Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Data Visualization for ESM and ELINT

Visualizing 3D and Hyper Dimensional Data

John Altoft

Defence R&D Canada – Ottawa

Contract Report
DRDC Ottawa CR 2011-084
June 2011

Canada

Data Visualization for ESM and ELINT

Visualizing 3D and Hyper Dimensional Data

John Altoft

ALTEK Systems Inc.
Ottawa, Ontario

Prepared By:
John Altoft

Contractor's Document Number: ASI-2010-09-003

PWGSC Contract Number: W7714-050965
CSA: Barbara Ford

Defence R&D Canada – Ottawa

Contract Report
DRDC Ottawa CR 2011-084
June 2011

Original signed by Barbara Ford

Barbara Ford

Scientific Authority

Approved by

Original signed by J.F. Rivest

J.F. Rivest

H/REW

Approved for release by

Original signed by Chris McMillan

Chris McMillan

Chair DRP

Abstract

The last decade has witnessed a massive shift in 3D (3 Dimensional) device technology from high end virtual reality labs into the consumer market. Up until a decade ago, a virtual reality headset with the computer to drive it would likely have cost hundreds of thousands of dollars. Today a capable PC (personal computer) equipped with a 3D graphics processor unit (GPU), 120Hz refresh rate LCD (Liquid Crystal Display) monitor, and a pair of LCD shutter glasses, costs around $2000 dollars.

This project investigated 3D display technology and data visualization techniques for application to the ESM (Electronic Support Measures) and ELINT (electronic intelligence) domains. ESM and ELINT signal analysis are inherently complex and involve multi-dimensional analysis. The application of 3D graphic presentation to improve user comprehension of interim or final ESM signal analysis results was examined. This investigation examined both 3D representations presented in 2D and also true 3D visualization as is currently possible with modest Virtual Reality (VR) display systems.

Résumé

La dernière décennie a connu un passage généralisé à la technologie des dispositifs 3D (tridimensionnels), qui est passée des laboratoires de réalité virtuelle haut de gamme au marché de consommation. Jusqu’à il y a dix ans, un casque de réalité virtuelle et son ordinateur de commande pouvaient facilement coûter des centaines de milliers de dollars. Aujourd’hui, un bon ordinateur personnel (PC), doté d’un processeur graphique (GPU) 3D, d’un écran ACL (affichage à cristaux liquides) à taux de rafraîchissement de 120 Hz et d’une paire de lunettes à obturateur ACL, coûte environ 2 000 $.

Le présent projet vise à étudier la technologie d’affichage et les techniques de visualisation de données 3D aux fins d’application dans les secteurs des mesures de soutien électronique (MSE) et de renseignement électronique (ELINT). Une analyse des signaux MSE et ELINT est par définition complexe et comporte une analyse multidimensionnelle. On examine l’application d’une présentation graphique 3D pour améliorer la compréhension de l’utilisateur des résultats provisoires et finaux d’analyse des signaux MSE. La présente étude porte à la fois sur les représentations 3D présentées en 2D et la visualisation 3D réelle actuellement possibles au moyen de systèmes d’affichage de réalité virtuelle (RV) modestes.

This page intentionally left blank.

Executive summary

Data Visualization for ESM and ELINT: Visualizing 3D and Hyper Dimensional Data

John Altoft; DRDC Ottawa CR 2011-084; Defence R&D Canada – Ottawa; June 2011.

Introduction or background: The last decade has witnessed a massive shift in 3D (3 Dimensional) device technology from high end virtual reality labs into the consumer market. This move has been funded by the enormous market for high end consumer video games. Up until a decade ago, a virtual reality headset with the computer to drive it would likely have cost hundreds of thousands of dollars. Today a capable PC (personal computer) equipped with a 3D graphics processor unit (GPU), 120Hz refresh rate LCD (Liquid Crystal Display) monitor, and a pair of LCD shutter glasses, cost around $2000 dollars. The large production volumes of the consumer market, combined with the competition in the market place, will continue to drive the prices down and the 3D performance up.

The goal of this project was to investigate 3D display technology and data visualization techniques for application to the ESM (Electronic Support Measures) and ELINT (electronic intelligence) domains. ESM and ELINT signal analysis are inherently complex and involve multi-dimensional analysis. The complexity of the analysis often makes it difficult to represent an interim state of the analysis, or the solution, in the form of text or even 2-dimensional graphics. The application of 3D graphic presentation to improve user comprehension of interim or final ESM signal analysis results was examined. This investigation considered both 3D representations presented in 2D and also true 3D visualization as is currently possible with modest Virtual Reality (VR) display systems.

