened when already open in another application. this catches and notifies of
that error.
        readfile = IO.File.OpenText(readfilespec) 'open files specified in the dialogue box's.
    Catch E As Exception
        MsgBox("The input file could not be read: " & E.Message, MsgBoxStyle.Critical)
        BackgroundWorker1.CancelAsync() 'will still run BackgroundWorker1_RunWorkerCompleted
        Exit Sub
    End Try

    Try
        writefile = IO.File.CreateText(writefilespec)
    Catch E As Exception
        MsgBox("The output file could not be written: " & E.Message, MsgBoxStyle.Critical)
        BackgroundWorker1.CancelAsync()
        Exit Sub
    End Try

    'read input
    If readfilespec.Substring(readfilespec.Length - 3).ToLower = "raw" Then
        rawread()
    ElseIf readfilespec.Substring(readfilespec.Length - 3).ToLower = "csv" Then
        csvread()
    ElseIf readfilespec.Substring(readfilespec.Length - 3).ToLower = "ply" Then
        plyread()
    ElseIf readfilespec.Substring(readfilespec.Length - 3).ToLower = "dxf" Then
        dxfread()
    Else
        csvread() 'any other types will be read as csv, if they are not csv format error may occur. txt
Points files from rhino should work.
    End If

    coordsys()
    frontface()
    If projectionpreset = "Axonometric: Azimuth and Elevation" Then
        rotateazel()
    End If

    If projectionpreset = "One Point Perspective" Then 'we need to use a different matrix for perspective
projection
        getVP()
        perspproj(ptnum)
```



```

Else
    proj(ptnum) `apply the projection matrix
End If

`write output
If writefilespec.Substring(writefilespec.Length - 3).ToLower = "dxf" Then
    dxfwrite()
ElseIf writefilespec.Substring(writefilespec.Length - 3).ToLower = "csv" Then
    csvwrite()
Else
    csvwrite()
End If

readfile.Close() `closes the files
writefile.Close()
End Sub

Private Sub getVP()
    If usedefaultpersp = True Then

        Dim yextentpos As Single
        Dim zextentpos As Single

        Dim yextentneg As Single
        Dim zextentneg As Single

        yextentpos = y(1)
        zextentpos = z(1)

        yextentneg = y(1)
        zextentneg = z(1)

        `find the extents of the model. largest number will be returned in yextentpos and zextentpos, and the
        smallest in the respective neg
        For q = 1 To ptnum
            If y(q) > yextentpos Then yextentpos = y(q)
            If z(q) > zextentpos Then zextentpos = z(q)

            If y2(q) > yextentpos Then yextentpos = y2(q)
            If z2(q) > zextentpos Then zextentpos = z2(q)

            If y(q) < yextentneg Then yextentneg = y(q)
            If z(q) < zextentneg Then zextentneg = z(q)

            If y2(q) < yextentneg Then yextentneg = y2(q)
            If z2(q) < zextentneg Then zextentneg = z2(q)
        Next

        Select Case defpersp
            Case Is = 0 `not supposed to happen, if defpersp = 0 then usedefaultpersp should = false
                Debug.Print("Chosen to use perspective default values but defpersp is set to zero.")
                MsgBox("Error in calculating Vanishing Point. This is possible a bug in the software.")

            Case Is = 1 `centre object
                VPy = (yextentpos + yextentneg) / 2
                VPz = (zextentpos + zextentneg) / 2

            Case Is = 2 `top left
                VPy = yextentpos
                VPz = zextentpos

            Case Is = 3 `top right
                VPy = yextentneg
                VPz = zextentpos

            Case Is = 4 `bottom left
                VPy = yextentpos
                VPz = zextentneg

            Case Is = 5 `bottom right
                VPy = yextentneg
                VPz = zextentneg

            Case Is = 6 `top left extend half object

```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```

object
    VPy = yextentpos + (Math.Sqrt(yextentpos ^ 2 + yextentneg ^ 2) / 2) 'left pos + half of
    VPz = zextentpos + (Math.Sqrt(zextentpos ^ 2 + zextentneg ^ 2) / 2)

    Case Is = 7 'top right extend half object
        VPy = yextentneg - (Math.Sqrt(yextentpos ^ 2 + yextentneg ^ 2) / 2)
        VPz = zextentpos + (Math.Sqrt(zextentpos ^ 2 + zextentneg ^ 2) / 2)

    Case Is = 8 'bottom left extend half object
        VPy = yextentpos + (Math.Sqrt(yextentpos ^ 2 + yextentneg ^ 2) / 2)
        VPz = zextentneg - (Math.Sqrt(zextentpos ^ 2 + zextentneg ^ 2) / 2)

    Case Is = 9 'bottom right extend half object
        VPy = yextentneg - (Math.Sqrt(yextentpos ^ 2 + yextentneg ^ 2) / 2)
        VPz = zextentneg - (Math.Sqrt(zextentpos ^ 2 + zextentneg ^ 2) / 2)

    Case Is = 10 'top centre extend half object
        VPy = (yextentpos + yextentneg) / 2
        VPz = zextentpos + (Math.Sqrt(zextentpos ^ 2 + zextentneg ^ 2) / 2)

    Case Is = 11 'bottom centre extend half object
        VPy = (yextentpos + yextentneg) / 2
        VPz = zextentneg - (Math.Sqrt(zextentpos ^ 2 + zextentneg ^ 2) / 2)

    Case Is = 12 'left centre extend half object
        VPy = zextentpos + (Math.Sqrt(zextentpos ^ 2 + zextentneg ^ 2) / 2)
        VPz = (zextentpos + zextentneg) / 2

    Case Is = 13 'right centre extend half object
        VPy = zextentneg - (Math.Sqrt(zextentpos ^ 2 + zextentneg ^ 2) / 2)
        VPz = (zextentpos + zextentneg) / 2

    Case Else 'again this should not happen.
        Debug.Print("Internal Program Error. The choosen default does not match with the defaults.")
        MsgBox("Error in default perspective. This is a bug in the software.")
End Select

Else 'false, therefore use values in text box's
    'values from text box are already in memory. ie,

    'VPy = VPy
    'VPz = VPz

End If
End Sub

Private Sub dxhread() 'in revision 4.6.0.18 this sub routine was cleaned up, previously it relied on
specific order, now data can come in any order.
    Dim code As String 'dxf code, eg. x coordinate code for POINT = 10
    Dim value As String 'dxf value, eg. for layer, LAYERNAME
    Dim buffer As String 'file line buffer

    ptnum = 0

    buffer = CStr(readfile.ReadLine)

    'opens dxf files. only set up to read LINE and POINT entries
    'When saving dxf file in Rhino choose "R12 Lines & Arcs"

    Do 'chews through the header and other junk at start of file
        buffer = CStr(readfile.ReadLine)
        If (readfile.Peek = -1) Then 'if no lines or points exist in whole dxf file
            MsgBox("Warning no ARC, CIRCLE, POINT or LINE objects were found in DXF input file.",
MsgBoxStyle.Critical)
            BackgroundWorker1.CancelAsync()
            Exit Sub
        End If
    Loop Until buffer = "LINE" Or buffer = "POINT" Or buffer = "CIRCLE" Or buffer = "ARC"

    Do While readfile.Peek <> -1 'keep reading data until End Of File, if End of Section then will also
terminate loop as seen below.
        Select Case buffer

            '-----LINE-----
            '-----LINE-----
            Case Is = "LINE" 'LINE object found

```



```

LINE:
    lines = True

    code = readfile.ReadLine()
    value = readfile.ReadLine()

    Select Case code
        Case Is = " 10" 'X1
            x(ptnum) = CSng(value)
            GoTo LINE

        Case Is = " 20" 'Y1
            y(ptnum) = CSng(value)
            GoTo LINE

        Case Is = " 30" 'Z1
            z(ptnum) = CSng(value)
            GoTo LINE

        Case Is = " 11" 'X2
            x2(ptnum) = CSng(value)
            GoTo LINE

        Case Is = " 21" 'Y2
            y2(ptnum) = CSng(value)
            GoTo LINE

        Case Is = " 31" 'Z2
            z2(ptnum) = CSng(value)
            GoTo LINE

        Case Is = " 8" 'layer
            ptnum += 1
            layername(ptnum) = CStr(value)
            GoTo LINE

        Case Is = " 0" 'end of entity, next entity began
            Select Case value 'continue importing
                Case Is = "LINE"
                    GoTo LINE
                Case Is = "POINT"
                    GoTo POINT
                Case Is = "CIRCLE"
                    GoTo CIRCLE
                Case Is = "ARC"
                    GoTo ARC
                Case Is = "ENDSEC" 'end of section, no more data.
                    Exit Sub
                Case Else
                    GoTo genread
            End Select

        Case Is = " 5" 'handle
            'don't need this data. new dxf handle is created upon writedxf()
            GoTo LINE

        Case Else 'if code is unreconised skip it
            Debug.Print("Unrecognised DXF code in known ENTITY. Code: " & code & " | Entity:
LINE")

            GoTo LINE

    End Select

    '-----POINT-----
    '-----POINT-----
Case Is = "POINT"

POINT:
    code = readfile.ReadLine()
    value = readfile.ReadLine()

    Select Case code
        Case Is = " 10" 'X
            x(ptnum) = CSng(value)

```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

If lines = True Then x2(ptnum) = x(ptnum) 'output cannot work with lines and points as it doesn't know if the ptnum is a line or point, so if both are present the points will be lines with endpoints at the same point

```
GoTo POINT

Case Is = " 20" 'Y
y(ptnum) = CSng(value)
If lines = True Then y2(ptnum) = y(ptnum)
GoTo POINT

Case Is = " 30" 'Z
z(ptnum) = CSng(value)
If lines = True Then z2(ptnum) = z(ptnum)
GoTo POINT

Case Is = " 8" 'layer
ptnum += 1
layername(ptnum) = CStr(value)
GoTo POINT

Case Is = " 0" 'end of entity, next entity began
Select Case value 'continue importing
    Case Is = "LINE"
        GoTo LINE
    Case Is = "POINT"
        GoTo POINT
    Case Is = "CIRCLE"
        GoTo CIRCLE
    Case Is = "ARC"
        GoTo ARC
    Case Is = "ENDSEC" 'end of section, no more data.
        Exit Sub
    Case Else
        GoTo genread
End Select

Case Is = " 5" 'handle
'don't need this data. new dxf handle is created upon writedxf()
GoTo POINT

Case Else 'if code is unreconised skip it
Debug.Print("Unrecognised DXF code in known ENTITY. Code: " & code & " | Entity:
POINT")

GoTo POINT

End Select

'-----CIRCLE-----
'-----CIRCLE-----
Case Is = "CIRCLE"

CIRCLE:
    lines = True
    Dim cx, cy, cz, cr, cx2, cy2, cz2 As Single
    Dim vectorpresent As Boolean
    vectorpresent = False

circleread:
    code = readfile.ReadLine()
    value = readfile.ReadLine()
    Select Case code
        Case Is = " 8" 'layer
            layer = CStr(value)
            circlecount += 1
            GoTo circleread

        Case Is = " 10" 'centre X
            cx = CSng(value)
            GoTo circleread

        Case Is = " 20" 'centre Y
            cy = CSng(value)
            GoTo circleread

        Case Is = " 30" 'centre Z
            cz = CSng(value)
            GoTo circleread
```



```

Case Is = " 40" `radius
    cr = CSng(value)
    GoTo circleread

Case Is = "210" `vector data X
    vectorpresent = True
    cx2 = CSng(value)
    GoTo circleread

Case Is = "220" `vector data Y
    cy2 = CSng(value)
    GoTo circleread

Case Is = "230" `vector data Z
    cz2 = CSng(value)
    GoTo circleread

Case Is = " 0" `end of entity, next entity began
    `as vector data is optional, if it was not present fill variables with default
    If vectorpresent = False Then `fill with default, else don't worry already filled
        cx2 = CSng(0)
        cy2 = CSng(0)
        cz2 = CSng(1)
    End If

    `end of entity so send away data to be converted to lines
    circletolines(cx, cy, cz, cr, cx2, cy2, cz2, layer)

    Select Case value `continue importing
        Case Is = "LINE"
            GoTo LINE
        Case Is = "POINT"
            GoTo POINT
        Case Is = "CIRCLE"
            GoTo CIRCLE
        Case Is = "ARC"
            GoTo ARC
        Case Is = "ENDSEC" `end of section, no more data.
            Exit Sub
        Case Else
            GoTo genread
    End Select

Case Is = " 5" `handle
    `don't need this data. new dxf handle is created upon writedxf()
    GoTo circleread

Case Else `if code is unreconised skip it
    Debug.Print("Unrecognised DXF code in known ENTITY. Code: " & code & " | Entity:
CIRCLE")

    GoTo circleread

End Select

`-----ARC-----
`-----ARC-----
Case Is = "ARC"

ARC:
    lines = True
    Dim cx, cy, cz, cr, cx2, cy2, cz2 As Single
    Dim vectorpresent As Boolean
    vectorpresent = False

arcread:

    code = readfile.ReadLine()
    value = readfile.ReadLine()
    Select Case code
        Case Is = " 8" `layer
            layer = CStr(value)
            circlecount += 1
            GoTo arcread

        Case Is = " 10" `centre X
            cx = CSng(value)
            GoTo arcread

        Case Is = " 20" `centre Y

```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```
        cy = CSng(value)
        GoTo arcread

    Case Is = " 30" 'centre Z
        cz = CSng(value)
        GoTo arcread

    Case Is = " 40" 'radius
        cr = CSng(value)
        GoTo arcread

    Case Is = " 50" 'start angle
        startangle = CDBl(value)
        startangle = startangle / 180 * pi 'angles in file are in degrees, program works with
radians so they are changed
        GoTo arcread

    Case Is = " 51" 'end angle
        endangle = CDBl(value)
        endangle = endangle / 180 * pi 'angles in file are in degrees, program works with
radians so they are changed
        GoTo arcread

    Case Is = "210" 'vector data X
        vectorpresent = True
        cx2 = CSng(value)
        GoTo arcread

    Case Is = "220" 'vector data Y
        cy2 = CSng(value)
        GoTo arcread

    Case Is = "230" 'vector data Z
        cz2 = CSng(value)
        GoTo arcread

    Case Is = " 0" 'end of entity, next entity began
        'as vector data is optional, if it was not present fill variables with default
        If vectorpresent = False Then 'fill with default, else don't worry already filled
            cx2 = CSng(0)
            cy2 = CSng(0)
            cz2 = CSng(1)
        End If

        'end of entity so send away data to be converted to lines
        arctolines(cx, cy, cz, cr, cx2, cy2, cz2, startangle, endangle, layer)

    Select Case value 'continue importing
        Case Is = "LINE"
            GoTo LINE
        Case Is = "POINT"
            GoTo POINT
        Case Is = "CIRCLE"
            GoTo CIRCLE
        Case Is = "ARC"
            GoTo ARC
        Case Is = "ENDSEC" 'end of section, no more data.
            Exit Sub
        Case Else
            GoTo genread
    End Select

    Case Is = " 5" 'handle
        'don't need this data. new dxf handle is created upon writedxf()
        'this case is not needed, if omitted will just go to case else. this was put in to
avoid massive amounts of debug.prints, as all objects have this "handle"
        GoTo arcread

    Case Else 'if code is unreconised skip it
        Debug.Print("Unrecognised DXF code in known ENTITY. Code: " & code & " | Entity:
ARC")

        GoTo arcread

End Select

Case Is = "ENDSEC" 'end of section, no more data.
```



```

        Exit Sub
    End Select
genread:
    buffer = CStr(readfile.ReadLine)
Loop
End Sub

Private Sub circletolines(ByVal cx As Single, ByVal cy As Single, ByVal cz As Single, ByVal cr As Single,
ByVal cx2 As Single, ByVal cy2 As Single, ByVal cz2 As Single, ByVal layer As String)
    Dim u As Single 'temp for single type loop number
    Dim firstptnum As Integer
    Dim lastptnum As Integer
    Dim rtheta As Single
    Dim rphi As Single

    Dim tx(690060) As Single
    Dim ty(690060) As Single
    Dim tz(690060) As Single
    Dim tx2(690060) As Single
    Dim ty2(690060) As Single
    Dim tz2(690060) As Single

    Dim tcx, tcy, tcz As Single

    'about y -up
    'about z +anticlock

    'calualate rotation angles
    'div zero error check
    If cx2 <> 0 Then
        rphi = -Math.Atan(cz2 / cx2) 'negative because of rotation direction
    Else 'cz = 0 -> vector will point on xy plane
        If cz2 > 0 Then rphi = -(pi / 2)
        If cz2 < 0 Then rphi = pi / 2
        If cz2 = 0 Then rphi = 0
    End If
    rphi = rphi + (pi / 2) 'add 90 deg

    If cx2 <> 0 Then
        rtheta = Math.Atan(cy2 / cx2)
    Else
        If cy2 > 0 Then rtheta = pi / 2
        If cy2 < 0 Then rtheta = -(pi / 2)
        If cy2 = 0 Then rtheta = 0
    End If

    '-----START ARBITRARY AXIS ALGORITHM-----
    'circles with normals 0,0,1 have their centres in WCS therefore should remain unchanged.
    'arbitrary axis can be calculated from the normal vector

    'cx is the OCS in the file
    'x is the answer in WCS
    'cx2 = arbitrary Z axis
    Dim vectorlength As Single
    Dim AXx, AXy, AXz, AYx, AYy, AYz, AZx, AZy, AZz As Single

    'calculate arbitrary X axis
    If Math.Abs(cx2) < (1 / 64) And Math.Abs(cy2) < (1 / 64) Then
        AXx = cz2 'gets vector for arbitrary x axis
        AXy = 0
        AXz = -cx2
    Else
        AXx = -cy2
        AXy = cx2
        AXz = 0
    End If

    'scale vector to r = 1
    vectorlength = Math.Sqrt(AXx ^ 2 + AXy ^ 2 + AXz ^ 2)
    AXx = AXx / vectorlength
    AXy = AXy / vectorlength
    AXz = AXz / vectorlength

    'calculate arbitrary Y axis
    AYx = cx2 * AXz - AXy * cz2
    AYy = cz2 * AXx - cx2 * AXz
    AYz = AXy * cx2 - cy2 * AXx

```



```

'scale vector to r = 1
vectorlength = Math.Sqrt(AYx ^ 2 + AYy ^ 2 + AYz ^ 2)
AYx = AYx / vectorlength
AYy = AYy / vectorlength
AYz = AYz / vectorlength

'arbitrary Z axis is cx2,cy2,cz2
AZx = cx2
AZy = cy2
AZz = cz2

'arbitrary origin is WCS 0,0,0

'set temps
tcx = cx
tcy = cy
tcz = cz

'calculates WCS from OCS [from arbitrary axis previously calculated]

'[ x y z ] matrix is unknown, rest are known
'
'          AXx AYx AZx
'[ cx cy cz ] = [ x y z ] x  AXy AYy AZy
'          AXz AYz AZz

'results from solving matrix above
cx = tcx * AXx + tcy * AYx + tcz * AZx
cy = tcx * AXy + tcy * AYy + tcz * AZy
cz = tcx * AXz + tcy * AYz + tcz * AZz

'-----END ARBITRARY AXIS ALGORITHM-----

firstptnum = ptnum + 1
'double for num of lines from circle
For u = -cr To cr - (1 / samples) Step 1 / samples 'split top half of circle
    ptnum += 1

    x(ptnum) = u + cx
    y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
    z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
    x2(ptnum) = (u + 1 / samples) + cx
    y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
    z2(ptnum) = cz
Next

For u = -cr To cr - (1 / samples) Step 1 / samples 'split bottom half of circle
    ptnum += 1

    x(ptnum) = u + cx
    y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
    z(ptnum) = cz
    x2(ptnum) = (u + 1 / samples) + cx
    y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
    z2(ptnum) = cz
Next
lastptnum = ptnum

For u = firstptnum To lastptnum
    If chkArcLayer.Checked = True Then
        layername(u) = layer & "_" & circlecount
    Else
        layername(u) = layer 'assigns the circle layer to the layer of the line in the circle
    End If
Next

'transform coordinates to get origin at centre of circle //must repeate opposite after rotation
For u = firstptnum To lastptnum
    x(u) = x(u) - cx
    y(u) = y(u) - cy
    z(u) = z(u) - cz

    x2(u) = x2(u) - cx

```



```
y2(u) = y2(u) - cy
z2(u) = z2(u) - cz
```

Next

For q = firstptnum To lastptnum 'now apply local coordinate system rotation matrix to match normal vector in DXF file

```
'set up temp's (as x(q)... is reset after x= it is wrong in the z=)
tx(q) = x(q)
ty(q) = y(q)
tz(q) = z(q)

tx2(q) = x2(q)
ty2(q) = y2(q)
tz2(q) = z2(q)

'must rotate phi about y then theta about z

'rotate phi about y [equations derived from matrix form]
x(q) = tx(q) * Math.Cos(rphi) + tz(q) * Math.Sin(rphi) 'do for start and end point
x2(q) = tx2(q) * Math.Cos(rphi) + tz2(q) * Math.Sin(rphi)
'y stays the same
z(q) = -tx(q) * Math.Sin(rphi) + tz(q) * Math.Cos(rphi)
z2(q) = -tx2(q) * Math.Sin(rphi) + tz2(q) * Math.Cos(rphi)

'reset temp's
tx(q) = x(q)
ty(q) = y(q)
tz(q) = z(q)

tx2(q) = x2(q)
ty2(q) = y2(q)
tz2(q) = z2(q)

'rotate theta about z [equations derived from matrix form]
x(q) = tx(q) * Math.Cos(rtheta) - ty(q) * Math.Sin(rtheta)
x2(q) = tx2(q) * Math.Cos(rtheta) - ty2(q) * Math.Sin(rtheta)
y(q) = tx(q) * Math.Sin(rtheta) + ty(q) * Math.Cos(rtheta)
y2(q) = tx2(q) * Math.Sin(rtheta) + ty2(q) * Math.Cos(rtheta)
'z stays the same
```

Next

For u = firstptnum To lastptnum 'send back to world coord system [translation]

```
x(u) = x(u) + cx
y(u) = y(u) + cy
z(u) = z(u) + cz

x2(u) = x2(u) + cx
y2(u) = y2(u) + cy
z2(u) = z2(u) + cz
```

Next

End Sub

Private Sub arctolines(ByVal cx As Single, ByVal cy As Single, ByVal cz As Single, ByVal cr As Single, ByVal cx2 As Single, ByVal cy2 As Single, ByVal cz2 As Single, ByVal startangle As Double, ByVal endangle As Double, ByVal layer As String)

'the following code was obtained from the circletolines function, make any non-arc changes there and then copy them to here. This is not the best programming solution, however it is only repeated once and it was faster and easier for me to do it this way.