Results: Stereo pair stereograms for printed media were judged to be better quality and better suited to data visualization than anaglyph stereograms. For electronic documents and 2D computer displays "3D" jitter (also known as wiggle 3D) animations of 3D perspective projection images can greatly enhance depth perception. For 3D workstation displays, the technologies are ranked as follows: LCD shutter glasses were best, followed by stereo pair stereograms, and last was anaglyph stereograms.

Future plans: The Parallel Coordinate Browser prototype developed under this tasking to examine radar intra-coefficients, should be pursued as the key element of a coordinated toolset with linked 3D scatter plots and possibly other tools. The Java 3D class library should be explored to determine the feasibility of accessing LCD shutter glasses support for 3D displays directly from MATLAB code.

Sommaire

Data Visualization for ESM and ELINT: Visualizing 3D and Hyper Dimensional Data

John Altoft; DRDC Ottawa CR 2011-084; R & D pour la défense Canada – Ottawa; Juin 2011.

Introduction : La dernière décennie a connu un passage généralisé à la technologie des dispositifs 3D (tridimensionnels), qui est passée des laboratoires de réalité virtuelle haut de gamme au marché de consommation. Cette transition a été financée par un marché énorme pour les jeux vidéo haut de gamme destinés aux consommateurs. Jusqu'à il y a dix ans, un casque de réalité virtuelle et son ordinateur de commande pouvaient facilement coûter des centaines de milliers de dollars. Aujourd'hui, un bon ordinateur personnel (PC), doté d'un processeur graphique (GPU) 3D, d'un écran ACL (affichage à cristaux liquides) à taux de rafraîchissement de 120 Hz et d'une paire de lunettes à obturateur ACL, coûte environ 2 000 $. Les gros volumes de production du marché de consommation, combinés à la concurrence, continueront à faire baisser les prix et améliorer les performances de la technologie 3D.

Le présent projet vise à étudier la technologie d'affichage et les techniques de visualisation de données 3D aux fins d'application dans les secteurs des mesures de soutien électronique (MSE) et de renseignement électronique (ELINT). Une analyse des signaux MSE et ELINT est par définition complexe et comporte une analyse multidimensionnelle. La complexité de l'analyse rend souvent difficile la représentation d'un état intermédiaire de l'analyse, ou la solution, dans un format texte ou même graphiques bidimensionnels (2D). On examine l'application d'une présentation graphique 3D pour améliorer la compréhension de l'utilisateur des résultats provisoires et finaux d'analyse des signaux MSE. La présente étude porte à la fois sur les représentations 3D présentées en 2D et la visualisation 3D réelle actuellement possibles au moyen de systèmes d'affichage de réalité virtuelle (RV) modestes.

Résultats : On estime que les stéréogrammes à couple stéréoscopique pour des médias imprimés sont de meilleure qualité et mieux adaptés à la visualisation de données que les stéréogrammes anaglyphes. Pour les documents électroniques et les écrans d'ordinateur 2D, les animations à scintillement 3D (aussi appelé ondulation 3D) des images de projection en perspective 3D peuvent améliorer beaucoup la perception de la profondeur. Pour les écrans 3D des postes de travail, les technologies sont classées comme suit : les lunettes à obturateur ACL sont les meilleures, elles sont suivies par les stéréogrammes à couple stéréoscopique, puis par les stéréogrammes à anaglyphes.

Recherches futures : Le développement du prototype du navigateur à coordonnées parallèles dans le cadre du présent contrat en vue d'examiner des coefficients internes de radar devrait être continué en tant qu'élément clé d'un jeu d'outils coordonnés comportant des tracés de dispersion 3D liés et d'autres outils possibles. On devrait examiner la bibliothèque de classe Java 3D pour déterminer la possibilité d'accéder au support à lunettes à obturation ACL pour l'affichage 3D directement à partir du code de MATLAB.