```
Dim u As Single 'temp for single type loop number
Dim firstptnum As Integer
Dim lastptnum As Integer
Dim rtheta As Single
Dim rphi As Single
```

```
Dim tx(690060) As Single
Dim ty(690060) As Single
Dim tz(690060) As Single
Dim tx2(690060) As Single
Dim ty2(690060) As Single
Dim tz2(690060) As Single
```

```
Dim tcx, tcy, tcz As Single
```

```
'about y -up
'about z +anticlock
```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```
`calualate rotation angles
`div zero error check
If cx2 <> 0 Then
    `rphi = Math.Atan(cz2 / Math.Sqrt((cx2) ^ 2 + (cy2) ^ 2)) `convert the direction vector from DXF
circle to 2 angles
    rphi = -Math.Atan(cz2 / cx2) `negative because of rotation direction
Else `cz = 0 -> vector will point on xy plane
    If cz2 > 0 Then rphi = -(pi / 2)
    If cz2 < 0 Then rphi = pi / 2
    If cz2 = 0 Then rphi = 0
    `rphi = -(pi / 2) `radians (-90degrees)
End If
rphi = rphi + (pi / 2) `add 90 deg

If cx2 <> 0 Then
    rtheta = Math.Atan(cy2 / cx2)
Else
    If cy2 > 0 Then rtheta = pi / 2
    If cy2 < 0 Then rtheta = -(pi / 2)
    If cy2 = 0 Then rtheta = 0
    `rtheta = -(pi / 2) `radians (-90degrees)
End If

`-----START ARBITRARY AXIS ALGORITHM-----
`circles with normals 0,0,1 have their centres in WCS therefore should remain unchanged.
`arbitrary axis can be calculated from the normal vector

`cx is the OCS in the file
`x is the answer in WCS
`cx2 = arbitrary Z axis

Dim vectorlength As Single
Dim AXx, AXy, AXz, AYx, AYy, AYz, AZx, AZy, AZz As Single

`calculate arbitrary X axis
If Math.Abs(cx2) < (1 / 64) And Math.Abs(cy2) < (1 / 64) Then
    AXx = cz2 `gets vector for arbitrary x axis
    AXy = 0
    AXz = -cx2
Else
    AXx = -cy2
    AXy = cx2
    AXz = 0
End If

`scale vector to r = 1 [this was done by getting theta and phi, then getting xyz using r = 1]
vectorlength = Math.Sqrt(AXx ^ 2 + AXy ^ 2 + AXz ^ 2)
AXx = AXx / vectorlength
AXy = AXy / vectorlength
AXz = AXz / vectorlength

`calculate arbitrary Y axis
AYx = cx2 * AXz - AXy * cz2
AYy = cz2 * AXx - cx2 * AXz
AYz = AXy * cx2 - cy2 * AXx

`scale vector to r = 1 [this was done by getting theta and phi, then getting xyz using r = 1]
vectorlength = Math.Sqrt(AYx ^ 2 + AYy ^ 2 + AYz ^ 2)
AYx = AYx / vectorlength
AYy = AYy / vectorlength
AYz = AYz / vectorlength

`arbitrary Z axis is cx2,cy2,cz2
AZx = cx2
AZy = cy2
AZz = cz2

`arbitrary origin is WCS 0,0,0

`set temps
tcx = cx
tcy = cy
tcz = cz
```



```

'calculates WCS from OCS [from arbitrary axis previously calculated]

'[ x y z ] matrix is unknown, rest are known

'
'          AXx AYx AZx
'[ cx cy cz ] = [ x y z ] * AXy AYy AZy
'          AXz AYz AZz

'results from soving matrix above
cx = tcx * AXx + tcy * AYx + tcz * AZx
cy = tcx * AXy + tcy * AYy + tcz * AZy
cz = tcx * AXz + tcy * AYz + tcz * AZz

'-----END ARBITRARY AXIS ALGORITHM-----

lines
'split the arc into many lines > more extensive than circle as the whole circle is not converted to
'so it is split into quadrants and delt with seperately
firstptnum = ptnum + 1

'logic tests follow
'in loops the to should be followed with "- (1 / samples)" this is done because this is the start of the
line and (1 / samples) gets added to find the endpoint. without the negative in the to statement, there would be
no real solution when the x value is substitiured thus an error will occur at the last endpoint.

'QUADRANT 1 -----
If testQ(startangle, 1) = False And testQ(endangle, 1) = False Then 'Neither angles
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q1_ef_sf.jpg
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q1.jpg
If startangle > endangle Then 'fill all
    For u = 0 To cr - (1 / samples) Step 1 / samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next
    'else fill none
End If

ElseIf testQ(startangle, 1) = True And testQ(endangle, 1) = False Then 'start angle only - Fill part
after startangle
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q1_ef_st.jpg
For u = 0 To cr * Math.Cos(startangle) - (1 / samples) Step 1 / samples
    ptnum += 1

    x(ptnum) = u + cx
    y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
    z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
    x2(ptnum) = (u + 1 / samples) + cx
    y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
    z2(ptnum) = cz
Next

ElseIf testQ(startangle, 1) = False And testQ(endangle, 1) = True Then 'end angle only - fill to right of
end angle till 0deg
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q1_et_sf.jpg
For u = 0 + cr * Math.Cos(endangle) To cr - (1 / samples) Step 1 / samples
    ptnum += 1

    x(ptnum) = u + cx
    y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
    z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
    x2(ptnum) = (u + 1 / samples) + cx
    y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
    z2(ptnum) = cz
Next

ElseIf testQ(startangle, 1) = True And testQ(endangle, 1) = True Then 'both angles in Quadrant
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q1_et_st.jpg
If startangle > endangle Then 'outer, exclude inner.
    For u = 0 To cr * Math.Cos(startangle) - (1 / samples) Step 1 / samples
        ptnum += 1

```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```

x(ptnum) = u + cx
y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
x2(ptnum) = (u + 1 / samples) + cx
y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
z2(ptnum) = cz
Next
For u = cr * Math.Cos(endangle) - (1 / samples) To cr - (1 / samples) Step 1 / samples
    ptnum += 1

    x(ptnum) = u + cx
    y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
    z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
    x2(ptnum) = (u + 1 / samples) + cx
    y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
    z2(ptnum) = cz
Next
Else
    For u = 0 + cr * Math.Cos(endangle) To cr * Math.Cos(startangle) - (1 / samples) Step 1 /
samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next
End If
End If

'QUADRANT 2 -----
If testQ(startangle, 2) = False And testQ(endangle, 2) = False Then 'neither in Q2
    'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q2_ef_sf.jpg
    'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q2.jpg
    If testQ(startangle, 1) And (testQ(endangle, 3) Or testQ(endangle, 4)) Then
        For u = -cr To 0 - (1 / samples) Step 1 / samples
            ptnum += 1

            x(ptnum) = u + cx
            y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
            z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
            x2(ptnum) = (u + 1 / samples) + cx
            y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
            z2(ptnum) = cz
        Next
    ElseIf testQ(startangle, 4) And testQ(endangle, 3) Then
        For u = -cr To 0 - (1 / samples) Step 1 / samples
            ptnum += 1

            x(ptnum) = u + cx
            y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
            z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
            x2(ptnum) = (u + 1 / samples) + cx
            y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
            z2(ptnum) = cz
        Next
    ElseIf ((testQ(startangle, 1) And testQ(endangle, 1)) Or (testQ(startangle, 3) And testQ(endangle,
3)) Or (testQ(startangle, 4) And testQ(endangle, 4))) And startangle > endangle Then 'trianlges
        For u = -cr To 0 - (1 / samples) Step 1 / samples
            ptnum += 1

            x(ptnum) = u + cx
            y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
            z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
            x2(ptnum) = (u + 1 / samples) + cx
            y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
            z2(ptnum) = cz
        Next
    End If

    ElseIf testQ(startangle, 2) = True And testQ(endangle, 2) = False Then 'start angle only
        'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q2_ef_st.jpg
        For u = -cr To cr * Math.Cos(startangle) - (1 / samples) Step 1 / samples
            ptnum += 1

```



```

x(ptnum) = u + cx
y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
x2(ptnum) = (u + 1 / samples) + cx
y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
z2(ptnum) = cz
Next

ElseIf testQ(startangle, 2) = False And testQ(endangle, 2) = True Then 'end angle only
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q2_et_sf.jpg
For u = cr * Math.Cos(endangle) To 0 - (1 / samples) Step 1 / samples
    ptnum += 1

    x(ptnum) = u + cx
    y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
    z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
    x2(ptnum) = (u + 1 / samples) + cx
    y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
    z2(ptnum) = cz
Next

ElseIf testQ(startangle, 2) = True And testQ(endangle, 2) = True Then 'both angles in Quadrant
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q2_et_st.jpg
If startangle > endangle Then 'outer, exclude inner.
    For u = -cr To cr * Math.Cos(startangle) - (1 / samples) Step 1 / samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next
    For u = cr * Math.Cos(endangle) - (1 / samples) To 0 - (1 / samples) Step 1 / samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next
Else
    For u = cr * Math.Cos(endangle) To cr * Math.Cos(startangle) - (1 / samples) Step 1 / samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz 'as the circles comming in are yet to be rotated by given vector
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next
End If
End If

'QUADRANT 3 -----
If testQ(startangle, 3) = False And testQ(endangle, 3) = False Then 'neither angles in Q3
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q3_ef_sf.jpg
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q3.jpg
If ((testQ(startangle, 1) Or testQ(startangle, 2)) And testQ(endangle, 4)) Or (testQ(startangle, 2)
And testQ(endangle, 1)) Then 'fill all
    For u = -cr To 0 - (1 / samples) Step 1 / samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next
    'else fill NONE

```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```
ElseIf (testQ(startangle, 1) Or testQ(startangle, 2)) And startangle > endangle Then 'triangle
sections fill
    For u = -cr To 0 - (1 / samples) Step 1 / samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next
ElseIf (testQ(startangle, 4) And testQ(endangle, 4)) And startangle > endangle Then 'fill all
    For u = -cr To 0 - (1 / samples) Step 1 / samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next
End If

ElseIf testQ(startangle, 3) = True And testQ(endangle, 3) = False Then 'start angle only
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q3_ef_st.jpg
For u = cr * Math.Cos(startangle) To 0 - (1 / samples) Step 1 / samples
    ptnum += 1

    x(ptnum) = u + cx
    y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
    z(ptnum) = cz
    x2(ptnum) = (u + 1 / samples) + cx
    y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
    z2(ptnum) = cz
Next

ElseIf testQ(startangle, 3) = False And testQ(endangle, 3) = True Then 'end angle only
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q3_et_sf.jpg
For u = -cr To cr * Math.Cos(endangle) - (1 / samples) Step 1 / samples
    ptnum += 1

    x(ptnum) = u + cx
    y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
    z(ptnum) = cz
    x2(ptnum) = (u + 1 / samples) + cx
    y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
    z2(ptnum) = cz
Next

ElseIf testQ(startangle, 3) = True And testQ(endangle, 3) = True Then 'both angles in Quadrant
'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q3_et_st.jpg
If startangle > endangle Then 'outer, exclude inner.
    For u = -cr To cr * Math.Cos(endangle) - (1 / samples) Step 1 / samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next
    For u = cr * Math.Cos(startangle) To 0 - (1 / samples) Step 1 / samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next
Else
    For u = cr * Math.Cos(startangle) To cr * Math.Cos(endangle) - (1 / samples) Step 1 / samples
        ptnum += 1
```



```

        x(ptnum) = u + cx
        y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next
End If
End If

'QUADRANT 4 -----
If testQ(startangle, 4) = False And testQ(endangle, 4) = False Then 'neither in Q4
    'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q4_ef_sf.jpg
    'file:///E:\mydata\ITG_MAJOR\InHouse%20Software\ViewIt\code_diagram_aids\Q4.jpg
    If startangle > endangle Then 'fill all
        For u = 0 To cr - (1 / samples) Step 1 / samples
            ptnum += 1

            x(ptnum) = u + cx
            y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
            z(ptnum) = cz
            x2(ptnum) = (u + 1 / samples) + cx
            y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
            z2(ptnum) = cz
        Next
        'else fill none
    End If

ElseIf testQ(startangle, 4) = True And testQ(endangle, 4) = False Then 'start angle only
    For u = cr * Math.Cos(startangle) To cr - (1 / samples) Step 1 / samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next

ElseIf testQ(startangle, 4) = False And testQ(endangle, 4) = True Then 'end angle only
    For u = 0 To cr * Math.Cos(endangle) - (1 / samples) Step 1 / samples
        ptnum += 1

        x(ptnum) = u + cx
        y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
        z(ptnum) = cz
        x2(ptnum) = (u + 1 / samples) + cx
        y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
        z2(ptnum) = cz
    Next

ElseIf testQ(startangle, 4) = True And testQ(endangle, 4) = True Then 'both angles in Quadrant
    If startangle > endangle Then 'outer, exclude inner.
        For u = 0 To cr * Math.Cos(endangle) - (1 / samples) Step 1 / samples
            ptnum += 1

            x(ptnum) = u + cx
            y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
            z(ptnum) = cz
            x2(ptnum) = (u + 1 / samples) + cx
            y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
            z2(ptnum) = cz
        Next
        For u = cr * Math.Cos(startangle) To cr - (1 / samples) Step 1 / samples
            ptnum += 1

            x(ptnum) = u + cx
            y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
            z(ptnum) = cz
            x2(ptnum) = (u + 1 / samples) + cx
            y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
            z2(ptnum) = cz
        Next
    Else

```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```
For u = cr * Math.Cos(startangle) To cr * Math.Cos(endangle) - (1 / samples) Step 1 / samples
    ptnum += 1

    x(ptnum) = u + cx
    y(ptnum) = -Math.Sqrt(cr ^ 2 - u ^ 2) + cy
    z(ptnum) = cz
    x2(ptnum) = (u + 1 / samples) + cx
    y2(ptnum) = -Math.Sqrt(cr ^ 2 - (u + 1 / samples) ^ 2) + cy
    z2(ptnum) = cz
Next
End If
End If

'-----
'-----

lastptnum = ptnum

For u = firstptnum To lastptnum
    If chkArcLayer.Checked = True Then
        layername(u) = layer & "_" & circlecount
    Else
        layername(u) = layer 'assigns the circle layer to the layer of the line in the circle
    End If
Next

'transform coordinates to get origin at centre of circle //must repeat opposite after rotation
For u = firstptnum To lastptnum
    x(u) = x(u) - cx
    y(u) = y(u) - cy
    z(u) = z(u) - cz

    x2(u) = x2(u) - cx
    y2(u) = y2(u) - cy
    z2(u) = z2(u) - cz
Next

For q = firstptnum To lastptnum 'now apply local coordinate system rotation matrix to match normal vector
in DXF file

    'set up temp's (as x(q)... is reset after x= it is wrong in the z=)
    tx(q) = x(q)
    ty(q) = y(q)
    tz(q) = z(q)

    tx2(q) = x2(q)
    ty2(q) = y2(q)
    tz2(q) = z2(q)

    'must rotate phi about y then theta about z

    'rotate phi about y [equations derived from matrix form]
    x(q) = tx(q) * Math.Cos(rphi) + tz(q) * Math.Sin(rphi) 'do for start and end point
    x2(q) = tx2(q) * Math.Cos(rphi) + tz2(q) * Math.Sin(rphi)
    'y stays the same
    z(q) = -tx(q) * Math.Sin(rphi) + tz(q) * Math.Cos(rphi)
    z2(q) = -tx2(q) * Math.Sin(rphi) + tz2(q) * Math.Cos(rphi)

    'reset temp's
    tx(q) = x(q)
    ty(q) = y(q)
    tz(q) = z(q)

    tx2(q) = x2(q)
    ty2(q) = y2(q)
    tz2(q) = z2(q)

    'rotate theta about z [equations derived from matrix form]
    x(q) = tx(q) * Math.Cos(rtheta) - ty(q) * Math.Sin(rtheta)
    x2(q) = tx2(q) * Math.Cos(rtheta) - ty2(q) * Math.Sin(rtheta)
    y(q) = tx(q) * Math.Sin(rtheta) + ty(q) * Math.Cos(rtheta)
    y2(q) = tx2(q) * Math.Sin(rtheta) + ty2(q) * Math.Cos(rtheta)
    'z stays the same
Next
```



```

For u = firstptnum To lastptnum 'send back to world coord system [translation]
    x(u) = x(u) + cx
    y(u) = y(u) + cy
    z(u) = z(u) + cz

    x2(u) = x2(u) + cx
    y2(u) = y2(u) + cy
    z2(u) = z2(u) + cz
Next
End Sub

Private Function testQ(ByVal angle As Single, ByVal quadrant As Byte) 'angle must be in radians and between
0 and 2pi
    'test whether an angle in radians is in a given quadrant. returns boolean
    Select Case quadrant
        Case Is = 1
            If angle >= 0 And angle <= (pi / 2) Then
                testQ = True
            Else
                testQ = False
            End If

        Case Is = 2
            If angle >= (pi / 2) And angle <= pi Then
                testQ = True
            Else
                testQ = False
            End If

        Case Is = 3
            If angle >= pi And angle <= (3 * pi / 2) Then
                testQ = True
            Else
                testQ = False
            End If

        Case Is = 4
            If angle >= (3 * pi / 2) And angle <= 2 * pi Then
                testQ = True
            Else
                testQ = False
            End If

        Case Else
            testQ = False
    End Select
End Function

Private Sub csvread()
    ptnum = 0
    'opens standard points file (comma seperated values)
    'example line: 0.087119,-8.908052,-57.161629
    'x,y,z
    Do While (readfile.Peek <> -1) 'do until all points have been calculated
        ptnum += 1
        textline = CStr(readfile.ReadLine) 'read the line if text eg."12,3,12.3"
        icoord = textline.Split(",") 'splits the string of all coords into individual string values

        x(ptnum) = CSng(icoord(0)) 'assigns the text coords to respective double variable
        y(ptnum) = CSng(icoord(1))
        z(ptnum) = CSng(icoord(2))
    Loop
End Sub

Private Sub rawread() 'points read only. triangle verticies are not connected with lines
    ptnum = 0
    'opens RAW Triangles (.raw) format files exported from rhino.
    'raw files are triangle mesh files where the mesh is saved as seperate triangles, each triangle has 3
points
    'sample line: 0.087119 -8.908052 -57.161629 0.085287 -8.929597 -57.162285 0.081625 -8.929085 -57.165710
    'ie. x1 y1 z1 x2 y2 z2 x3 y3 z3
    Do While (readfile.Peek <> -1) 'do until all points have been calculated
        ptnum += 1
        textline = CStr(readfile.ReadLine) 'read the line if text eg."12,3,12.3"
        If textline.Contains(" ") Then 'in case line is object identifier line (they do appear in this
format)

```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```
        icoord = textline.Split(" ") 'splits the string of all coords into individual string values
        x(ptnum) = CSng(icoord(0)) 'assigns the text coords to respective variable
        y(ptnum) = CSng(icoord(1))
        z(ptnum) = CSng(icoord(2))
    End If
Loop
End Sub

Private Sub plyread() 'points read only. triangle verticies are not connected with lines
    Dim buffer As String

    ptnum = 0
    readfile.ReadLine() 'chews ply line
    readfile.ReadLine() 'chews format line
    readfile.ReadLine() 'chews comment line
    buffer = readfile.ReadLine() 'element line
    buffer = buffer.Substring(15, buffer.Length - 15) 'number of points in file
    readfile.ReadLine() 'chews header
    readfile.ReadLine()
    readfile.ReadLine()
    readfile.ReadLine()
    readfile.ReadLine()
    readfile.ReadLine()

    For ptnum = 1 To CInt(buffer)
        textline = CStr(readfile.ReadLine) 'read the line if text eg."12,3,12.3"
        icoord = textline.Split(" ") 'splits the string of all coords into individual string values
        x(ptnum) = CSng(icoord(0)) 'assigns the text coords to respective variable
        y(ptnum) = CSng(icoord(1))
        z(ptnum) = CSng(icoord(2))
    Next
    ptnum = buffer
End Sub

Private Sub dxfwrite() 'Writes line OR points to simple DXF format
    'write dxf header
    writefile.WriteLine(" 0")
    writefile.WriteLine("SECTION")

    'comments
    writefile.WriteLine("999")
    writefile.WriteLine("DXF File created by ViewIt " & My.Application.Info.Version.Major & "." &
My.Application.Info.Version.Minor & "." & My.Application.Info.Version.Build & "." & My.Application.Info.Version.
Revision _
    & " | " & System.DateTime.Today & " | " & ptnum & " output lines | " & circlecount & " circles or arcs |
    " & projectionpreset & ".")

    'start of entities header
    writefile.WriteLine(" 2")
    writefile.WriteLine("ENTITIES")

    dxfhandle = 0
    If lines = True Then 'line
        For q = 1 To ptnum
            dxfhandle += 1
            writefile.WriteLine(" 0")
            writefile.WriteLine("LINE")
            writefile.WriteLine(" 5") 'Handle
            writefile.WriteLine(dxfhandle)
            writefile.WriteLine(" 8") 'Layer
            writefile.WriteLine(layername(q))

            'error checking code
            If CStr(xp(q)) = "NaN" Then
                Debug.Print("NaN in output. X start point. Temp fix was made. DXF Handle: " & dxfhandle)
                xp(q) = 0
            End If

            If CStr(yp(q)) = "NaN" Then
                Debug.Print("NaN in output. X end point. Temp fix was made. DXF Handle: " & dxfhandle)
                yp(q) = 0
            End If

            If CStr(x2p(q)) = "NaN" Then
                Debug.Print("NaN in output. X end point. Temp fix was made. DXF Handle: " & dxfhandle)
                x2p(q) = xp(q)
            End If
        Next
    End If
```




```

If CStr(y2p(q)) = "NaN" Then
    Debug.Print("NaN in output. Y end point. Temp fix was made. DXF Handle: " & dxfhandle)
    y2p(q) = yp(q)
End If
'end error checking

writefile.WriteLine(" 10") 'X
writefile.WriteLine(xp(q))
writefile.WriteLine(" 20") 'Y
writefile.WriteLine(yp(q))
writefile.WriteLine(" 30") 'Z
writefile.WriteLine("0.0")
writefile.WriteLine(" 11") 'X2
writefile.WriteLine(x2p(q))
writefile.WriteLine(" 21") 'Y2
writefile.WriteLine(y2p(q))
writefile.WriteLine(" 31") 'Z2
writefile.WriteLine("0.0") 'if debugging this can be z2p(q), turn of process() and should be same
as input (circles will be lines)

Next
Else 'point
    For q = 1 To ptnum
        dxfhandle += 1
        writefile.WriteLine(" 0")
        writefile.WriteLine("POINT")
        writefile.WriteLine(" 5") 'Handle
        writefile.WriteLine(dxfhandle)
        writefile.WriteLine(" 8") 'Layer
        'If layername(q) = "" Then layername(q) = "UNLAYERED" 'in case no layer was assigned
        writefile.WriteLine(layername(q))
        writefile.WriteLine(" 10") 'X
        writefile.WriteLine(xp(q))
        writefile.WriteLine(" 20") 'Y
        writefile.WriteLine(yp(q))
        writefile.WriteLine(" 30") 'Z
        writefile.WriteLine("0.0")
    Next
End If

'write dxf footer
writefile.WriteLine(" 0")
writefile.WriteLine("ENDSEC")
writefile.WriteLine(" 0")
writefile.WriteLine("EOF")
End Sub

Private Sub csvwrite()
    For q = 1 To ptnum
        writefile.WriteLine(xp(q) & "," & yp(q) & ",0") 'haha, this format is easier to write to! just x,y,z
        with each point on a new line
    Next
End Sub

Private Sub coordsys() 'change the coord system to z-up, right hand
    Select Case CoordSysCombo
        Case Is = "Z-up Right Hand Coordinate System"
            For q = 1 To ptnum
                y(q) = -y(q) 'mirror y
            Next
        Case Is = "Y-up Left Hand Coordinate System"
            For q = 1 To ptnum
                y(q) = z(q) 'switch y and z
                z(q) = y(q)
            Next
        Case Is = "Y-up Right Hand Coordinate System"
            For q = 1 To ptnum
                y(q) = z(q) 'switch y and z
                z(q) = y(q)
                y(q) = -y(q) 'then mirror new y
            Next
        Case Else ' Z-up Left Hand Coordinate System
            'proj() is set up to deal with this system, so it is left as is
    End Select
End Sub

```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```
Private Sub frontface()  
    Dim tx, ty, tz, tx2, ty2, tz2 As Single `temps  
  
    Select Case FrontFaceCombo  
        Case Is = "yz Visible"  
            `do nothing set up for this  
  
        Case Is = "xz Visible" `rotate -90deg about z  
            For q = 1 To ptnum  
                tx = x(q)  
                ty = y(q)  
                tz = z(q)  
                tx2 = x2(q)  
                ty2 = y2(q)  
                tz2 = z2(q)  
  
                x(q) = ty  
                y(q) = -tx  
  
                x2(q) = ty2  
                y2(q) = -tx2  
                `x(q) = tx * Math.Cos((-pi / 2)) - ty * Math.Sin((-pi / 2))  
                `y(q) = tx * Math.Sin((-pi / 2)) + ty * Math.Cos((-pi / 2))  
            Next  
  
        Case Is = "xz Hidden" `rotate +90deg about z  
            For q = 1 To ptnum  
                tx = x(q)  
                ty = y(q)  
                tz = z(q)  
                tx2 = x2(q)  
                ty2 = y2(q)  
                tz2 = z2(q)  
  
                x(q) = -ty  
                y(q) = tx  
  
                x2(q) = -ty2  
                y2(q) = tx2  
                `x(q) = tx * Math.Cos((pi / 2)) - ty * Math.Sin((pi / 2))  
                `y(q) = tx * Math.Sin((pi / 2)) + ty * Math.Cos((pi / 2))  
            Next  
  
        Case Is = "yz Hidden" `rotate +180deg about z  
            For q = 1 To ptnum  
                tx = x(q)  
                ty = y(q)  
                tz = z(q)  
                tx2 = x2(q)  
                ty2 = y2(q)  
                tz2 = z2(q)  
  
                x(q) = -tx  
                y(q) = -ty  
  
                x2(q) = -tx2  
                y2(q) = -ty2  
                `x(q) = tx * Math.Cos((pi)) - ty * Math.Sin((pi))  
                `y(q) = tx * Math.Sin((pi)) + ty * Math.Cos((pi))  
            Next  
  
        Case Is = "xy Visible" `rotate -90deg about y  
            For q = 1 To ptnum  
                tx = x(q)  
                ty = y(q)  
                tz = z(q)  
                tx2 = x2(q)  
                ty2 = y2(q)  
                tz2 = z2(q)  
  
                x(q) = -tz  
                z(q) = tx  
  
                x2(q) = -tz2  
                z2(q) = tx2  
                `x(q) = tx * Math.Cos((-pi / 2)) + tz * Math.Sin((-pi / 2))  
                `z(q) = -tx * Math.Sin((-pi / 2)) + tz * Math.Cos((-pi / 2))  
            Next  
        End Select  
    End Sub
```



Next

```
Case Is = "xy Hidden" `rotate +90deg about y
    tx = x(q)
    ty = y(q)
    tz = z(q)
    tx2 = x2(q)
    ty2 = y2(q)
    tz2 = z2(q)

    x(q) = tz
    z(q) = -tx

    x2(q) = tz2
    z2(q) = -tx2
    `x(q) = tx * Math.Cos((-pi / 2)) + tz * Math.Sin((-pi / 2))
    `z(q) = -tx * Math.Sin((-pi / 2)) + tz * Math.Cos((-pi / 2))
```

End Select

End Sub

```
Private Sub rotateazel()
    Dim tx, ty, tz, tx2, ty2, tz2 As Single `temps

    `change from degrees to radians
    azimuth = (azimuth / 180) * pi
    elevation = (elevation / 180) * pi

    `order of these rotations is important. matricies are NON commutative

    `rotate -azimuth about z
    For q = 1 To ptnum
        tx = x(q)
        ty = y(q)
        tz = z(q)
        tx2 = x2(q)
        ty2 = y2(q)
        tz2 = z2(q)

        x(q) = tx * Math.Cos(-azimuth) - ty * Math.Sin(-azimuth)
        y(q) = tx * Math.Sin(-azimuth) + ty * Math.Cos(-azimuth)

        x2(q) = tx2 * Math.Cos(-azimuth) - ty2 * Math.Sin(-azimuth)
        y2(q) = tx2 * Math.Sin(-azimuth) + ty2 * Math.Cos(-azimuth)
    Next
```

Next

```
`rotate -elevation about y
For q = 1 To ptnum
    tx = x(q)
    ty = y(q)
    tz = z(q)
    tx2 = x2(q)
    ty2 = y2(q)
    tz2 = z2(q)

    x(q) = tx * Math.Cos(-elevation) + tz * Math.Sin(-elevation)
    z(q) = -tx * Math.Sin(-elevation) + tz * Math.Cos(-elevation)

    x2(q) = tx2 * Math.Cos(-elevation) + tz2 * Math.Sin(-elevation)
    z2(q) = -tx2 * Math.Sin(-elevation) + tz2 * Math.Cos(-elevation)
Next
```

Next

End Sub

```
Private Sub proj(ByVal ptnum As Integer)
    `z-up, left hand coordinate system used
    If lines = True Then `lines
        For q = 1 To ptnum `apply specified projection
            xp(q) = Sx * x(q) * Math.Cos(alpha) - Sy * y(q) * Math.Cos(beta)
            yp(q) = Sz * z(q) + Sx * x(q) * Math.Sin(alpha) + Sy * y(q) * Math.Sin(beta)

            x2p(q) = Sx * x2(q) * Math.Cos(alpha) - Sy * y2(q) * Math.Cos(beta)
            y2p(q) = Sz * z2(q) + Sx * x2(q) * Math.Sin(alpha) + Sy * y2(q) * Math.Sin(beta)
        Next
    Else `points only
        For q = 1 To ptnum `apply specified projection
            xp(q) = Sx * x(q) * Math.Cos(alpha) - Sy * y(q) * Math.Cos(beta)
```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```

        yp(q) = Sz * z(q) + Sx * x(q) * Math.Sin(alpha) + Sy * y(q) * Math.Sin(beta)
    Next
End If
End Sub

Private Sub perspproj(ByVal ptnum As Integer)
    'z-up, left hand coordinate system used
    If lines = True Then 'lines
        For q = 1 To ptnum 'apply one point perspective
            xp(q) = -(VPdepth * ((y(q) - VPy) / (VPdepth + x(q))) + VPy)
            yp(q) = VPdepth * ((z(q) - VPz) / (VPdepth + x(q))) + VPz

            x2p(q) = -(VPdepth * ((y2(q) - VPy) / (VPdepth + x2(q))) + VPy)
            y2p(q) = VPdepth * ((z2(q) - VPz) / (VPdepth + x2(q))) + VPz
        Next

    Else 'points only
        For q = 1 To ptnum 'apply one point perspective
            xp(q) = -(VPdepth * ((y(q) - VPy) / (VPdepth + x(q))) + VPy)
            yp(q) = VPdepth * ((z(q) - VPz) / (VPdepth + x(q))) + VPz
        Next
    End If
End Sub

Private Sub btnInput_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnInput.
Click
    OFD.Title = "Input File"
    OFD.Filter = "AutoCAD Drawing Exchange File (*.dxf)|*.dxf|Text Files (*.txt)|*.txt|Comma Separated
Values (*.csv)|*.csv|RAW Trianlges (*.raw)|*.raw|Polygon File Format (*.ply)|*.ply|All files (*.*)|*.*"
    OFD.ShowDialog() 'shows the dialogue box to select the input file
    readfilespec = OFD.FileName

    inclicked = True
    If inclicked = True And outclicked = True Then
        btnProcess.Enabled = True
    End If
End Sub

Private Sub btnOutput_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnOutput.
Click
    SFD.Title = "Output File"
    SFD.Filter = "AutoCAD Drawing Exchange File (*.dxf)|*.dxf|Text Files (*.txt)|*.txt|Comma Separated
Values (*.csv)|*.csv|All files (*.*)|*.*"
    SFD.ShowDialog() 'shows the dialogue box to select the output file
    writefilespec = SFD.FileName

    outclicked = True
    If inclicked = True And outclicked = True Then
        btnProcess.Enabled = True
    End If
End Sub

Private Sub btnProcess_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnProcess.
Click
    'azimuth & elevation check
    If azel_notready = True Then
        MsgBox("The specified projection 'Axonometric: Azimuth and Elevation' has not been set up. Please
click 'Get from Azimuth and Elevation' or choose a differnt projection.",MsgBoxStyle.Critical)
        Exit Sub
    End If

    'button that initiates whole process
    ToolStripStatusLabel1.Text = "Processing..."
    ToolStripProgressBar1.Style = ProgressBarStyle.Marquee 'stats the progress bar moving
    btnProcess.Text = "Processing... Please wait." 'notifies user task in progress
    btnProcess.Enabled = False 'stops users from pressing the button while it is progressing
    btnInput.Enabled = False
    btnOutput.Enabled = False
    Me.Cursor = Cursors.WaitCursor

    alpha = (CSng(txtAlpha.Text) / 180) * pi 'stores the variables (converting units to RADIANS)
    beta = (CSng(txtBeta.Text) / 180) * pi
    Sx = CSng(txtScaleX.Text)
    Sy = CSng(txtScaleY.Text)
    Sz = CSng(txtScaleZ.Text)

```



```

VPy = CSng(txtVPy.Text) 'vanishing point, stores even if not doing perspective
VPz = CSng(txtVPz.Text)
VPdepth = CSng(txtDepth.Text)

CoordSysCombo = cmbCoordSys.SelectedItem
FrontFaceCombo = cmbFrontFace.SelectedItem
projectionpreset = cmbPreSet.SelectedItem

BackgroundWorker1.RunWorkerAsync() 'starts the calculation work on a separate thread

End Sub

Private Sub cmbPreSet_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles cmbPreSet.SelectedIndexChanged
    Select Case cmbPreSet.SelectedItem 'if a preset is selected then the values appear in the
respective text box
        Case Is = "Custom..."
            txtAlpha.Text = "30"
            txtBeta.Text = "30"
            txtScaleX.Text = "1"
            txtScaleY.Text = "1"
            txtScaleZ.Text = "1"
            btnDefaultAxonScales.Enabled = False
            btnAzEl.Enabled = False
            azel_notready = False
            projectionvarsenabled(True)
            projectionvarsvisible(True)
            btnAzEl.Visible = False
            btnDefaultAxonScales.Visible = False

        Case Is = "Front/Back View" 'yz
            txtAlpha.Text = "0" 'irrelevant
            txtBeta.Text = "0"
            txtScaleX.Text = "0"
            txtScaleY.Text = "1"
            txtScaleZ.Text = "1"
            btnDefaultAxonScales.Enabled = False
            btnAzEl.Enabled = False
            azel_notready = False
            projectionvarsenabled(False)
            projectionvarsvisible(True)
            btnAzEl.Visible = False
            btnDefaultAxonScales.Visible = False

        Case Is = "Top/Bottom View" 'yx
            txtAlpha.Text = "90"
            txtBeta.Text = "0"
            txtScaleX.Text = "1"
            txtScaleY.Text = "1"
            txtScaleZ.Text = "0"
            btnDefaultAxonScales.Enabled = False
            btnAzEl.Enabled = False
            azel_notready = False
            projectionvarsenabled(False)
            projectionvarsvisible(True)
            btnAzEl.Visible = False
            btnDefaultAxonScales.Visible = False

        Case Is = "Left/Right View" 'xz
            txtAlpha.Text = "0"
            txtBeta.Text = "0" 'irrelevant
            txtScaleX.Text = "1"
            txtScaleY.Text = "0"
            txtScaleZ.Text = "1"
            btnDefaultAxonScales.Enabled = False
            btnAzEl.Enabled = False
            azel_notready = False
            projectionvarsenabled(False)
            projectionvarsvisible(True)
            btnAzEl.Visible = False
            btnDefaultAxonScales.Visible = False

        Case Is = "Isometric"
            txtAlpha.Text = "30"
            txtBeta.Text = "30"
            txtScaleX.Text = "1"
            txtScaleY.Text = "1"
    
```




```
txtScaleZ.Text = "1"
btnDefaultAxonScales.Enabled = False
btnAzEl.Enabled = False
azel_notready = False
projectionvarsenabled(False)
projectionvarsvisible(True)
btnAzEl.Visible = False
btnDefaultAxonScales.Visible = False

Case Is = "Standard Dimetric"
'approx. a = 7, b = 41.5, then x:y:z = 1:1:0.5
txtAlpha.Text = "7.1807557814582813230389956261486"
txtBeta.Text = "41.409622109270859338480502186926"
txtScaleX.Text = "0.942809046515349"
txtScaleY.Text = "0.942809041323189"
txtScaleZ.Text = "0.471404511442209"
btnDefaultAxonScales.Enabled = True
btnAzEl.Enabled = False
azel_notready = False
projectionvarsenabled(True)
txtScaleX.Enabled = False
txtScaleY.Enabled = False
txtScaleZ.Enabled = False
projectionvarsvisible(True)
btnAzEl.Visible = False
btnDefaultAxonScales.Visible = True

Case Is = "Standard Trimetric"
txtAlpha.Text = "10"
txtBeta.Text = "45"
txtScaleX.Text = "0.936234226565589"
txtScaleY.Text = "0.90756433205541"
txtScaleZ.Text = "0.547533064013444"
btnDefaultAxonScales.Enabled = True
btnAzEl.Enabled = False
azel_notready = False
projectionvarsenabled(True)
txtScaleX.Enabled = False
txtScaleY.Enabled = False
txtScaleZ.Enabled = False
projectionvarsvisible(True)
btnAzEl.Visible = False
btnDefaultAxonScales.Visible = True

Case Is = "Cabinet Oblique"
txtAlpha.Text = "45"
txtBeta.Text = "0"
txtScaleX.Text = "0.5"
txtScaleY.Text = "1"
txtScaleZ.Text = "1"
btnDefaultAxonScales.Enabled = False
btnAzEl.Enabled = False
azel_notready = False
projectionvarsenabled(False)
projectionvarsvisible(True)
btnAzEl.Visible = False
btnDefaultAxonScales.Visible = False

Case Is = "Cavalier Oblique"
txtAlpha.Text = "45"
txtBeta.Text = "0"
txtScaleX.Text = "1"
txtScaleY.Text = "1"
txtScaleZ.Text = "1"
btnDefaultAxonScales.Enabled = False
btnAzEl.Enabled = False
azel_notready = False
projectionvarsenabled(False)
projectionvarsvisible(True)
btnAzEl.Visible = False
btnDefaultAxonScales.Visible = False

Case Is = "Planometric"
txtAlpha.Text = "45"
txtBeta.Text = "45"
txtScaleX.Text = "1"
txtScaleY.Text = "1"
```



```
txtScaleZ.Text = "1"
btnDefaultAxonScales.Enabled = False
btnAzEl.Enabled = False
azel_notready = False
projectionvarsenabled(False)
projectionvarsvisible(True)
btnAzEl.Visible = False
btnDefaultAxonScales.Visible = False

Case Is = "Axonometric: Azimuth and Elevation"
txtAlpha.Text = "90"
txtBeta.Text = "0"
txtScaleX.Text = "0"
txtScaleY.Text = "1"
txtScaleZ.Text = "1"
btnDefaultAxonScales.Enabled = False
btnAzEl.Enabled = True
azel_notready = True
projectionvarsenabled(False)
projectionvarsvisible(True)
btnAzEl.Visible = True
btnDefaultAxonScales.Visible = False

Case Is = "One Point Perspective"
btnDefaultAxonScales.Enabled = False
btnAzEl.Enabled = False
azel_notready = False
projectionvarsenabled(False)
projectionvarsvisible(False)
btnAzEl.Visible = False
btnDefaultAxonScales.Visible = False

Case Else
    Debug.Print("Unknown projection selected from pre-set projection drop-down box. This is
a bug in the software.")
    MsgBox("Unknown projection selected from pre-set projection drop-down box. This is a
bug in the software.")
End Select
End Sub

Private Sub projectionvarsvisible(ByVal visible As Boolean)
    Select Case visible
        Case Is = True
            Label4.Visible = True
            Label5.Visible = True
            Label6.Visible = True
            Label7.Visible = True
            Label8.Visible = True

            txtAlpha.Visible = True
            txtBeta.Visible = True
            txtScaleX.Visible = True
            txtScaleY.Visible = True
            txtScaleZ.Visible = True

            lblVPy.Visible = False
            lblVPz.Visible = False
            txtVPy.Visible = False
            txtVPz.Visible = False
            lblDepth.Visible = False
            txtDepth.Visible = False

            btnAzEl.Visible = True
            btnDefaultAxonScales.Visible = True

            btnDefaultPersp.Visible = False

        Case Is = False
            Label4.Visible = False
            Label5.Visible = False
            Label6.Visible = False
            Label7.Visible = False
            Label8.Visible = False

            txtAlpha.Visible = False
            txtBeta.Visible = False
            txtScaleX.Visible = False
```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```
txtScaleY.Visible = False
txtScaleZ.Visible = False

lblVPy.Visible = True
lblVPz.Visible = True
txtVPy.Visible = True
txtVPz.Visible = True
lblDepth.Visible = True
txtDepth.Visible = True

txtVPy.Enabled = True
txtVPz.Enabled = True
txtDepth.Enabled = True

btnDefaultPersp.Enabled = True

btnAzEl.Visible = False
btnDefaultAxonScales.Visible = False

btnDefaultPersp.Visible = True

End Select
End Sub

Private Sub projectionvarsenabled(ByVal enabled As Boolean)
    If enabled = False Then
        txtAlpha.Enabled = False
        txtBeta.Enabled = False
        txtScaleX.Enabled = False
        txtScaleY.Enabled = False
        txtScaleZ.Enabled = False
    Else
        txtAlpha.Enabled = True
        txtBeta.Enabled = True
        txtScaleX.Enabled = True
        txtScaleY.Enabled = True
        txtScaleZ.Enabled = True
    End If
End Sub

Private Sub btnShowHideHelp_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnShowHideHelp.Click
    'widens the form to show the help steps and text
    If btnShowHideHelp.Text = "Hide Help <-" Then
        btnShowHideHelp.Text = "Show Help ->"
        Me.Width = 292
    Else
        btnShowHideHelp.Text = "Hide Help <-"
        Me.Width = 961
    End If
End Sub

Private Sub btnAbout_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnAbout.Click
    AboutBox1.ShowDialog() 'about dialogue box
End Sub

Private Sub btnDefaultAxonScales_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnDefaultAxonScales.Click
    Dim a, b As Single

    a = CSng(txtAlpha.Text)
    b = CSng(txtBeta.Text)

    If a + b >= 90 Then 'checks if angles are valid
        MsgBox("Invalid Alpha & Beta values. Must statisfy, a + b < 90", MsgBoxStyle.Critical, "ViewIt")
    Exit Sub
    End If

    If cmbPreSet.SelectedItem = "Dimetric" Then
        If a + 2 * b <> 90 Or a <> b Or 2 * a + b <> 90 Then 'checks if angles are valid
            MsgBox("Warning. For conventional dimetric drawing, a and b must satisfy, a = b OR 2a + b = 90 OR a + 2b = 90", MsgBoxStyle.Exclamation, "ViewIt")
        Exit Sub
    End If
    ElseIf cmbPreSet.SelectedItem = "Trimetric" Then
```



```

        If a + 2 * b = 90 Or 2 * a + b = 90 Or a = b Then 'checks if angles are valid
            MsgBox("Warning. For conventional trimetric drawing, a and b must satisfy, a <> b OR 2a
+ b <> 90 OR a + 2b <> 90", MsgBoxStyle.Exclamation, "ViewIt")
            Exit Sub
        End If
    End If

    a = a / 180 * pi 'convert to radians
    b = b / 180 * pi

    'all angles must be valid if reached here, so calculate the scales
    txtScaleX.Text = Math.Sqrt((Math.Tan(b) * (1 + Math.Tan(a) * Math.Tan(a))) / (Math.Tan(a) +
Math.Tan(b)))
    txtScaleY.Text = Math.Sqrt(1 - Math.Tan(a) * Math.Tan(b))
    txtScaleZ.Text = Math.Sqrt((Math.Tan(a) * (1 + Math.Tan(b) * Math.Tan(b))) / (Math.Tan(a) +
Math.Tan(b)))
    End Sub