Table of contents

Abstract …….. i
Résumé …..... i
Executive summary iii
Sommaire iv
Table of contents v
List of figures vii
List of tables ix
1 Introduction........ 1
2 3D Visual Perception 3
3 Static and Dynamic Media........ 7
4 3D Projection onto 2D 8
4.1 3D Projections for 2D Display 8
4.2 Adding Depth Perception to 3D Projections 8
4.2.1 Relative Sizes and Distance Fog 8
4.2.2 Motion Parallax........ 12
4.2.3 Occlusion of Background Objects 16
5 Stereoscopic 3D Presentation 20
5.1 Introduction 20
5.2 Anaglyphs........ 20
5.3 Stereo Pairs........ 22
5.4 Interlaced Stereo Displays........ 24
5.5 Software for Stereo Imaging 27
6 Hyper Dimentional Data........ 29
6.1 Introduction 29
6.2 Test Data........ 29
7 Visualizing Hyper Dimentional Data........ 33
7.1 Parallel Coordinate Graphs........ 33
7.2 Signal Analysis using Parallel Coordinates Graphs 35
7.3 Parallel Coordinates View for Navigating Hyper Dimensional Datasets........ 42
7.4 Glyph Plots for Viewing 4 to 12 Dimensions........ 46
7.5 Projection onto 2 or 3 Dimensions 48
8 Possible Applications........ 53
8.1 Problem Domains 53
8.2 Candidate Visualizations 54
8.2.1 Detection 54

8.2.2 Clustering 55
8.2.3 Correlation 56
8.2.4 Fusion 56
9 Summary 58
References 60
Annex A .. MATLAB Code Developed 61
A.1 Function: Camera 3D Pic 61
A.2 Function: Camera Jitter 3D 62
A.3 Function: Camera Jitter 3D Record 63
A.4 Function: Camera Record 3D Stripe 66
A.5 Function: Camera Distance Sort 68
A.6 Demo: Occlusion of Background Points 69
A.7 Function: Get Figure Handle 70
A.8 Function: Make Stereo Pair 70
A.9 Function: Movie2GIF 71
A.10 Function: Parallel Coordinates Browser 73
A.11 Demo: Simulate Distance Perspective for 3D 73
A.12 Function: Simple Tour 75
Annex B .. MATLAB File Exchange Programs 78
B.1 STEREOVIEW 78
B.2 ANAGLYPH 79
B.3 IMAGE2ANIMATION 80
List of acronyms 83

List of figures

Figure 1: Mono colour 3D projection of a data cluster 9
Figure 2: Size encoded depth information for 3D projection of a data cluster........ 10
Figure 3: Colour encoded depth information for 3D projection of a data cluster 11
Figure 4: Colour & Size encoded depth information for 3D projection........ 12
Figure 5: First frame of a Jitter 3D image sequence for Radar data........ 14
Figure 6: Second frame of a Jitter 3D image sequence for Radar data 14
Figure 7: Mono colour data cloud 15
Figure 8: Data cloud structure enhanced by colour........ 15
Figure 9: Unordered 3D projection display of two data classes........ 17
Figure 10: Depth Ordered 3D display of two data classes 18
Figure 11: Rotating the view point shows the two data classes are separate........ 18
Figure 12: Anaglyph image created by ANAGLYPH by I.G. Marino........ 21
Figure 13: Stereoview a MATLAB file exchange program by I.G. Marino 23
Figure 14: PokeScope stereogram viewer for side-by-side Stereo Pairs........ 24
Figure 15: nVidia 3D Shutter Glasses with Infrared Controller for Synchronization........ 26
Figure 16: WEST19B recorded radar intercepts 30
Figure 17: nDim 5,ooo simulated radar pulses........ 31
Figure 18: nDim simulated radar intercept parameters 32
Figure 19: Parallel Coordinates plot of a single point........ 33
Figure 20: MATLAB Parallel Coordinates Graph Representation 34
Figure 21: Interpreting a Parallel Coordinates Graph........ 35
Figure 22: The 24 IP coefficient dimensions for 24 pulses from Emitter 85 36
Figure 23: The 24 IP coefficient dimensions for 48 pulses from Emitter 22 36
Figure 24: “Standardized” display of Emitter 85 emphasizes noise........ 37
Figure 25: The 24 IP coefficient dimensions for three simulated radar classes 38
Figure 26: First prototype Parallel Coordinates Browser........ 39
Figure 27: Selecting data with band pass filters........ 40
Figure 28: Suppressing data outside of the filter band 40
Figure 29: Selected data after two filters applied........ 41
Figure 30: Selected data compared to ground truth ID information........ 41