Private Sub frmMain_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
    cmbCoordSys.SelectedItem = "Z-up Left Hand Coordinate System" 'sets the default item
    Me.BringToFront()
    Me.CenterToScreen()
End Sub

Private Sub BackgroundWorker1_DoWork(ByVal sender As System.Object, ByVal e As System.
ComponentModel.DoWorkEventArgs) Handles BackgroundWorker1.DoWork
    'background worker is used to run the process on a seperate thread, this leaves the ui
responsive
    process() 'runs the process sub
    'to cancel e.cancel = true. however can only be called in this sub. as process() sub is called
cannot cancel till it is complete. todo: possibly move process() to here, that gives more ability to
cancel, get progress, etc.
End Sub

Private Sub BackgroundWorker1_RunWorkerCompleted(ByVal sender As System.Object, ByVal e As System.
ComponentModel.RunWorkerCompletedEventArgs) Handles BackgroundWorker1.RunWorkerCompleted
    'reset variables so they don't interfere if the process is done again
    Array.Clear(x, 0, x.GetLength(0))
    Array.Clear(y, 0, y.GetLength(0))
    Array.Clear(z, 0, z.GetLength(0))
    Array.Clear(x2, 0, x2.GetLength(0))
    Array.Clear(y2, 0, y2.GetLength(0))
    Array.Clear(z2, 0, z2.GetLength(0))
    Debug.Print("Total Output Lines: " & ptnum)
    ptnum = 0
    Array.Clear(xp, 0, xp.GetLength(0))
    Array.Clear(yp, 0, yp.GetLength(0))
    Array.Clear(x2p, 0, x2p.GetLength(0))
    Array.Clear(y2p, 0, y2p.GetLength(0))
    Array.Clear(layername, 0, layername.GetLength(0))
    dxfhandle = 0
    lines = False
    circlecount = 0

    btnProcess.Text = "Process" 'resets button text now that process is done
    btnProcess.Enabled = True 'allows users to click the button again
    btnInput.Enabled = True
    btnOutput.Enabled = True
    ToolStripProgressBar1.Style = ProgressBarStyle.Blocks 'stops the progress bar moving
    ToolStripStatusLabel1.Text = "Ready"
    Me.Cursor = Cursors.Default
    My.Computer.Audio.PlaySystemSound(Media.SystemSounds.Asterisk)
End Sub

Private Sub cmbFrontFace_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles cmbFrontFace.SelectedIndexChanged
    Select Case cmbFrontFace.SelectedItem
        Case Is = "yz Visible"
            PictureBox1.Image = ViewIt.My.Resources.Resources.yz
        Case Is = "xz Visible"
            PictureBox1.Image = ViewIt.My.Resources.Resources.zx
        Case Is = "xz Hidden"
            PictureBox1.Image = ViewIt.My.Resources.Resources.xz_hidden
        Case Is = "yz Hidden"
            PictureBox1.Image = ViewIt.My.Resources.Resources.yz_hidden
        Case Is = "xy Visible"
    
```



- 12. APPENDIX A - DEVELOPED SOFTWARE -

```
        PictureBox1.Image = ViewIt.My.Resources.Resources.yx
        Case Is = "xy Hidden"
            PictureBox1.Image = ViewIt.My.Resources.Resources.yx_hidden
        End Select
    End Sub

    Private Sub btnAzEl_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btnAzEl.Click
        Dim result As Integer
        result = frmAzEl.ShowDialog()

        If result = vbCancel Then
            azel_notready = True
        Else
            azel_notready = False
        End If
    End Sub

#Region " Menu"
    Private Sub ExitToolStripMenuItem1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles ExitToolStripMenuItem1.Click
        End
    End Sub

    Private Sub OpenToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles OpenToolStripMenuItem.Click
        OFD.Title = "Input File"
        OFD.Filter = "AutoCAD Drawing Exchange File (*.dxf)|*.dxf|Text Files (*.txt)|*.txt|Comma
Seperated Values (*.csv)|*.csv|RAW Trianlges (*.raw)|*.raw|Polygon File Format (*.ply)|*.ply|All files
(*.*)|*.*"
        OFD.ShowDialog() 'shows the dialogue box to select the input file
        readfilespec = OFD.FileName

        inclicked = True
        If inclicked = True And outclicked = True Then
            btnProcess.Enabled = True
        End If
    End Sub

    Private Sub SaveToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles SaveToolStripMenuItem.Click
        SFD.Title = "Output File"
        SFD.Filter = "AutoCAD Drawing Exchange File (*.dxf)|*.dxf|Text Files (*.txt)|*.txt|Comma
Seperated Values (*.csv)|*.csv|All files (*.*)|*.*"
        SFD.ShowDialog() 'shows the dialogue box to select the output file
        writefilespec = SFD.FileName

        outclicked = True
        If inclicked = True And outclicked = True Then
            btnProcess.Enabled = True
        End If
    End Sub

    Private Sub NewToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles NewToolStripMenuItem.Click
        resetall()
    End Sub

    Private Sub resetall()
        'clear everything, as if program just started

        Array.Clear(x, 0, x.GetLength(0))
        Array.Clear(y, 0, y.GetLength(0))
        Array.Clear(z, 0, z.GetLength(0))
        Array.Clear(x2, 0, x2.GetLength(0))
        Array.Clear(y2, 0, y2.GetLength(0))
        Array.Clear(z2, 0, z2.GetLength(0))
        ptnum = 0
        Array.Clear(xp, 0, xp.GetLength(0))
        Array.Clear(yp, 0, yp.GetLength(0))
        Array.Clear(x2p, 0, x2p.GetLength(0))
        Array.Clear(y2p, 0, y2p.GetLength(0))
        Array.Clear(layername, 0, layername.GetLength(0))
        dxfhandle = 0
        lines = False
        circlecount = 0
        alpha = 0
    End Sub
End Sub
```




```

beta = 0
Sx = 0
Sy = 0
Sz = 0
samples = 0
startangle = 0
endangle = 0
circlecount = 0
layer = ""
azimuth = 0
elevation = 0
azel_notready = True
outclicked = False
inclicked = False
readfilespec = ""
writefilespec = ""
textline = ""
q = 0
CoordSysCombo = ""
FrontFaceCombo = ""
projectionpreset = ""

usedefaultpersp = False
defpersp = 0
VPy = 0
VPz = 0
VPdepth = 0

projectionvarsvisible(True)

btnProcess.Text = "Process"
btnProcess.Enabled = True
btnInput.Enabled = True
btnOutput.Enabled = True
ToolStripProgressBar1.Style = ProgressBarStyle.Blocks 'stops the progress bar moving
ToolStripStatusLabel1.Text = "Ready"

cmbCoordSys.SelectedItem = "Z-up Left Hand Coordinate System"
cmbFrontFace.SelectedItem = "yz Visible"
cmbPreSet.SelectedItem = "Isometric"

projectionvarsenabled(False)
txtSamples.Text = "100"
chkArcLayer.Checked = True
btnDefaultAxonScales.Enabled = False
btnAzEl.Enabled = False
btnProcess.Enabled = False
Me.Width = 292
End Sub

Private Sub AboutToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles AboutToolStripMenuItem.Click
AboutBox1.ShowDialog() 'about dialogue box
End Sub

Private Sub ContentsToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles ContentsToolStripMenuItem.Click
'widens the form to show the help steps and text
If btnShowHideHelp.Text = "Hide Help <-" Then
btnShowHideHelp.Text = "Show Help ->"
Me.Width = 292
Else
btnShowHideHelp.Text = "Hide Help <-"
Me.Width = 961
End If
End Sub

#End Region

Private Sub btnDefaultPersp_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnDefaultPersp.Click
Dim result As Integer

txtVPy.Enabled = False
txtVPz.Enabled = False

result = frmPersp.ShowDialog()

```



```
If result = vbCancel Then
    usedefaultpersp = False
Else
    usedefaultpersp = True
End If

End Sub

End Class
```

EVALUATION

In this tool, compared to the oblique projection tool, the user interface was cleaned up and modernized. I rewrote most of the code making it better and adding much more functionality. Including widening the import and export file formats. I added the functionality to import DXF, CSV, RAW and PLY files and export DXF and CSV files. I could now use various methods to make the drawings that I wanted. I could use the mesh from a complex object, or I could use 3D lines of the 3D model (made in Rhinoceros by using the *silhouette* command, or in AutoCAD by simply exporting the model as DXF R12), or I could just transform a cluster of 3D points from Rhinoceros. The problem with using meshes is that they contain many vertices (my watch in high quality mesh contained 341 116 vertices) and the output I got was only a cluster of points, not to mention hidden points were also mixed in with the visible ones, so I mainly used the *ExtractWireframe* command which kept the lines to the minimum. The *silhouette* command would have been better but it often omitted critical lines. That being said, the *silhouette* command still worked most of the time with simple objects.

I would have liked to implement hidden line removal algorithms, however they would have taken too much time and effort to implement. Larger and more complex data management processes would have to be implemented to accommodate hidden line removal, which would still require more work as some of the standard algorithms only work for projection lines normal to the projection plane, ie. axonometric, they would not work for distorted views such as oblique and planometric. Also some algorithms are only designed for perspective projection.

I could have also included two and three point perspective projection. However there are many more user specified variables where there is no standard value. The view point is the main variable and it would work best with a graphical method of selection, I however did not implement this, mainly because Rhinoceros was very good at generating 2D three point perspective drawings. But Rhinoceros could only do this three-point perspective, it cannot generate one-point or two-point perspective. One-point perspective was incorporated into ViewIt 4.8 but two-point was not, mainly because it is slightly more difficult and I did not have the time to incorporate it.

Also I spent much of my time getting ViewIt to read circles and arcs. One way that I could have avoided this is to write a simple plug-in for Rhino that splits arcs and circles and any other curves (DXF export code split curves for me, but not circles and arcs) into many line segments. I ended up writing the code to support circles and arcs for several reasons including that Rhino had no capabilities to split a circle or arc into many line segments. However after I had written all the code to allow ViewIt to read arcs and circles I found a script for a Rhino plug-in that could do this curve splitting (I probably could have wrote it myself, but I didn't think of it), and since ViewIt gave lines as output anyway there was no advantage of getting it to read circles and arcs. This is unless of course I got ViewIt to recognise that circles in isometric become ellipses, and that ViewIt should export these as ellipses not line segments. However this would have minimal advantage in the end. I don't regret having ViewIt read circles and arcs as coding this has exposed me to and allowed me to become aware of the technical DXF issues. It has also allowed me to use maths outside the classroom and it has allowed me to showcase evidence of a range of skills, even if these skills are barely related to the syllabus.

The backbone of ViewIt is the projection system that is based upon two angles and three scales. It would have been much easier if I could code this into the heart of Rhinoceros, however that code is really fundamental and stuck in cement. I cannot edit it, thus I had to rewrite a lot of other code myself.



PROBLEMS AND SOLUTIONS

MULTITHREADING

I set up my application to run the CPU draining tasks on a separate thread to the GUI (graphical user interface), this allowed the GUI to remain responsive whilst a large complex task was in process. Without this feature the program would appear frozen, as if it had crashed while it was processing, even though it had not crashed.

DRAWING EXCHANGE FILE (DXF)

Although I had the choice of a range of file formats I mostly used DXF. DXF (AutoCAD Drawing Exchange Format) is a widely accepted and used format. It is commonly used when transferring data from one application to another. Almost all 2D and 3D CAD and non-CAD applications can open and/or save in DXF. DXF can contain 2D drawing data including lines, circles, arcs, splines, etc. and additional data such as layers, colour, lineweight, etc. It can also contain 3D meshes, and combinations 2D and 3D. The compatibility, made possible by its current wide usage and the freely available technical file format details, along with its ability to have much of the optional data omitted made it the best and simplest option for me when I wanted to export vector data. One negative aspect of DXF is the compression. DXF files can very easily be big in file size, however tools such as WinZIP or WinRAR can be used to compress the file, and in some cases by up to 95%.

In my software I have only used basic features of DXF as I didn't need things like lineweights, etc.

DXF BASIC FORMAT OVERVIEW¹

The following is on the most basic level, the level I used. DXF format consists of a header, body and footer.

Header:

```
0
SECTION
2
ENTITIES
```

Footer:

```
0
ENDSEC
0
EOF
```

Next is the body. The example below is for one point. If two points are required the whole block is repeated directly after it. After "POINT" in the example the properties are set. The format used here is property identifier (eg. Layer is 8, x point is 10, etc. (there are many more)) the line after that is the value (eg. after 8 is LAYERNAME that means that the point will go in the LAYERNAME layer.). A sample is below. Note that my comments appear after //.

```
0 //used to show object
POINT //indicates that the following data is for a point object
5 //handle (a handle is an identifier, each line, circle, etc has a handle)
108 //handle value (different for each object)
8 //layer
LAYERNAME // layer value
10 //x coordinate of point
11.7677669529664 //x value
20 //y coordinate of point
11.7677669529664 //y value
30 //z coordinate of point
0.0 //z value
```

¹ DXF file format information was obtained from many sources. My primary source of information was the AutoCAD 2007 DXF Reference.



MULTIPURPOSE PROJECTION MATRIX

When I decided to expand the oblique projection tool and allow it to generate a huge range of other drawing types I needed a matrix (or algorithm) that did this projection. As I could not find one through research, I derived my own. I did the maths (see derivation in next sub section) and came up with a matrix that projected the 3D points into 2D in a range of drawing projections. (it is not the conventional way to do axonometric projections but it does work 100%. It also means that it works for non axonometric projections (oblique and planometric).)

The reason why I could do so many different drawing types is because it took two angles and three scales as input. Instead of applying the conventional rotation matrices and then project normal to the projection plane. By varying these angles and scale ratios a wide range of drawings can be produced. The core matrix that completed this projection is shown below.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x \cos \alpha & -S_y \cos \beta & 0 & 0 \\ S_x \sin \alpha & S_y \sin \beta & S_z & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Because of the way this algorithm works, if users wanted some standard drawing types some input was taken and the rest was calculated based on that input. This was used for dimetric and trimetric drawings.

DERIVATION OF PROJECTION MATRIX

Firstly I picked a 3D cartesian coordinate system to work with. I choose the Z-up Left Hand Coordinate System, shown in figure 12.8. I imagined a rectangular prism in the 3D coordinate system as shown also in figure 12.8.

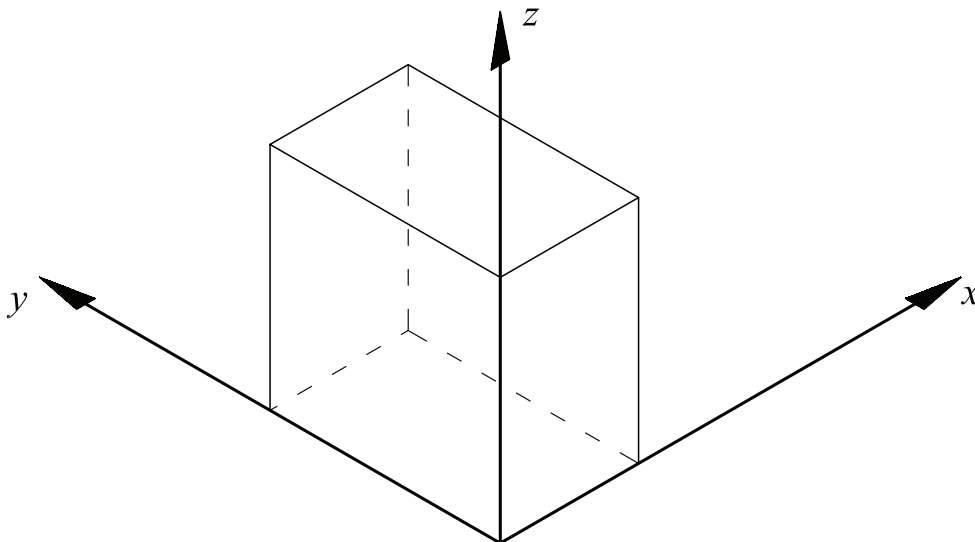


Figure 12.8 - Block in 3D Coordinate System

I then imagined what it would look like in a 2D coordinate system using isometric projection. This is shown in figure 12.9.



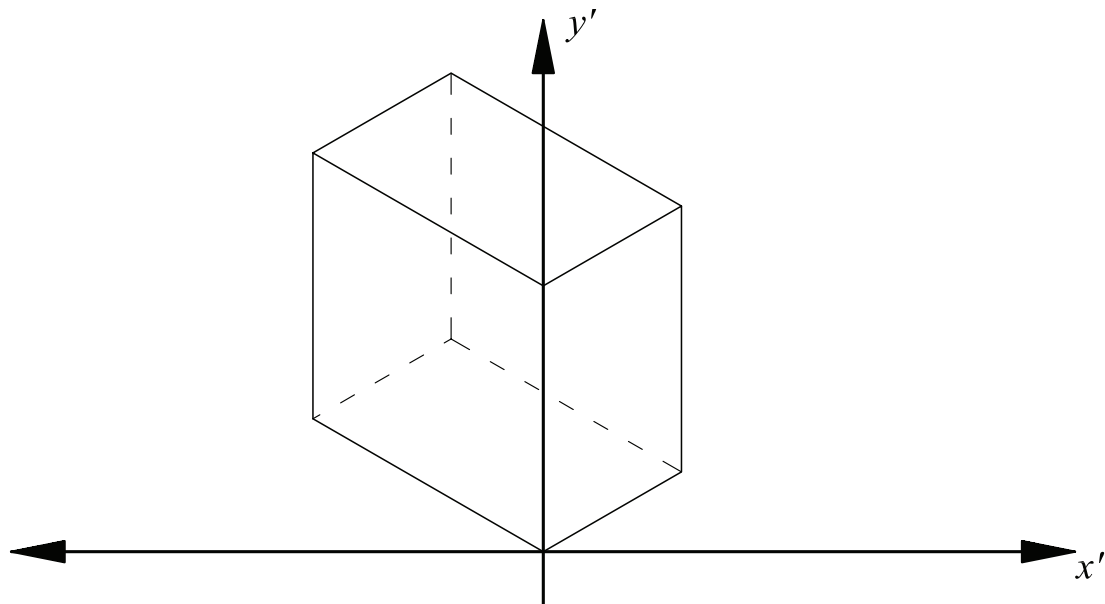


Figure 12.9 - Block in 2D coordinate system (isometric)

As the alpha and beta angles as show in figure 12.10 can change, and therefore not limited to a specific projection, I needed to use alpha and beta in my equations.

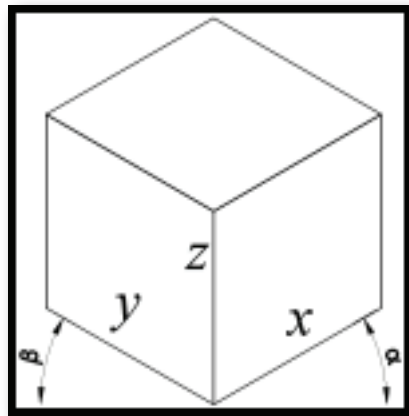
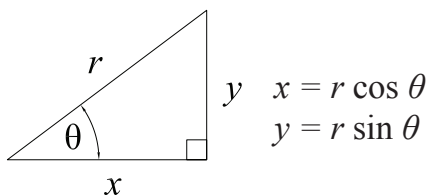


Figure 12.10 - Alpha & Beta Angles

The follow assumed the equations shown below.



The logic process I took follows.

All the points on the xz plane have $y = 0$

Therefore the x' and y' values on the 2D plane will follow the trig property shown above, so:

$$x' = x \cos \alpha$$

$$y' = z + y \sin \alpha$$

However not all the points lie on the xz plane, y is not always equal to zero. By visualising a point with a fixed x and z value but growing larger in y value, its x' will become lower, and y' will become larger. The extent of the x' and y' growth can again be expressed with the trig property shown, and this value can be added in the respective sense to obtain the final combined x' and y' (separately). Hence,



$$\begin{aligned}x' &= x \cos \alpha - y \cos \beta \\y' &= z + x \sin \alpha + y \sin \beta\end{aligned}$$

If y is in the negative direction then the sign will automatically change accordingly. The next step is to incorporate the scaling of the axes. This was done by replacing the x , y & z with a the scale factor as a multiple of the x , y & z . Hence,

$$\begin{aligned}x' &= S_x x \cos \alpha - S_y y \cos \beta \\y' &= S_z z + S_x x \sin \alpha + S_y y \sin \beta\end{aligned}$$

This can now easily be transferred into matrix form or left as is.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x \cos \alpha & -S_y \cos \beta & 0 & 0 \\ S_x \sin \alpha & S_y \sin \beta & S_z & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

ONE POINT PERSPECTIVE ALGORITHM

I derived my own algorithm to preform one-point perspective. I did so using the similar triangles method. However due to time restrains I have not tested this solution for a wide range of data. I have only tested it (and thing it only works) when the object is not on the same side of the picture plane as the vanishing point. However as in my algorithm the picture plane is static on the yz axis, the object can not overlap this. Also due to time restraints I have left it up to the user to choose the distance from the vanishing point to the picture plane (this could probably be calculated).

The coordinate system that my algorithm is based on is shown below in figure 12.11. I have only taken into account when the whole of the model is in the positive x part of the 3D space. I have not considered negative values for x . However I think that to solve this I could translate the object prior to the perspective projection along the x axis so that all points have positive x value.

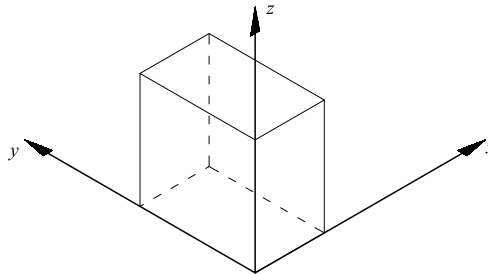


Figure 12.11 - Coordinate system used for perspective projection algorithm

In my derived algorithm I have allowed the user to place the vanishing point (VP) anywhere they want, so long as $VP_x > 0$. However I did build in the functionality for ViewIt to automatically calculate the VP for standard perspective views. Also in my derived algorithm I placed the picture plane (PP) on the yz plane (see figure 12.11). Next I looked at what happened when view from above (see figure 12.12) and when view from the right (see figure 12.14).



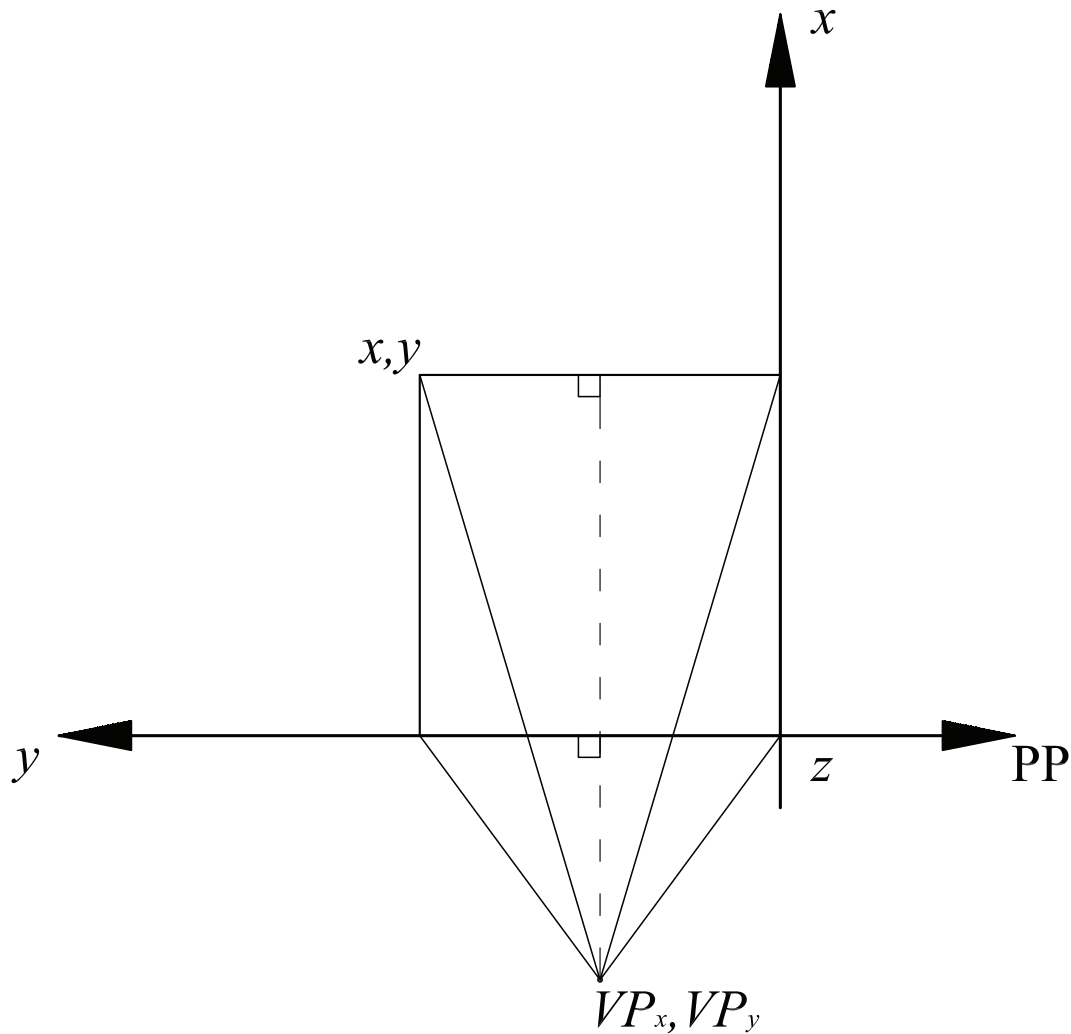


Figure 12.12 - Perspective projection viewed from top

Now if we take a closer look at two of the triangles shown below in figure 12.13, we see that triangle abc is similar to triangle ecd. ie. $\triangle ABC \sim \triangle ECD$. So we know that $\frac{ab}{ce} = \frac{ad}{ed}$ which can be rearranged into $\frac{ab}{ad} = \frac{ce}{ed}$. We can also get this result by using the fact that as the two triangles share the same angle their gradients will be the same thus $\frac{ab}{ad} = \frac{ce}{ed}$.

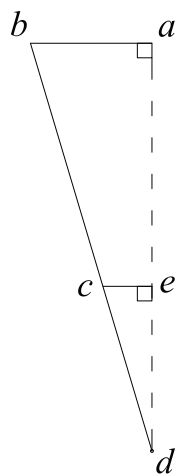


Figure 12.13 - Similar triangles

Now by replacing the letters for the sides of the triangles with the coordinates as shown in figure 12.12 we get the following ratio,



$$\frac{y - VP_y}{VP_{dept h} + x} = \frac{PP_y - VP_y}{VP_{dept h}}$$

Which can be rearranged with PP_y the subject as follows,

$$PP_y = \left(\frac{y - VP_y}{VP_{dept h} + x} \right) VP_{dept h} + VP_y$$

A similar process can be done when viewing from the front to get PP_z .

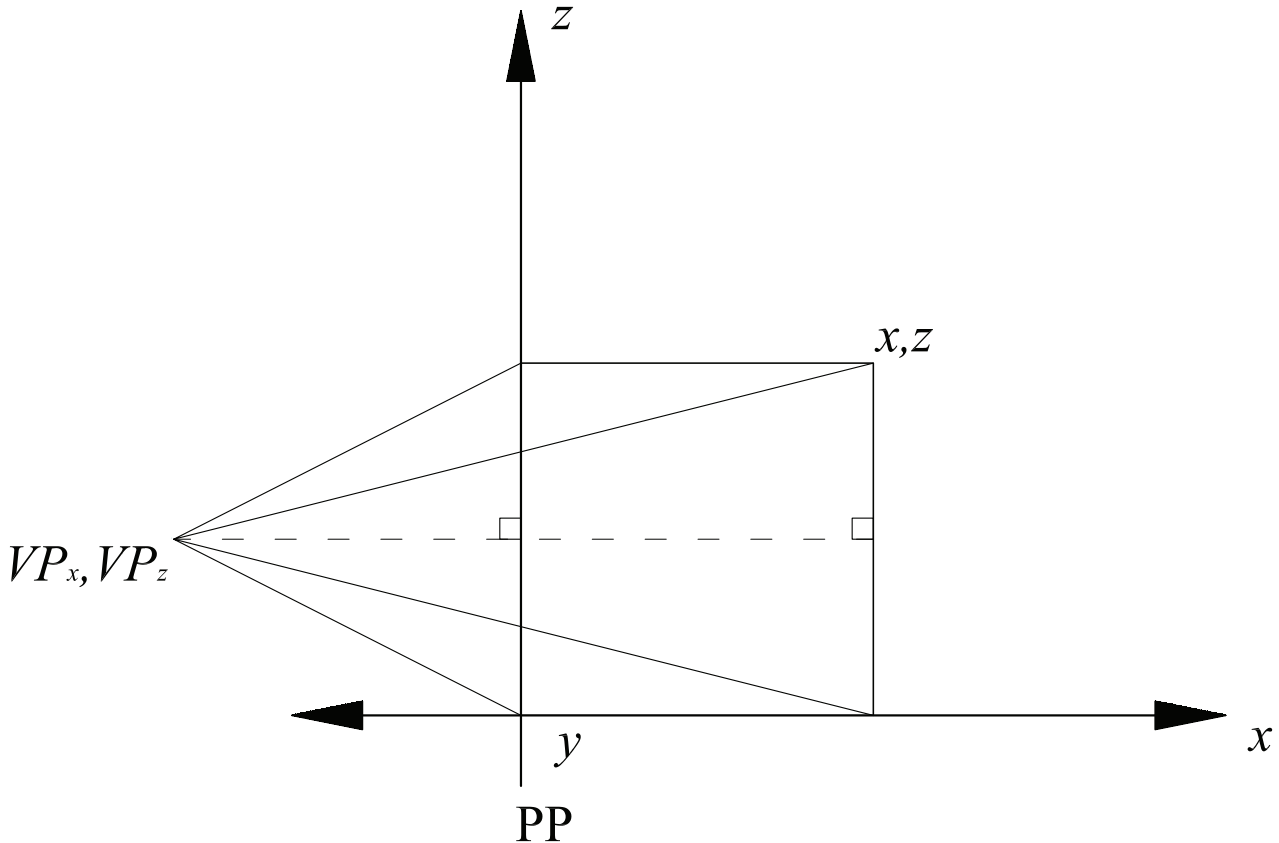


Figure 12.14 - Perspective projection viewed from front

And again by using the similar triangles method we come up with the following ratio,

$$\frac{z - VP_z}{VP_{dept h} + x} = \frac{PP_z - VP_z}{VP_{dept h}}$$

Which can be rearranged with PP_z the subject as follows,

$$PP_z = \left(\frac{z - VP_z}{VP_{dept h} + x} \right) VP_{dept h} + VP_z$$

As in this case the picture plane is on the yz axis, the $PP_x = 0$.

However the projected point that we get from using the derived formula above are based on the 3D coordinate system used. To change these projected coordinates into the 2D coordinate system used the following formula can be used.

$$\begin{aligned} x' &= -PP_y \\ y' &= PP_z \end{aligned}$$



IMPORTING DXF CIRCLES & ARCS PROBLEMS

My program only (internally) took points as input, but the input coming in had lines, circles and arcs. Lines were easy and I changed my program to suit them. I just projected the start point then the end point then connected the line again. But circles and arcs posed a problem. So what I did is I subdivided the circles and arcs into many short lines before the core projection algorithm. I used constant samples which is shown in figure 12.15. I could have used adaptive sampling, where more samples are taken where needed. I could have done this by using the derivative to find the slope and the slope would determine the sampling quality needed but I didn't have the time to implement this, so the efficiency of the algorithm was sacrificed. Also if you notice how in figure 12.15 the samples are taken at regular x intervals, I could have taken the samples at a given angle from the centre rather than an x value, as then I would have the same quality the whole way through.

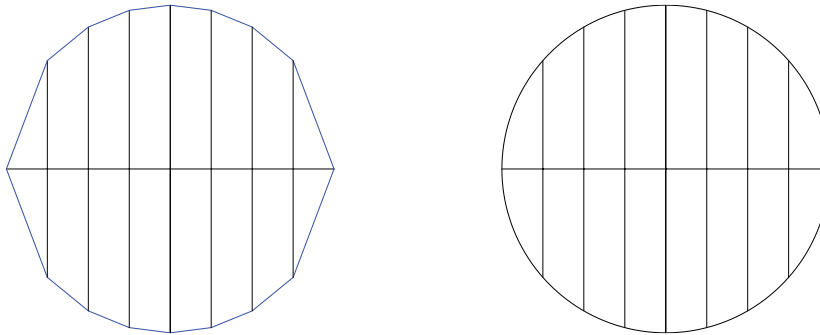


Figure 12.15 - Subdivision of circles

The next problem that arose is that when importing circles, they were not flat on a plane, they were rotated in all sorts of orientations. In mathematics this kind of circle would be defined as a sphere equation equal to a plane equation. An explanation of this is below;