Figure 31: Scatter plot of coordinates 9, 11, and 13 43
Figure 32: Scatter plot of coordinates 11, 13, and 19 44
Figure 33: Scatter plot of coordinates 3, 23, and 24 45
Figure 34: Scatter plot of coordinates 19, 21, and 23 46
Figure 35: Average coordinate values for the three emitter types 47
Figure 36: Individual samples from each emitter type 47
Figure 37: A 5D Matrix of Scatter Plots for three Emitters 49
Figure 38: Automated versus Manual Data Exploration 50
Figure 39: MATLAB waterfall display 55

List of tables

Table 1: Standard Terminology for Stereoscopic Imaging... 4

This page intentionally left blank.

1 Introduction

The last decade has witnessed a massive shift in 3D (3 Dimensional) device technology from high end virtual reality labs into the consumer market. This move has been funded by the enormous market for high end consumer video games. Up until a decade ago, a virtual reality headset with the computer to drive it would likely have cost hundreds of thousands of dollars. Today a capable PC (personal computer) equipped with a 3D graphics processor unit (GPU), 120Hz refresh rate LCD (Liquid Crystal Display) monitor, and a pair of LCD shutter glasses, cost around $2000 dollars. The large production volumes of the consumer market, combined with the competition in the market place, will continue to drive the prices down and the 3D performance up.

The cost for immersive 3D technology, such as the CAVE[1] systems, remains moderately high due to the complexity of coordinating multiple large screen displays and technical difficulty in obtaining a perceived 3D image from a projection screen. The CAVE systems are targeted at problem domains where multiple operators collaborate on a single problem. The physical size of the CAVE allows multiple operators to be present and to interact with the display and each other. Some 3D headset displays can also provide the feeling of being immersed in the image, and multiple operators can participate in the same 3D virtual reality, however, the operator to operator interaction is not well developed at this time, partly due to the limitations of operator input devices. The future improvement of operator avatars and their ability to interact in a manner analogous to real world perceptions and interactions will likely limit the acceptance of CAVE technology in the long run. The near term improvement of inexpensive operator avatars and input devices is nearly guaranteed by the huge commercial market in first person multi-player games. The advantages of headset displays over CAVE systems are both cost and the fact that the participating operators need not be physically present.

The COTS market for 3D headsets has two dominant designs at this time, the LCD shutter glasses that operate with a display screen, or the stand alone headsets with LCD displays integrated into the headset. The shutter glasses have the advantage of cheaper cost and they support multiple viewers of a common display. In addition, the image resolution is limited only by the external display. The integrated LCD headsets are generally more expensive and the consumer models have limited resolution at this time (typically 640x480 or 800x600). However, the integrated LCD headsets have a distinct advantage over the shutter glasses for immersive virtual reality since the operator can scan the image scene by way of natural head movements. Several integrated LCD headsets support head trackers with 3 or 6 degrees of freedom (DOF). The 3 DOF trackers use roll, pitch, and yaw sensors and are normally attached to the glasses. The 6 DOF head tracker adds X, Y, Z head location and is generally a fixed external tracking system. A single 6 DOF head tracker will likely have limited range of motion for yaw and pitch since they normally track reference points on the operator's head which can become obscured if the head is turned too far away from the tracking sensor. High end virtual reality labs solve this problem by using multiple distributed trackers to cover the full range of motion.

[1] CAVE is a name given to 3D display systems where the operator walks "into" the virtue space. Typically these are coordinated displays on 3 walls, ceiling and perhaps the floor. The intent is for the image to fill the entire field of view of the operator to create the perception that the operator is immersed in the scene. CAVES are used in place of 3D headsets for a group of operators to collaborate on a problem.

The low cost integrated headset displays are also spawning interest in related areas. In 2010-2011 timeframe, several manufactures will market augmented reality headsets that are basically a consumer version of a heads-up display (HUD). At least two different approaches are proposed. The first uses a transparent LCD display in the headset to allow the operator to look through the display image thus creating a display overlay on the actual external scene and providing a true HUD experience. The other approach uses a standard 3D LCD headset with headset mounted stereo video cameras added. The operator can choose between external view, virtual view, or any mix of the two. One advantage to this technology is that other participants can exactly follow the operator viewpoint. This could have significant impact in areas like medical telepresence for surgery. Not only could the remote specialist see what the doctor sees, the remote specialist could provide visual direction to the doctor via the HUD overlay.