The equation of a 2D circle is $x^2 + y^2 = r^2$ and the equation of a 3D sphere is $(x - i)^2 + (y - j)^2 + (z - k)^2 = r^2$. The equation of a line in 2D is $c = x + y$, the equivalent 3D equation $c = x + y + z$ is actually a plane in 3D. If you solve the simultaneous equation of the sphere and the plane then you only get the points where the two overlap, and if the plane passes through the centre you get a circle with the correct radius. However this can get mathematically complicated and computers are not so good at equation solving. So what DXF files (and many other formats) do when storing a circle in 3D is they store a 2D circle (centre x , centre y and radius (so it appears flat on the xy plane)), and a vector acting as the normal of the circle. This is shown in figure 12.16.

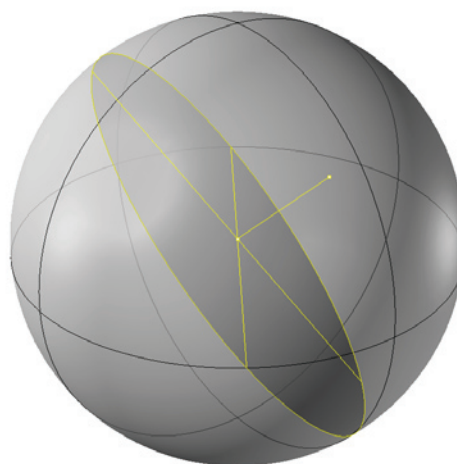


Figure 12.16 - Circle defined by centre, radius & normal

So I split the flat circle into many lines as described previously, then I applied a local rotation matrix to each point. The rotation angles were determined from the vector using simple polar-cartesian coordinate system conversion formulae. Now I had a 2D circle (actually just a group of lines joined) correctly defined in 3D.



Another problem, which was probably the most difficult one to solve arose due to the different coordinate systems used (WCS (world coordinate system) & OCS (object coordinate system)), and this took me much time, research, knowledge and problem solving ability to solve. I did not have a problem with lines as they are always represented in WCS, but circles and arcs use OCS, this was the problem. As it is a complicated problem details are not fully documented here, however I will say that I used the so called "Arbitrary Axis Algorithm" to solve the problem, which I researched about from the internet and the *AutoCAD 2007 DXF Reference*¹, which involved calculating an arbitrary axis from the normal vector data given, then the world coordinates could be calculated from the object coordinates given as the centre (*the equations to do this can be seen in the code*). I did need an extensive knowledge of matrices, inverse matrices and three dimensional vectors, which is university level mathematics. I did utilise AutoLISP to help me during the debugging stages of production. Because some of the functions that I had to reinvent were incorporated in AutoLISP so I used it to get sample data and correct answers to test my code.

Arcs were processed the same way circles were, however only an arc of the whole circle was converted to lines. In the DXF file the arc was defined using the same properties as circles, except a start and end angle were present. From this start and end angle I determined (by splitting the circle into quadrants) the parts of the circle to omit. How I did this is documented in the next section.

DRAWING OF ARCS

As mentioned above arcs in the DXF file contained the same data as circles with the addition of a start and end angle of the arc. So during the process of subdividing the arc into many lines like with the circle I implemented algorithms that calculated which lines to omit and fill to make the arc. To do this I dealt with the arc's base circle in separate quadrants. (see figure 12.17)

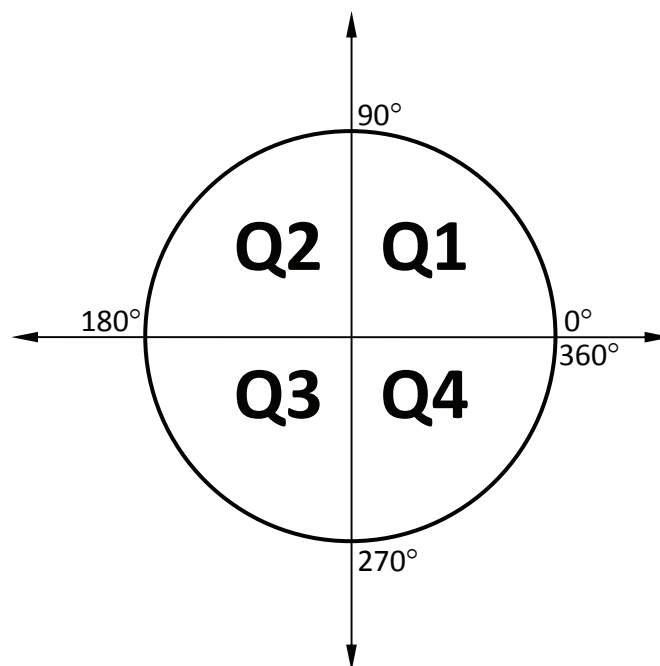


Figure 12.17 - Quadrants of circle

So in my algorithm I drew in lines for quadrant 1, then 2, then 3 and then 4. For each quadrant there are four cases. Either both the start and end angles of the arc are in the specified quadrant, neither angles are in the specified quadrant, the start angle only is in that quadrant or the end angle only is in that quadrant.

In the analysis that follows are simplified diagrams. Figure 12.18 shows a more detailed version of these diagrams.



¹ From <http://www.autodesk.com/dxf/>

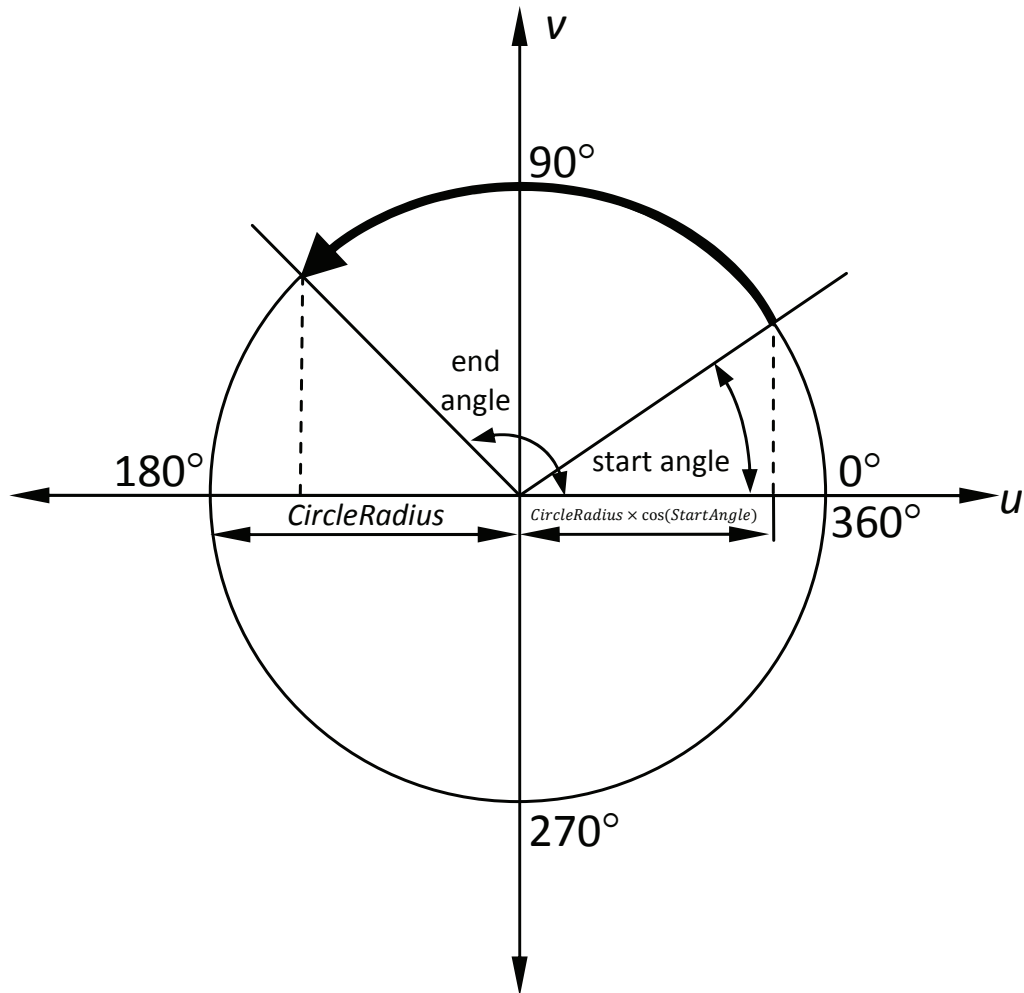


Figure 12.18 - Arc Details

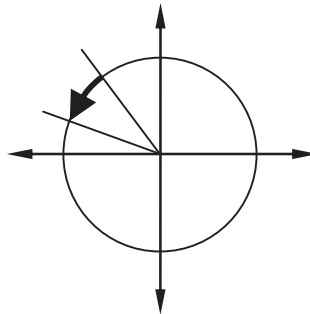
I developed a function in Visual Basic which given an angle and quadrant will return a true/false depending if the given angle is in the given quadrant. This function is used extensively in the following. Each quadrant is analysed separately as follows (where true is when the angle is in the quadrant and false is when the angle is not in the quadrant).



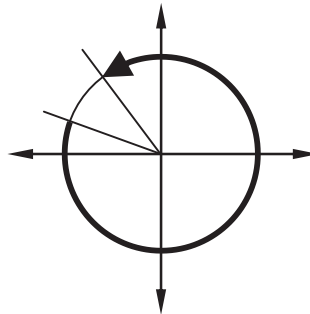
Quadrant 1

Both Angles
 $START = TRUE$
 $END = TRUE$

This has two possibilities. As shown below. Which option can be determined by examining if $StartAngle > EndAngle$.

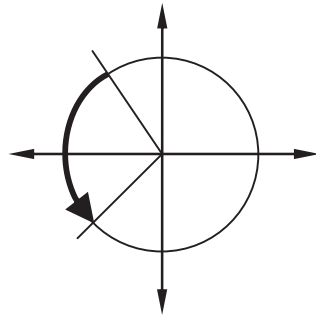


Fill from $x = CircleRadius \times \cos(EndAngle)$ to
 $x = CircleRadius \times \cos(StartAngle)$



Fill from $x = 0$ to $x = CircleRadius \times \cos(EndAngle)$. Then from $x = CircleRadius \times \cos(StartAngle)$ to $x = CircleRadius$

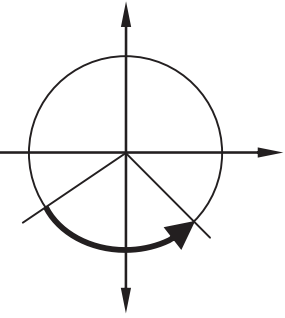
Start Angle ONLY
 $START = TRUE$
 $END = FALSE$



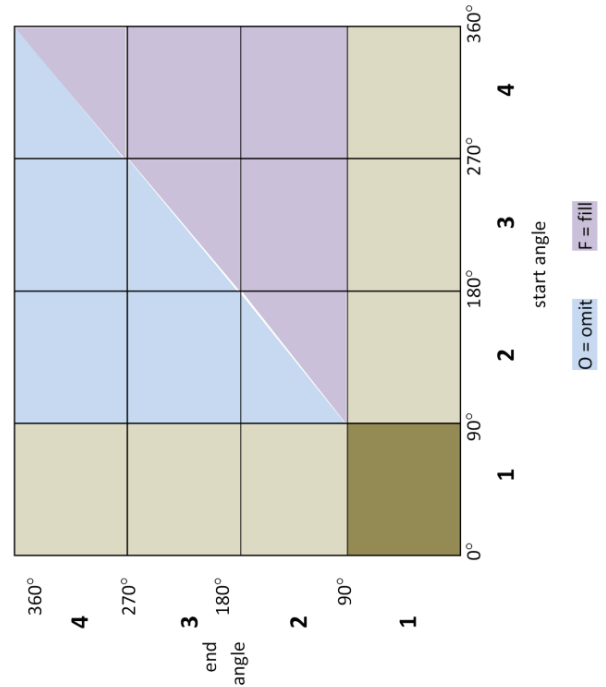
Fill from $x = 0$ to
 $x = CircleRadius \times \cos(StartAngle)$

End Angle ONLY
 $START = FALSE$
 $END = TRUE$

This is impossible. However may occur due to small rounding error when start angle gets pushed into fourth quadrant. So in this case fill from $x = CircleRadius \times \cos(StartAngle)$ to $x = CircleRadius$



Quadrant 1



The graph to the left shows how these two possibilities can be calculated. It simplifies down to, if $start\ angle > end\ angle$ then fill the entire quadrant, otherwise don't fill it.

This has two possibilities. Fill the entire quadrant or fill none of the quadrant.

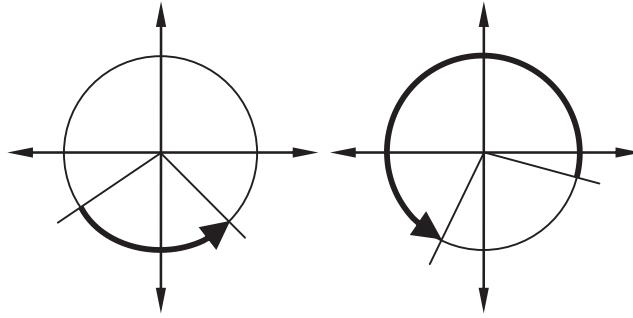
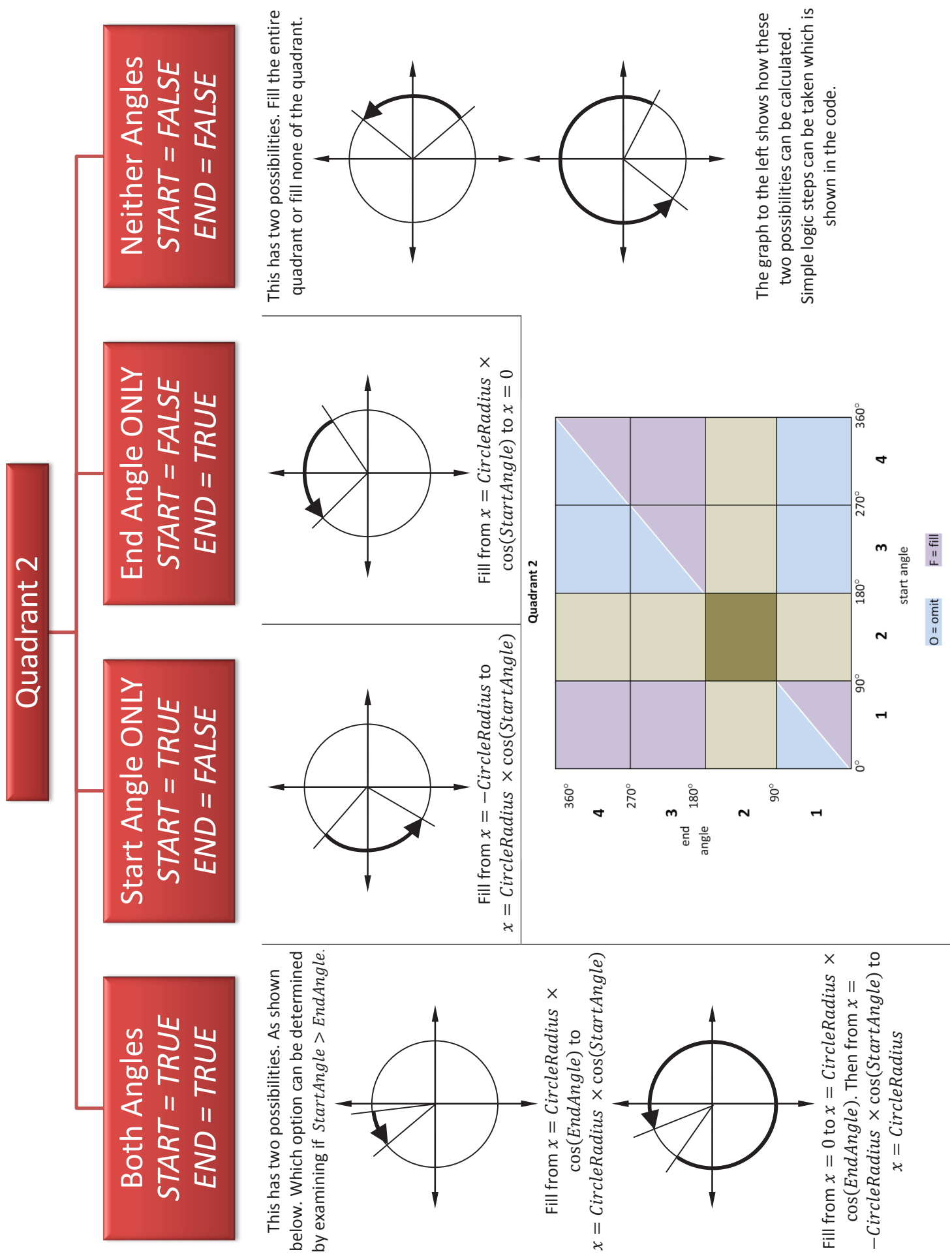
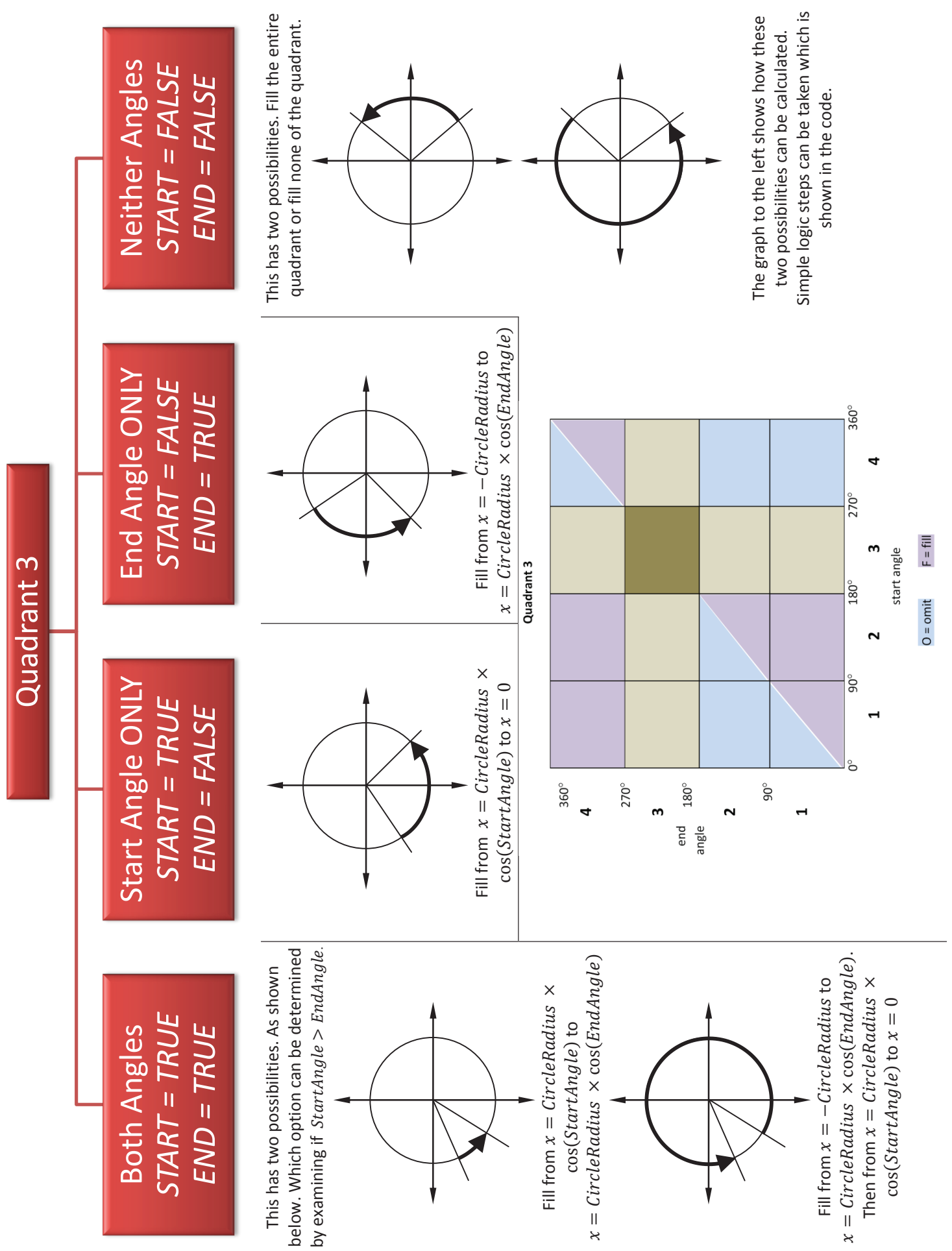
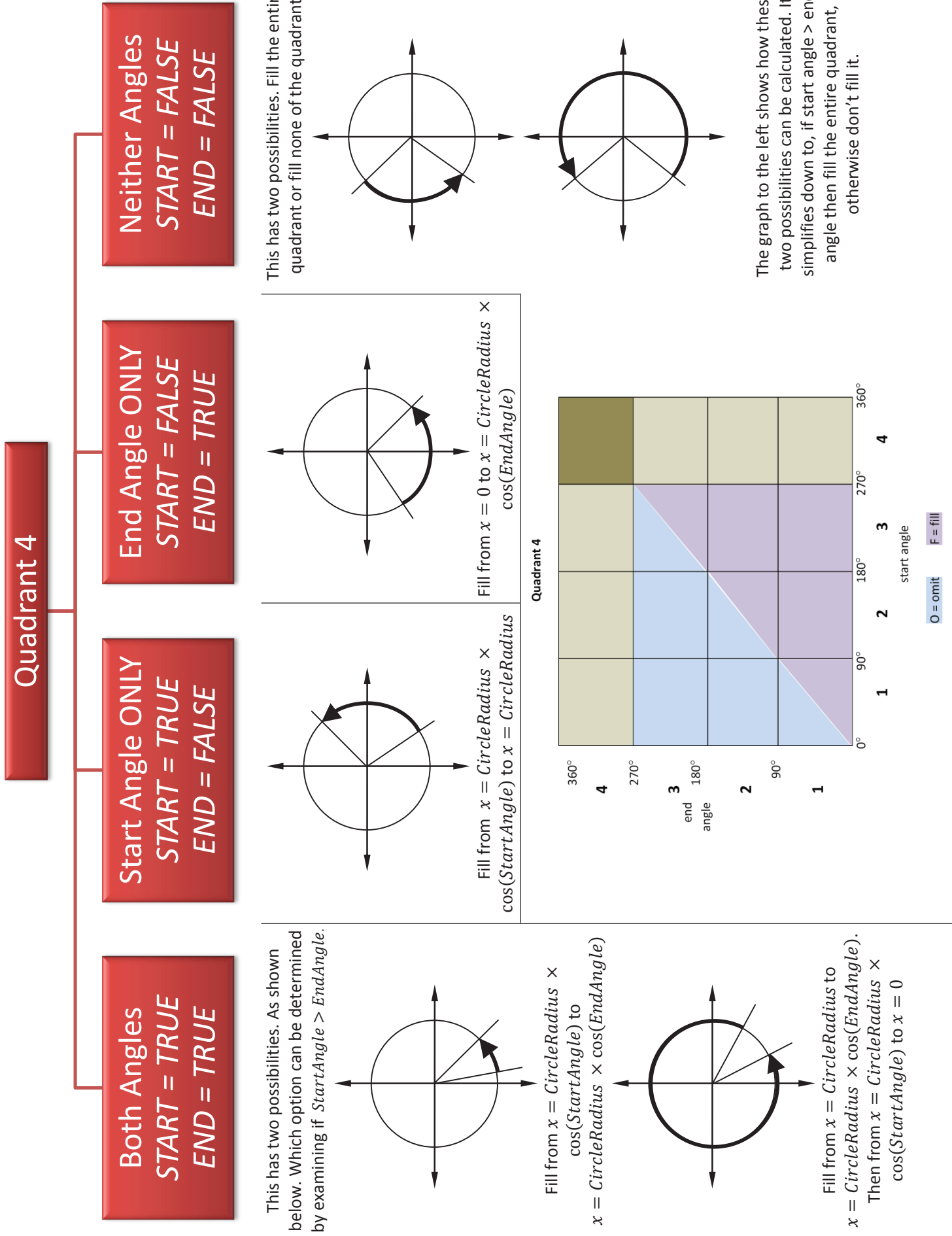


Figure 12.19 - Quadrant 1 Possibilities







CALCULATION OF AXONOMETRIC SCALES

In figure 12.23 the five variables that *ViewIt* takes into account when applying the *projection* are shown.

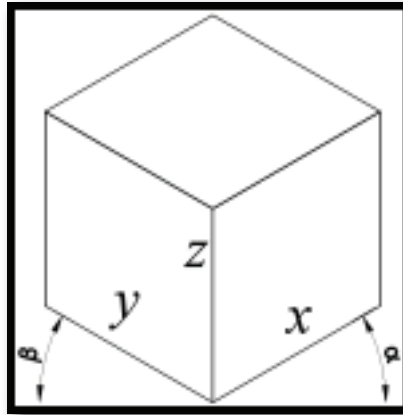


Figure 12.23 - Five Variables of *ViewIt* Projection

When the preset projection is set to either dimetric or trimetric, then alpha and beta are specified but the scales, x, y, z are calculated to get a correct dimetric or trimetric drawing. The scales are calculated with the formulae in figure 12.24.

$$x \text{ scale} = \sqrt{\frac{\tan \beta (1 + \tan^2 \alpha)}{\tan \alpha + \tan \beta}}$$

$$y \text{ scale} = \sqrt{1 - \tan \alpha \times \tan \beta}$$

$$z \text{ scale} = \sqrt{\frac{\tan \alpha (1 + \tan^2 \beta)}{\tan \alpha + \tan \beta}}$$

Figure 12.24 - Axonometric Scale Formulae

Note: These formula were not derived. I got them from an Engineering Drawing Text Book, see reference [2]

POSSIBLE IMPROVEMENTS

HIDDEN LINE REMOVAL

The most dominating thing that I feel *ViewIt* lacks is hidden line removal, or hidden line identification so that the linetype can be changed. There are very few circumstances (I can't think of any) where you would want a drawing with no discrimination between hidden and visible lines. Also as previously mentioned currently *ViewIt* takes lines as input from this alone hidden line removal can not be done, because it is a wireframe the faces are hollow, thus you can see through to the lines behind. The first thing I would need to do to implement hidden line removal would be to develop the importing capabilities to import the 3D data as surfaces not just lines. This is nowhere near as easy to importing lines. I cannot just use a mesh as that would result in lines that are a result of triangulation, which are unwanted, I would need to use a CAD format such as IGES, STEP or open-NURBS' *.3DM format. However these formats are much more complex compared to DXF. For a start DXF is in ASCII whereas 3DM is binary.

I did some research into existing hidden line removal algorithms and found some simple ones and some more longer and complex algorithms, or methods should I say, I found few actual algorithms. However the simple methods yielded incorrect results in some circumstances. Some of the methods I found that may rise difficulties due to the projection I used, as many methods were based on representing objects by projecting them on to a plane, however the method I used for projection is slightly different to that. After research I am confident that if I were to implement hidden line removal I would use a method that used a z-buffer. When the 3D coordinates are changed to two



to lie on a projection plane the z coordinate of the projection plane becomes zero, that is the point of projection. All the points lie on the one flat surface, so without any z information you could argue that as all the points have z equal to zero then no lines are hidden and thus you cannot calculate and hidden lines as they are all visible. The z -buffer gives each of the projected points a z value that is not just when defining that point or positioning it in space, but it tells us the depth of the point from its 3D origin. This z -buffer could then be used to determine hidden lines as we could calculate which points are further back and hidden behind certain faces or planes.