The goal of this project was to investigate 3D display technology and data visualization techniques for application to the ESM (Electronic Support Measures) and ELINT (electronic intelligence) domains. ESM and ELINT signal analysis are inherently complex and involve multi-dimensional analysis. The complexity of the analysis often makes it difficult to represent an interim state of the analysis, or the solution, in the form of text or even 2-dimensional graphics. The application of 3D graphic presentation to improve user comprehension of interim or final ESM signal analysis results will be presented herein. This investigation considers both 3D representations presented in 2D and also true 3D visualization as is currently possible with modest Virtual Reality (VR) display systems.

2 3D Visual Perception

This section will provide a general overview of 3D visual perception starting with a definition of standard terminology used in the field of stereoscopic imaging.[2]

Accommodation	The refocusing of the eyes as their vision shifts from one distance plane to another.
Accommodation-Convergence Link	The physiological link that causes the eyes to change focus as they change convergence, a link that has to be overcome in stereo viewing since the focus remains unchanged on the plane of the constituent flat images.
Convergence	The meeting of lines of sight through the eyes, or light rays through the optical system, at a common point closer than infinity.
Far Point	The feature in a stereo image that appears to be farthest from the viewer.
Stereo Acuity	The ability to distinguish different planes of depth, measured by the smallest angular differences of parallax that can be resolved binocularly.
Stereogram	A general term for any arrangement of left-hand (LH) and right-hand (RH) views that produces a three-dimensional result, which may consist of the following: (1) a side-by-side or over-and-under pair of images, (2) superimposed images projected onto a screen, (3) a colour-coded composite (anaglyph), (4) lenticular images, (5) a vectograph or (6) in film or video, alternate projected LH and RH images that fuse by means of the persistence of vision.
Steriopsis	The process in visual perception leading to the sensation of depth from the two slightly different projections of the world onto the retinas of the two eyes.

[2] http://www.stereoscopy.com/isu/glossary-a.html {2010/10/30}

Stereoscopic	'Solid looking': having visible depth as well as height and width. May refer to any experience or device that is associated with binocular depth perception.
Stereoscopy	The reproduction of the effects of binocular vision by photographic or other graphic means.
Near Point	The feature in a stereo image that appears to be nearest to the viewer.
Parallax	Generally, the differences in a scene when viewed from different points
Planar (image)	Flat, or in a single plane; as opposed to stereoscopic or three-dimensional.

Table 1: Standard Terminology for Stereoscopic Imaging

The human eye only records 2D images of the world; our ability to extract depth perception from these images comes from sensory processing in the brain. The primary cue used to create depth perception is our binocular vision. The parallax caused by the physical separation of our eyes presents our brain with left and right images which differ slightly along the inter-ocular axis. The differences in the two retinal images are called horizontal disparity, retinal disparity, or binocular disparity. The convergence point is where the left and right lines-of-sight cross; it is normally located closer than infinity and could, in some cases, be very close to the observer. The binocular disparity is largest for objects furthest from the convergence point, either in the foreground or in the background. The brain is able to interpret this binocular disparity as depth information.

Binocular vision is only one of the depth cues available to humans; others include motion parallax, perspective, occlusion, kinetic depth perception, familiar and relative sizes, distance fog, texture gradient, accommodation, etc. Humans who only have vision in one eye cannot experience depth perception from a stereogram; however, they can be fully capable of depth perception when viewing a real scene due to the other depth cues available.

Motion parallax is the apparent relative motion of objects against the background when the observer moves. Nearby objects will have larger relative displacements against the background features than more distant objects, while objects at large distances will exhibit no perceivable relative motion. Some animals, such as pigeons, do not have binocular vision and their constant head bobbing is their method of creating motion parallax to obtain depth information.

Perspective is the visual perception that causes parallel lines to appear to converge at the horizon (infinity). As objects become more distant they appear smaller because their observed visual angle decreases. Although two parallel lines (e.g., railway tracks) maintain a constant separation distance, the angular separation decreases as the distance from the observe increases, which causes them to appear to converge. Perspective is also related to the depth cues based on familiar and relative sizes. If the observer is familiar with the size of an object (e.g., the average height of a human) then the observer will interpret unusually small visual angles for that object as indications that it is far away. Similarly, two entities that are perceived as the same object type will be interpreted as separated in distance (depth) if they have dissimilar visual angles.