So I came up with an algorithm to calculate the z -buffer, this gives an accurate z -buffer given the two angles and three axis scales that are required by the projection algorithm. The derivation of it is as follows.

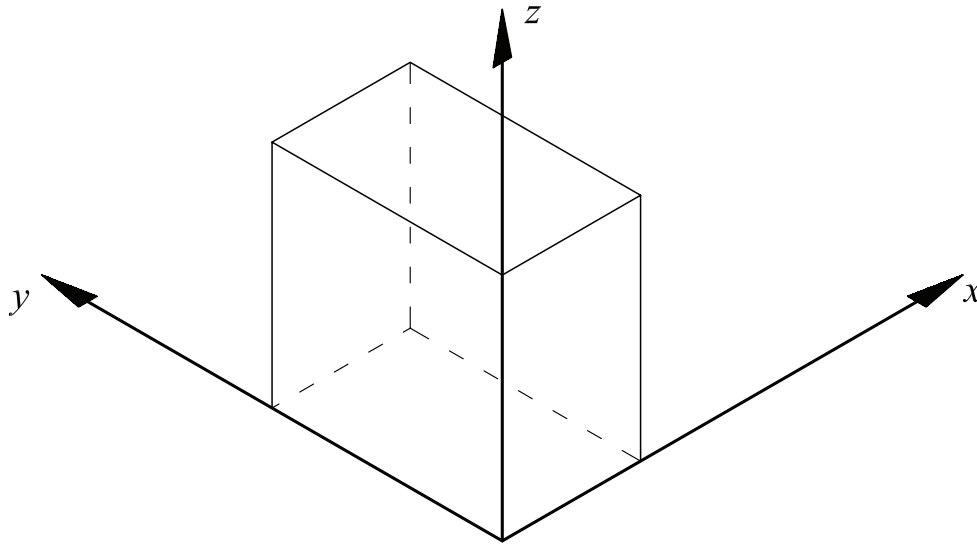


Figure 12.25 - Block in 3D Coordinate System

If $\alpha = 0^\circ$ then all points on xz plane will have a z depth of 0. It can also be noted that $\sin 0^\circ = 0$. If $\alpha = 90^\circ$ then all points on xz plane will have a z depth of x . It can also be noted that $\sin 90^\circ = 1$. So for all the points on xz plane, z depth = $x \sin \alpha$. Similarly for all the points on yz plane, z depth = $y \sin \beta$. These can now be considered components and so for any point in xyz space the z depth is the addition of these two linear components. The previous assumes a scaling of one on each axis. So the final result including the scaling factors is below,

$$z\text{-depth}_{(x,y)} = S_x x \sin \alpha + S_y y \sin \beta$$

This result works for both projections such as isometric and oblique, which is what it should. This z -depth information is the basis for z -buffer hidden line removal algorithms.

TWO POINT PERSPECTIVE ALGORITHM

Once I had derived and implemented one point perspective algorithms the next logical step would be to derive and implement two point perspective algorithms. Figure 12.26 below, shows points on the model's x and y axis being projected onto the picture plane (PP). With two point perspective points on the x plane will be projected to the right vanishing point (RVP) and points on the y plane will be projected to the left vanishing point (LVP). The coordinate system used is x, y, z represents the point in model space, prior to projection and x', y', z' are the projected points on the PP from the respective x, y, z point. As the projected point will be on the PP, z' will always be zero. Also the VP's coordinates are in the PP coordinate system.

Now that the coordinate systems have been addressed, I can continue with the derivation. So as shown in the diagram, the further back along the x axis the point is, the smaller the projected increment is, which is what is supposed to happen. But what if the point is somewhere in the xy space? What if the point is not on the x or y plane but somewhere else in space? Well I figured, just like vectors how we split a vector into the x component and the y component, I could take



the x and y component of the point in xy space and project the x component to the RVP and the y component to the LVP. Now as x' of the points projected to the LVP will be negative, to combine these components back to incorporate a point in xy space simply add the x' from the RVP projection and the x' from the LVP projection. This will be used later on, but now I need to work out how to project the the left and right vanishing point.

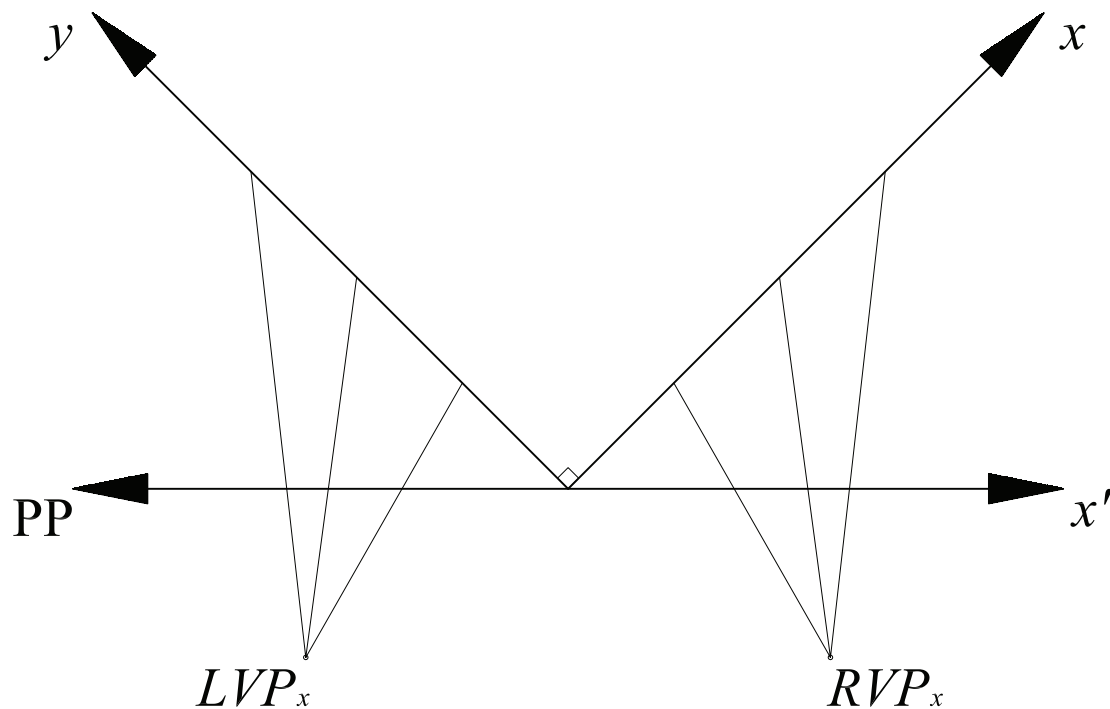


Figure 12.26 - Two Point Perspective Projection Viewed from Top

Figure 12.27 below shows a point A, which has coordinates in the model coordinate system of x, θ, z . As in this case we are just looking to find the projected x' , we need not worry about z . E is the RVP. So $OA = x$ and we are trying to find an expression for OC which is x' . Also note that this is based on the fact that we are using an object rotation of 45° . I probably should have derived a general result for θ° , however that is the next step.

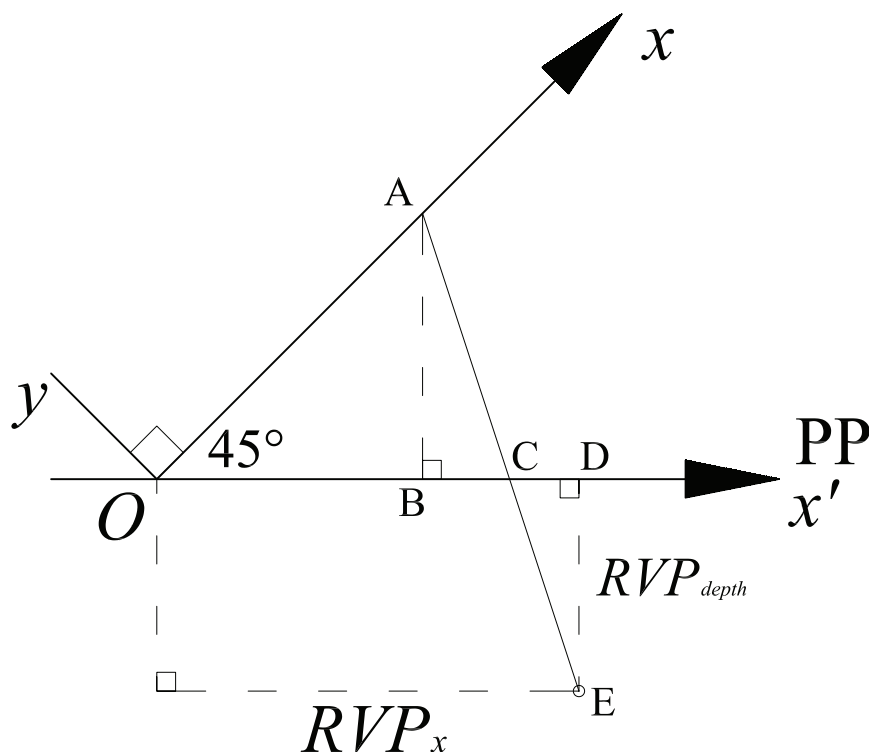


Figure 12.27 - Projection of a point on the x plane



We want to find OC, but as:

$$\begin{aligned} OC &= OB + BC \\ &= AB + BC \quad (\text{equal sides of isosceles triangle}) \\ &= \frac{OA}{\sqrt{2}} + BC \quad (\text{using Pythagoras theorem or trigonometry}) \\ &= \frac{x}{\sqrt{2}} + BC \end{aligned}$$

So now we need to find BC. Once we find BC we can find OC.

The two triangles $\triangle ABC$ & $\triangle CDE$ are similar. Here is the proof:

$$\begin{aligned} &\text{In } \triangle ABC \text{ \& } \triangle CDE \\ \angle ABC &= \angle EDC = 90^\circ \quad (\text{given}) \\ \angle ACB &= \angle DCE \quad (\text{vertically opposite angles}) \\ \therefore \triangle ABC &\parallel \triangle CDE \quad (\text{equiangular}) \end{aligned}$$

So as they are similar we can get a ratio of the sides:

$$\begin{aligned} \therefore \frac{AB}{DE} &= \frac{BC}{CD} = \frac{AC}{CE} \quad (\text{corresponding sides in similar triangles}) \\ AB \times CD &= DE \times BC \\ CD &= \frac{DE \times BC}{AB} \end{aligned}$$

As $BC + CD = BD$ (sum of parts)

$$\begin{aligned} BC &= BD - CD \\ BC &= BD - \frac{DE \times BC}{AB} \end{aligned}$$

We have found BC, but not all the variables are known.

$$\begin{aligned} OD &= RVP_x \\ OD &= OB + BD \\ &= AB + BD \end{aligned}$$

$$\begin{aligned} AB + BD &= RVP_x \\ BD &= RVP_x - AB \\ &= RVP_x - \frac{x}{\sqrt{2}} \end{aligned}$$

Now we have found BD, and we can substitute it back into the formula we got above.

$$\begin{aligned} BC &= RVP_x - \frac{x}{\sqrt{2}} - \frac{DE \times BC}{AB} \\ BC + \frac{DE \times BC}{AB} &= RVP_x - \frac{x}{\sqrt{2}} \\ BC \left(1 + \frac{DE}{AB}\right) &= RVP_x - \frac{x}{\sqrt{2}} \\ BC &= \frac{\left(RVP_x - \frac{x}{\sqrt{2}}\right)}{\left(1 + \frac{DE}{AB}\right)} \end{aligned}$$



So we now have BC and we can substitute this back into the formula that we got at the very top.

$$OC = \frac{x}{\sqrt{2}} + BC$$

$$OC = \left(\frac{x}{\sqrt{2}} \right) + \left[\frac{\left(RVP_x - \frac{x}{\sqrt{2}} \right)}{\left(1 + \frac{DE}{AB} \right)} \right]$$

Now as:

$$DE = RVP_{depth}$$

$$AB = \frac{x}{\sqrt{2}}$$

$$OC = PP_x$$

$$PP_x = \left(\frac{x}{\sqrt{2}} \right) + \frac{\left(RVP_x - \frac{x}{\sqrt{2}} \right)}{\left(1 + \frac{RVP_{depth}}{\left(\frac{x}{\sqrt{2}} \right)} \right)}$$

$$= \frac{x}{\sqrt{2}} + \frac{\left(RVP_x - \frac{x}{\sqrt{2}} \right)}{\left(1 + \frac{RVP_{depth} \sqrt{2}}{x} \right)}$$

This is for points strictly on the x plane. So for points on the y plane:

$$PP_x = - \left[\frac{y}{\sqrt{2}} + \frac{\left(LVP_x - \frac{y}{\sqrt{2}} \right)}{\left(1 + \frac{LVP_{depth} \sqrt{2}}{y} \right)} \right]$$

As we said earlier for a point in xy space, simply add the two components. So for a point in xy space:

$$PP_x = \left[\frac{x}{\sqrt{2}} + \frac{\left(RVP_x - \frac{x}{\sqrt{2}} \right)}{\left(1 + \frac{RVP_{depth} \sqrt{2}}{x} \right)} \right] - \left[\frac{y}{\sqrt{2}} + \frac{\left(LVP_x - \frac{y}{\sqrt{2}} \right)}{\left(1 + \frac{LVP_{depth} \sqrt{2}}{y} \right)} \right]$$



We now have PP_x . The next thing we need to do is calculate PP_y . My algorithm assumes that the elevation of the LVP and the RVP is the same. This greatly simplifies the process and this is almost always the case when doing perspective by manual hand methods, (mechanical or measuring point).

The derivation is the same as for one point perspective. So we can use that derived formula and change the variable names. Except that where x was used before we now need to use $\frac{x}{\sqrt{2}} + \frac{y}{\sqrt{2}}$, this is because the object is rotated 45° .

$$PP_y = \left(\frac{z - RVP_y}{RVP_{depth} + \left(\frac{x}{\sqrt{2}} + \frac{y}{\sqrt{2}} \right)} \right) RVP_{depth} + RVP_y$$

This same principal could be applied to three vanishing points, to generate three point perspective. However as Rhinoceros can already generate three point perspective there was no need to include this. This two point perspective algorithm was implemented into ViewIt however it did not give expected results. Either my mathematical derivation is wrong or I implemented it incorrectly. Either way I did not have the time to find the problem. This thing I noticed was that as I made the VP close to the PP then projected drawing became distorted in a way that was not a result of the perspective distortion. An extreme value of VP_z illustrates the problem as shown in figure 12.28.

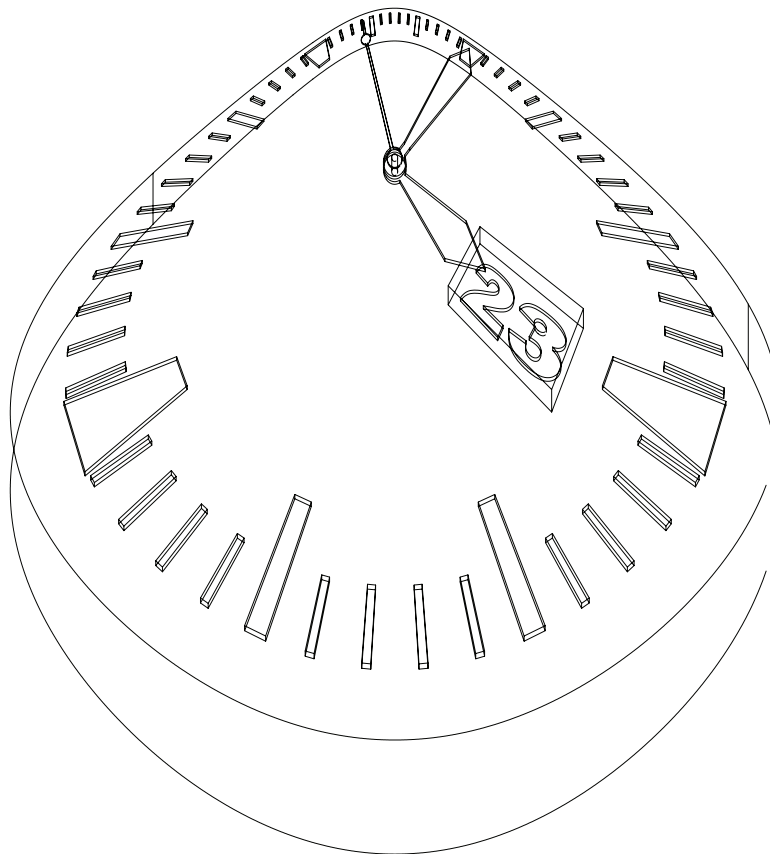


Figure 12.28 - As I made the VP closer to the PP, the drawing became distorted in a way that was not expected

By further examination and testing of different variables for the VP, I found that this was the perspective algorithm in action. As this is obviously wrong and as lines that should be straight became curved, I can conclude that either my perspective projection algorithm is wrong or my implementation is wrong.



RASTER IMAGE SUPPORT

In theory this is not very hard to implement, however it is much harder in practice and I did not have the time to implement it. Raster image support would mean that ViewIt would project, pixel by pixel, any raster image plane (AutoCAD only) in the projected coordinate system. This means that if I added to my 3D model, or 3D wireframe, in AutoCAD a raster image that is the DD logo that appears on the face, would be projected just like the rest of the object into the projected coordinate system. This means that the logo will be correctly projected into the projected coordinate system.

This sounds easy, and if I could get a Visual Basic library that reads/writes various raster image formats then all I would need to do is use the library to read the pixels into a matrix, apply the projection matrix to each pixel, and then use the library to write the pixel matrix back to the raster image file. Then I would need to get ViewIt to also recognise to put the image reference in the DXF file and also position it correctly. In theory, this sounds easy. However even with a read/write library reading and processing the pixels would be difficult. However the most dominating thing that would make it difficult to me is that due to the projection of pixels, many problems will arise. For a start I need to take into account scaling of an image which is not that difficult. However to avoid aliasing and other artifacts that make the image non-smooth and unpleasant could be removed by resizing algorithms such as bicubic interpolation scaling. Also some projections will change the relation of the relative pixels. This means that I cannot directly map each pixel in the object coordinate system to a pixel in the projected coordinate system. Without exploring and documenting the pages of information I could write about this I will say that I am not an expert in the areas of sampling, aliasing and MIP mapping which are the areas this problem is related to, and therefore implementing this would be difficult. Also for orthogonal and planometric projections I could achieve this by manually placing the raster image in the 2D drawings and translating, rotating and scaling it in AutoCAD. For all other projections I could get the projected 2D raster image by translating, rotating, non-isotropically (non-uniformly) scaling and shearing it, however this could not be done in AutoCAD. This possible improvement is also related to another thing that I could have benefited from, but never had the time or knowledge to implement, see Evaluation → Ideas That Never Made Reality → Rendering in Less Common Projections, page 90.

AVOID SAMPLING CIRCLES WHEN UNNECESSARY

I got ViewIt to sample all circles. However this is only necessary for some projection types. For example in orthogonal circles that are parallel to one of the main axis planes stay true circles when projected. This means that the circle was sampled unnecessarily. This is bad because the program will have taken longer to process, will produce much larger files, manipulation in AutoCAD is now made more difficult, and also hidden line removal is much more tedious. Also for oblique and one point perspective, circles on the front face also stay as true circles when projected, similarly for circles in planometric on the top face. Furthermore, circles in isometric projection become ellipses, which also need not be sampled. However, my philosophy of production of ViewIt was that I needed to get results quickly. Things that weren't necessary and would consume much time should be briefly noted but not implemented.

DRAWINGS CREATED USING VIEWIT

I used ViewIt for the creation of drawing number 79 and 82. You might be thinking why did I go to all this work of creating ViewIt if I only created two drawing from the program. Well there are many reasons that I have already mentioned. But since I went to all this trouble I have shown some raw output drawings that of parts of my watch that were generated from ViewIt but not included as part of the production, simply because these drawing types were not needed, the orthogonal, isometric and perspective projections were sufficient. I also show these as evidence of what exactly ViewIt can produce.



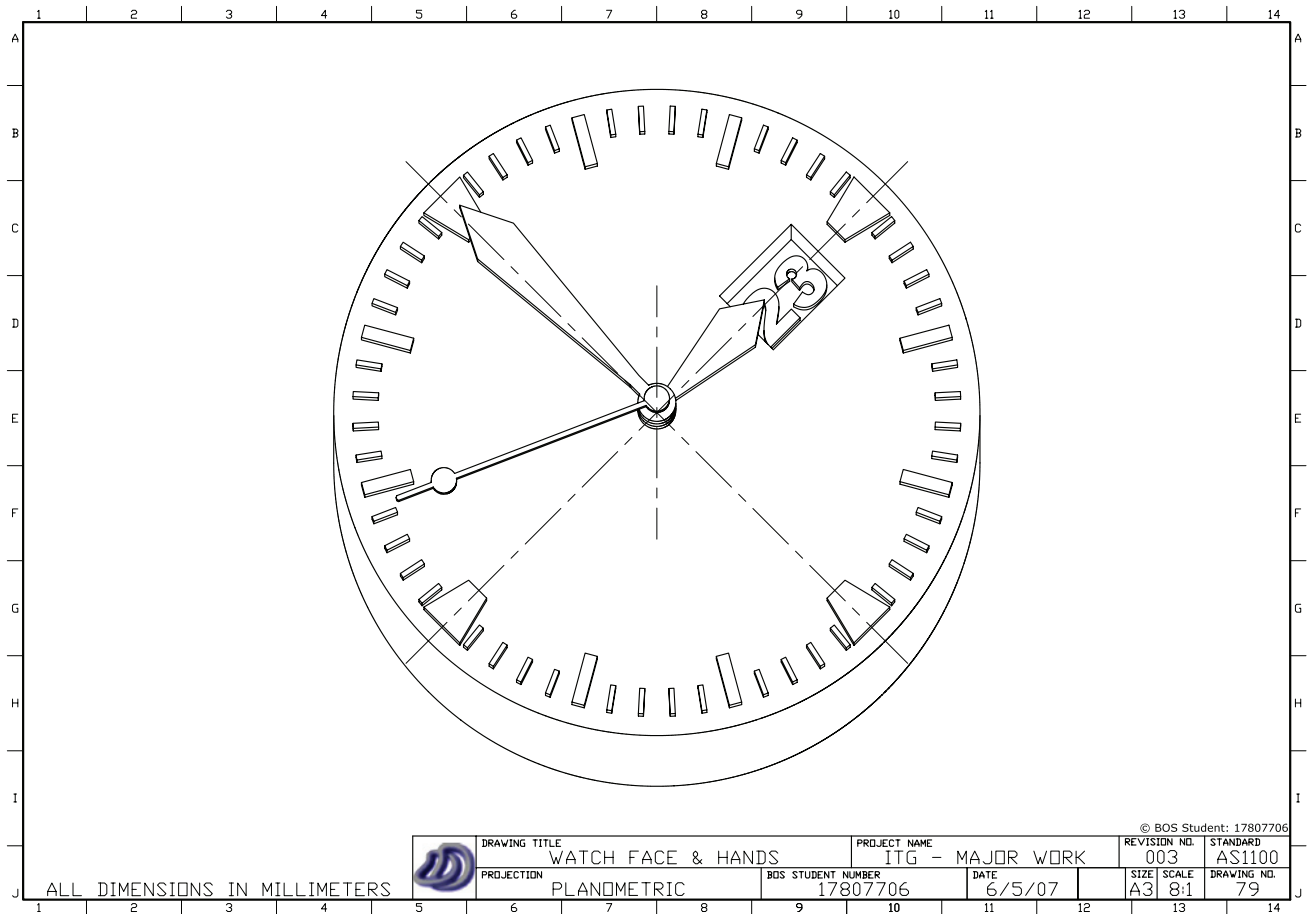


Figure 12.29 - Drawing No. 79

The following are raw output and if were to be actual drawings, I would very easily and quickly apply correct AS1100 standards (easily done with pre-made template), title block (easily done with pre-made template) and remove hidden lines.



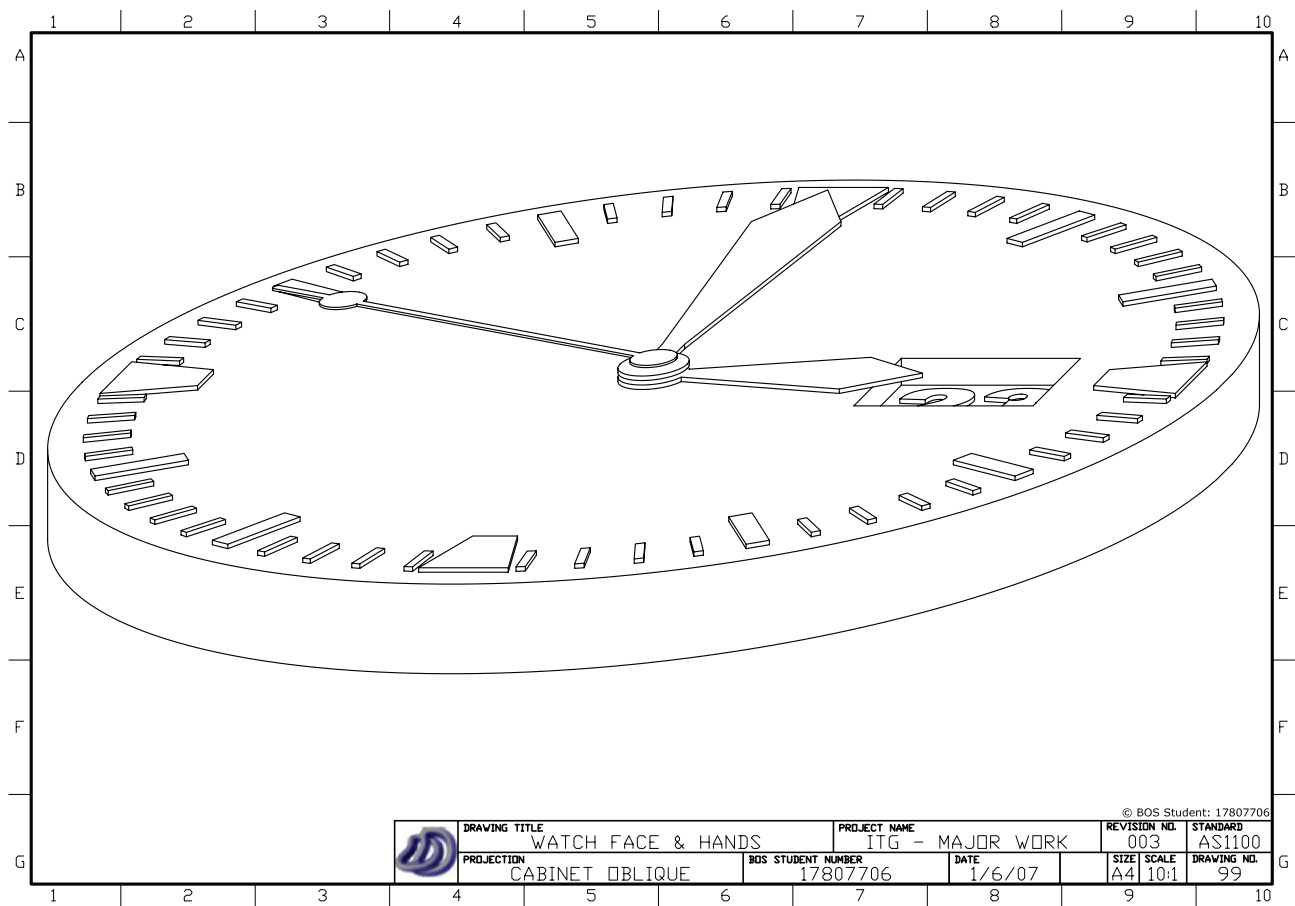


Figure 12.30 - Drawing No. 99, Based on Generated Cabinet Oblique Drawing from ViewIt

By comparing Figure 12.30 and Figure 12.31 you can see why cabinet oblique looks more realistic and is preferred over cavalier oblique.

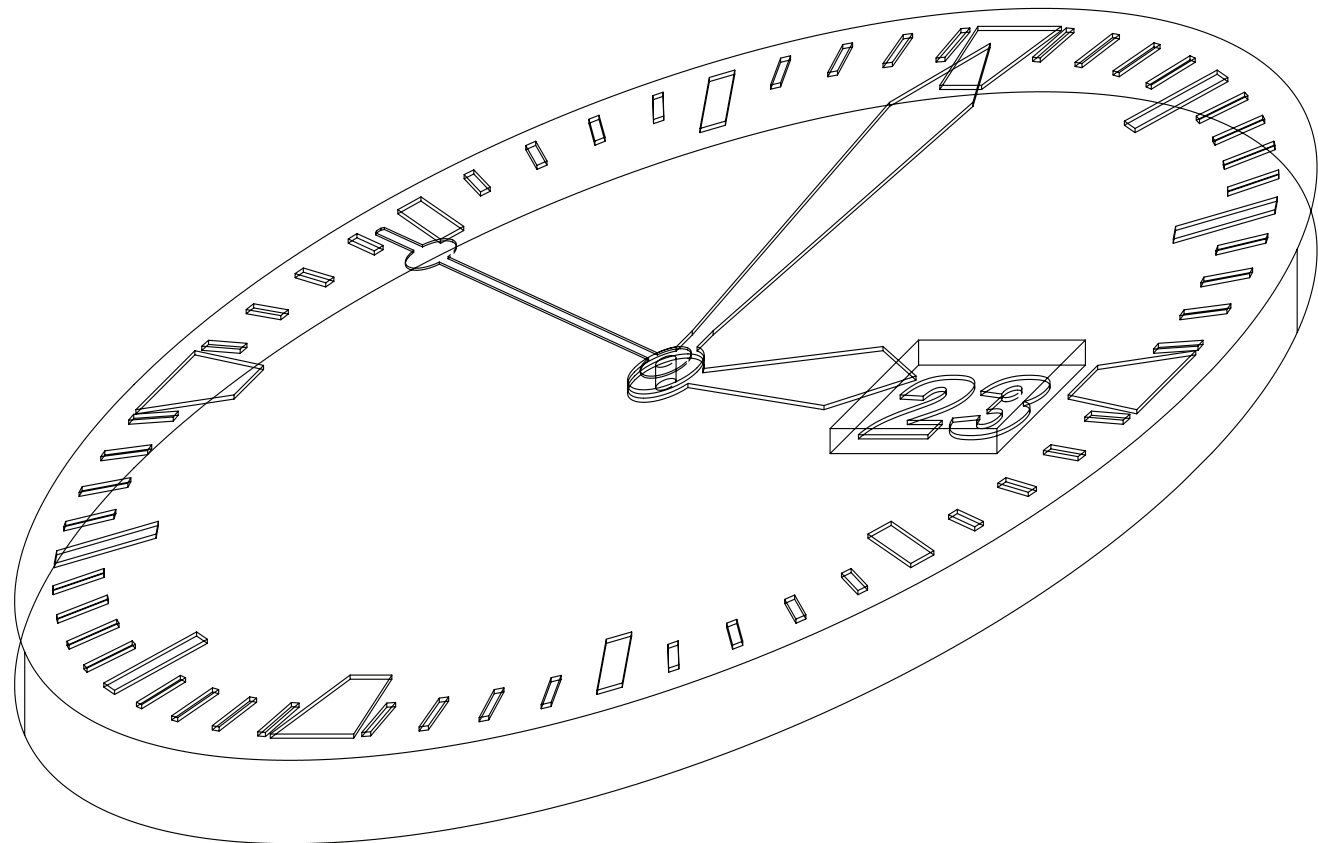


Figure 12.31 - Generated Cavalier Oblique Drawing from ViewIt



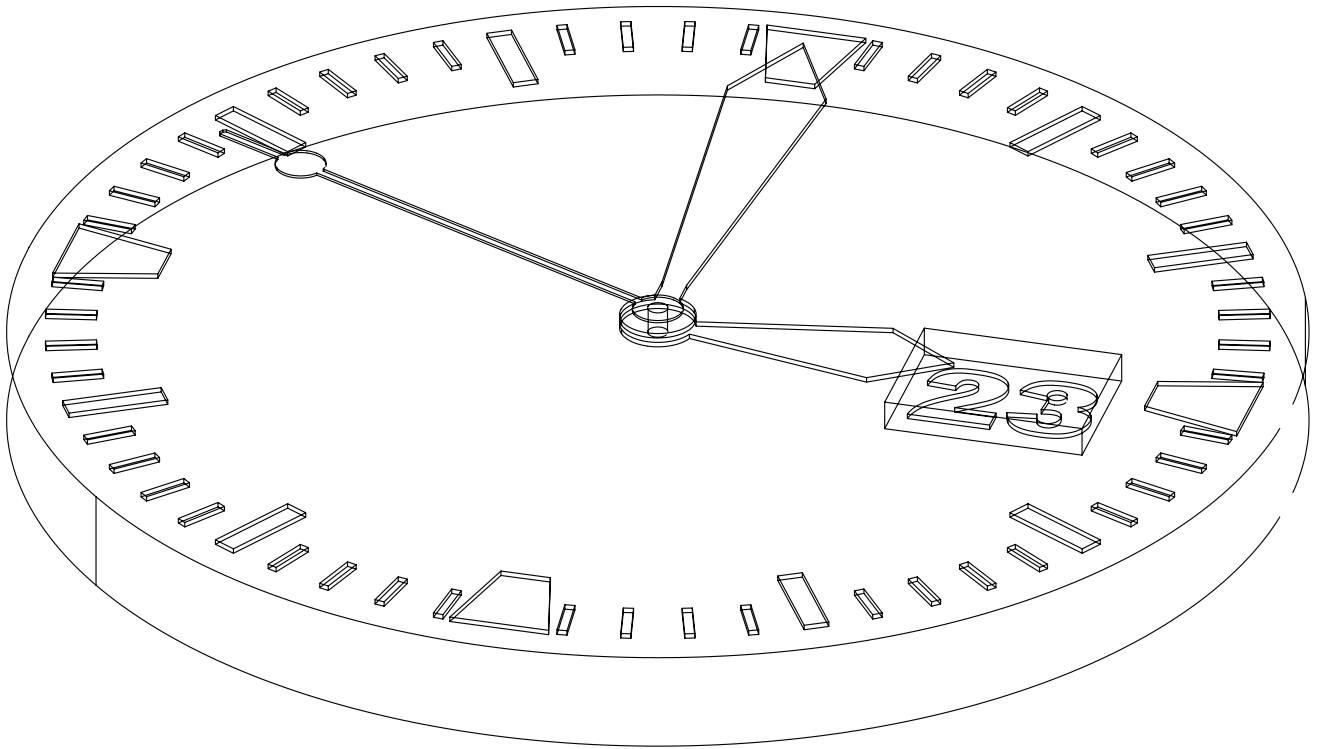


Figure 12.32 - Non-standard view generated using a set azimuth and elevation

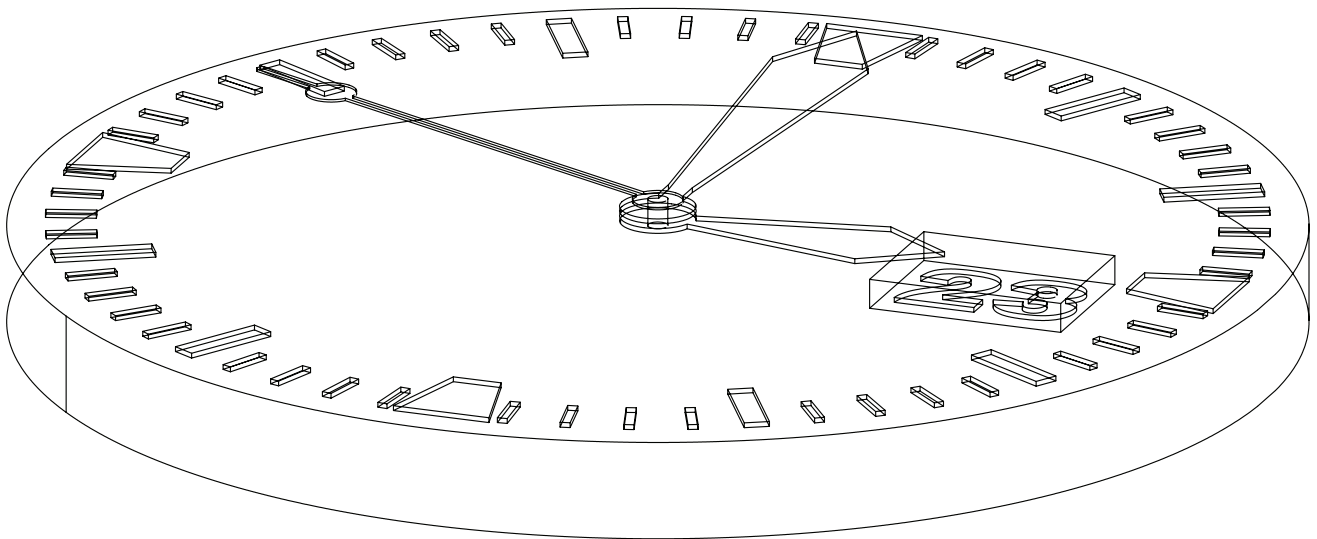


Figure 12.33 - Generated Dimetric Drawing from ViewIt



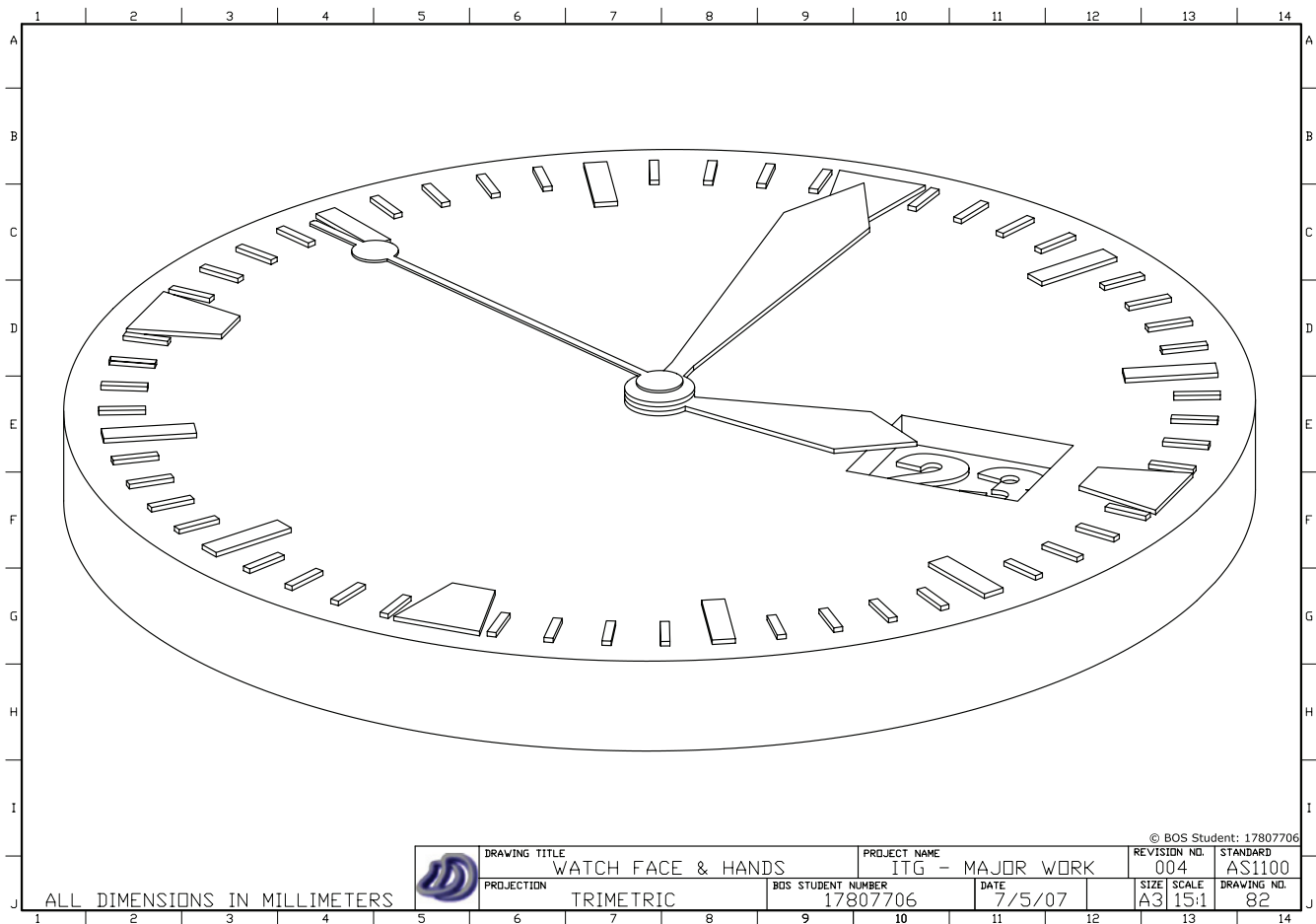


Figure 12.34 - Trimetric Drawing based on output data from ViewIt

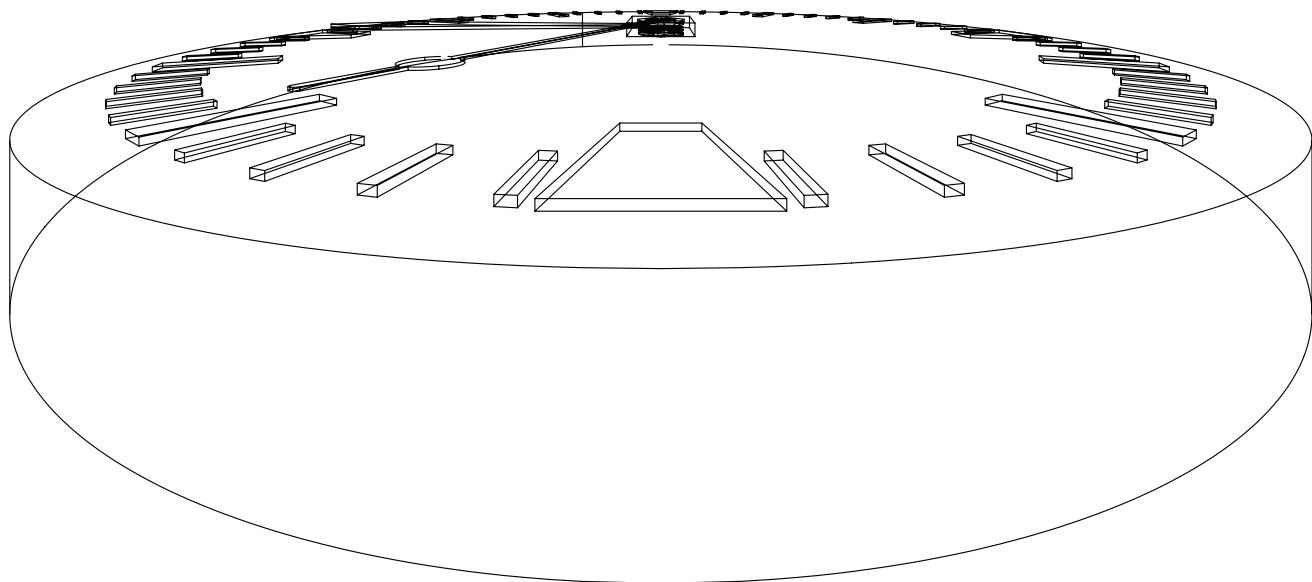


Figure 12.35 - Generated One Point Perspective Drawing from ViewIt, with VP extended above top centre



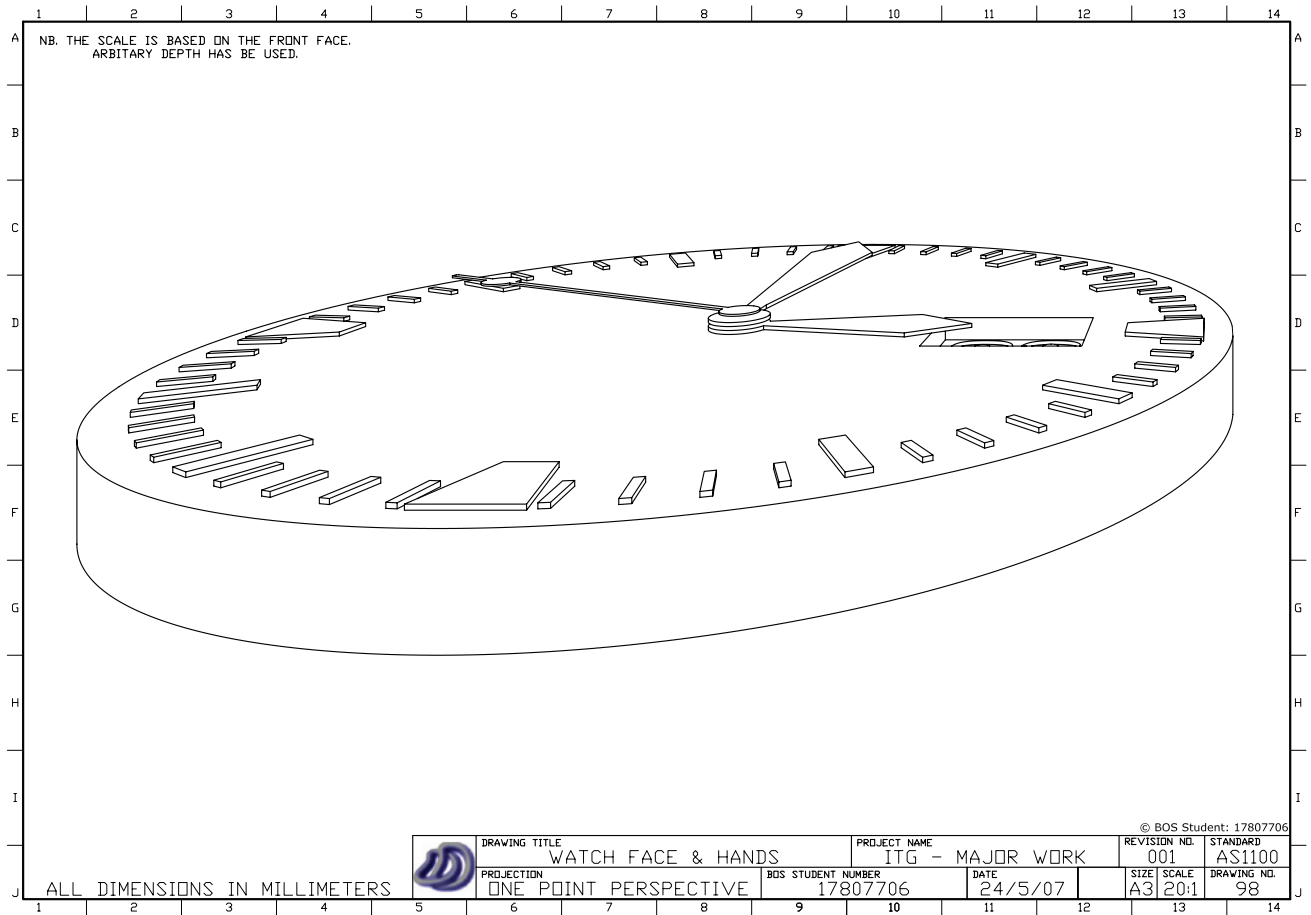


Figure 12.36 - Drawing No. 98. One Point Perspective Drawing created using ViewIt



REFERENCES

Any aspects of this project which were either partially or fully not completed by me, 17807706 are cited with footnotes or with a specific mention next to the work not completed by me. Therefore ALL graphics, figures, text, layout design, and other not documented or cited (either with footnotes, below caption) as being either developed, owned and/or copyrighted by another individual or organisation was produced by me, 17807706. This also includes the production component of the major project.

Some exceptions include in the Selection and Justification section images of software boxes came from various internet sources. Logos on the finance plan are from their respective web sites. These images may be copyright protected, however I believe they have been used here in accordance with the *Copyright Act 1968*. The exclamation logo on page 109 has been sourced from Microsoft Windows.

A note about screenshots, all of the screenshots (except screenshots of ViewIt) are of commercial software. However all the work shown in the screenshot was produced by me.

As mentioned on page 109, Production Item No. 102 (Physical Model) was partially outsourced to a commercial company. I produced the .stl file of the model which was sent to the company. The company then printed the .stl file using their ZCORP 3D printer. Details of the company are below.

3H Engineering Pty. Ltd.

(details unavailable)

Also the printing of some of the full colour A3/A2 production items was done by a commercial company (Officeworks). This was done as their equipment provided better quality than what equipment I had available. All the work contained in these items was however produced by me, 17807706. Extensive knowledge of printing and printing knowledge is known by me and shown in other production components and areas of the folio. Outsourcing of the printing of these items was NOT done because I didn't know how to print.

General Information Sources (excluding product research and in no particular order)

- [1] AutoCAD 2007 DXF Reference
Copyright © 2006 Autodesk, Inc.
- [2] Australian Engineering Drawing Handbook
The Institution of Engineers, Australia
ISBN 0 85825 057 8
1976
- [3] Engineering Drawing
Fourth Edition
A. W. Boundy
ISBN 0 07 452947 1
Copyright © 1992 by McGraw-Hill Book Company Australia Pty Limited
- [4] Complete Guide to Digital Printing
Rob SHEPPARD
Lark Books
ISBN 1-57990-427-0



- [5] TECHNICAL DRAWING for students
SAA HB1-1992
Standards Australia
ISBN 0 7262 7876 9

- [6] An Introduction to Programming Using Visual Basic 2005
Sixth Edition
Pearson Prentics Hall
David I. Schneider
ISBN 0-13-227922-3
© 2006 Pearson Education, Inc.

- [7] Fundamentals of Graphics Communication
Second Edition
WCB/McGraw-Hill
ISBN 0-07-289201-3
Copyright © 1998 by The McGraw-Hill Companies Inc.
Gary R. Bertoline

- [8] Industrial Technology Stage 6 Syllabus
Board of Studies NSW
May 2002 Published Version
ISBN: 0 7313 4228 3
© Board of Studies NSW 1999

- [9] 2006 HSC Notes from the Marking Centre – Industrial Technology
© 2007 Copyright Board of Studies NSW
ISBN 978 174147 6095

- [10] The Art of 3D Computer Animation and Effects
Third Edition
Isacc V. Kerlow
John Wiley & Sons, Inc.
ISBN 0-471-43036-6
Copyright © 2004 by Isacc Victor Kerlow

- [11] 4.500 Introduction to Design Computing, Spring 2006
Lecture Notes and Course ware (Accessed March 2007)
MIT OCW, <http://ocw.mit.edu/OcwWeb/index.htm>
© 2005 MIT

Acknowledgements:

Mr. M Rutledge

Mr. S Moses , Head Teacher TAS Industrial Arts, Blakehurst High School