Occlusion is the blocking of the line of sight to an object by another object in the foreground. Occlusion of objects only provides relative ranking of the objects with respect to depth.

Kinetic depth perception is related to perspective in that it involves changes in the visual angle of an object as it approaches or recedes from the observer. The apparent increase in an object’s size is interpreted as movement toward the observer, and vice versa. Kinetic depth perception is what allows us to estimate time to collision/contact with an approaching object.

Distance fog describes the visual differences between near and distant objects due to the effects of light scattering by the atmosphere. Distant objects have lower luminance contrast and lower color saturation. In addition, the colour spectrum of the object may be shifted toward the blue end of the spectrum.

Texture gradient is the loss of fine texture detail as the distance from the observer increases. This is another example of the effect of the visual angle decreasing with distance from the observer. As the visual angle becomes too small to resolve, the texture is lost. Standing on a gravel road we see individual stones on the road surface at our feet but we see only a mottled grey road surface if we look 500m down the road.

Accommodation is the refocusing of the eyes as our vision shifts from one depth plane to another. If the eyes need to refocus when shifting our attention from one object to another, then the brain will interpret that change in accommodation as a difference in distance (depth).

Some of the above cues for depth perception are important to data visualization techniques and others are not. In particular, accommodation cannot be used with 2D media (including stereograms) to convey depth information since the entire scene exists in a single focal plane. In fact, the absence of the need for the eyes to accommodate will work against the visual perception of depth from a stereogram until the brain becomes accustomed to separating the parallax information from the accommodation. A small percentage of people never succeed in perceiving the 3D depth of a scene from a stereogram.

The depth perception cues of primary interest for data visualization are the following:

- Parallax
- Motion Parallax
- Occlusion
- Relative Sizes
- Perspective.

A static image stereogram can contain all of the above depth cues except motion parallax (due to the image being static). An animated image from a single viewpoint can contain all of the above depth cues except parallax (due to the lack of binocular viewpoints). An animated stereogram can contain all five of the depth cues listed above.

A widely used stereogram technology is anaglyph imaging, which the International Stereoscopic Union defines as follows:

> A type of stereogram (either printed, projected or viewed on a TV or computer screen) in which the two images are superimposed but are separated, so each eye sees only the desired image, by the use of coloured filters and viewing spectacles (commonly red and cyan, or red and green).[3]

Anaglyph images can be made from black and white, or colour originals, however, with colour originals some colour details must be suppressed so as to not interfere with the depth encoding. In an anaglyph stereogram the image intended for one eye could be encoded in red (usually the left eye) while the image for the other eye would be encoded as cyan or green. With black and white images this is a simple colouring process where the white areas take on the designated colour. For colour images the process is more complicated. All portions of the scene located at the same depth distance as the convergence point will have zero binocular disparity, meaning that the left and right images have no horizontal shift for those image elements. Since there is no horizontal shift to encode, these elements need not be colour encoded and can appear in natural colour. Picture elements much closer or much further away from the convergence point will require the largest horizontal shifts and, as a consequence, will appear more monochromatic so as to maximize the effect of the colour encoding. If the high disparity elements of the red eye image were allowed to retain green/cyan hues those elements would appear in the other eye as ghost images around the fringe of the object.

Another popular stereogram technique, stereo pairs, was very popular with the public decades ago (View-Master[4]) and is enjoying a technological revival. Stereo pairs can be monochrome or full colour and they can produce very life-like 3D images. Stereo pairs can be presented side-by-side, above-and-below, or interlaced at the same location on a display screen. The disadvantage of stereo pairs is that they generally require more elaborate external viewing equipment. Whereas, an anaglyph stereogram requires only a simple pair of filter glasses, a side-by-side or an above-and-below stereo pair stereogram generally requires a stereoscope that employs prisms, lenses, or mirrors to facilitate the visual fusion of images. An interlaced stereo pair can be viewed with LCD shutter glasses that are synchronized to the interlacing of the image. On the other hand, the significant advantage of a stereo pair over an anaglyph stereogram is the colour purity and quality of the 3D image.

This project will also examine non-stereogram imaging techniques that can still provide 3D information using the other depth cues apart from binocular parallax. In particular, motion parallax will be examined in more detail due to its importance to depth perception and its compatibility with dynamic display media such as computer screens.

[3] http://www.stereoscopy.com/isu/glossary-a.html {2010/10/30}
[4] http://en.wikipedia.org/wiki/View-Master {2010/10/30}

3 Static and Dynamic Media

The presentation or display media will have a direct impact on what techniques and technologies can be applied to present 3D information. The primary media characteristic is static versus dynamic presentation. Printed reports are static media, whereas electronic documents presented on display screens can be dynamic presentations. Dynamic presentations can be user interactive but it is not a requirement; a canned animation qualifies as a dynamic presentation.

Clearly dynamic media offers more opportunity to exploit a richer set of depth cues to enhance the 3D experience for the user. User interactive 3D display technology provides the highest degree of dynamic media behaviour, with immersive virtual reality being the current pinnacle of technology.

Static paper reports will continue in common use for the foreseeable future therefore this project will include 3D techniques that are suitable for static media; however the primary emphasis will be on dynamic presentation media. As more information systems move toward electronic documents (e.g., PDF documents) the opportunity to incorporate dynamic presentations will increase and become more commonplace.

Static or dynamic media used to present stereograms will normally require additional external equipment to view the 3D image. The type of equipment will depend on the type of stereogram and the encoding used. The lack of a universal standard for stereograms may cause some problems for early adopters of the technology. Stereograms based on stereo pairs can be either side-by-side pairs or above-and-below pairs. Anaglyph stereograms typically use either red-green or red-cyan filters for the images and the red filter is commonly but not universally used for the left eye. Interlaced stereograms can use either cross polarized images (vertical-horizontal or circular) or they may use shutter glasses synchronized to the image interlacing.

The lack of widely accepted standards represents an element of risk for early adopters of the technology. The risk is most significant with long life documents, such as reports. The risk is less significant with analysis workstations since the computer technology that the workstation is based on has a short lifespan which allows for adoption of new standards. For the analysis workstation the 3D technology is an analysis tool and is not necessarily part of the analysis product; therefore, non-standardized technology can still be useful and effective. Even if the analysis product does contain 3D content it would be a simple conversion process to export the analysis product in whichever 3D format is desired, irrespective of the 3D technology used in the analysis.

4 3D Projection onto 2D

4.1 3D Projections for 2D Display

A 3D data visualization such as a 3D scatter plot can be presented on 2D media as a 3D projection onto the 2D plane. The projection of a 3D scatter plot often uses parallel projection where lines that are parallel in the 3D space are also parallel in the projection. The less common alternative is a perspective view where parallel lines converge at some point on the image horizon. In either case the depth information is included and the user can logically infer depth relationships between elements of the image, however there is no natural visual perception of depth.

A 2D projection of a 3D space can be an effective way of presenting information on 2D media that cannot support true 3D imaging. Interactive 2D media, such as computer display screens, can greatly enhance user comprehension by providing user control of the viewing position. Allowing the user to zoom in or out, or to move around the structure to view it from all angles, can significantly improve the user's perception of the 3D structure. Smooth flowing animated movement is much preferred to jumping between viewpoints as it allows the user to maintain orientation and context, as well as providing an opportunity for true depth perception by motion parallax and occlusion cues.

2D projection of a 3D image does however present some problems for operator interaction with the data. Any pointing cursor or selection tool used by the operator will be restricted to its 2D screen coordinates which map directly to the 2D data projection in a one-to-one fashion. The ambiguity arises when the 2D projected data point is converted back into the 3D space. Each point on the 2D projection is ambiguous with a line in the 3D space that is oriented normal to the plane of the display screen. Supporting operator selection of 3D points or volumes from 2D projections will require customized controls that operate in 3D and are mapped into the 2D projection for visual feedback to the operator.

4.2 Adding Depth Perception to 3D Projections

4.2.1 Relative Sizes and Distance Fog

Human visual perception and depth perception, in particular, can be strongly influenced by our interpretation of relative sizes of objects. Relative size cues for depth perspective are so strong that it is routinely used in the film industry where small models can stand in for large distant structures by careful use of camera perspective. This section will examine the use of relative size and distance fog to encode depth information. Recall that distance fog is the loss of colour saturation caused by atmospheric effects over long distances.

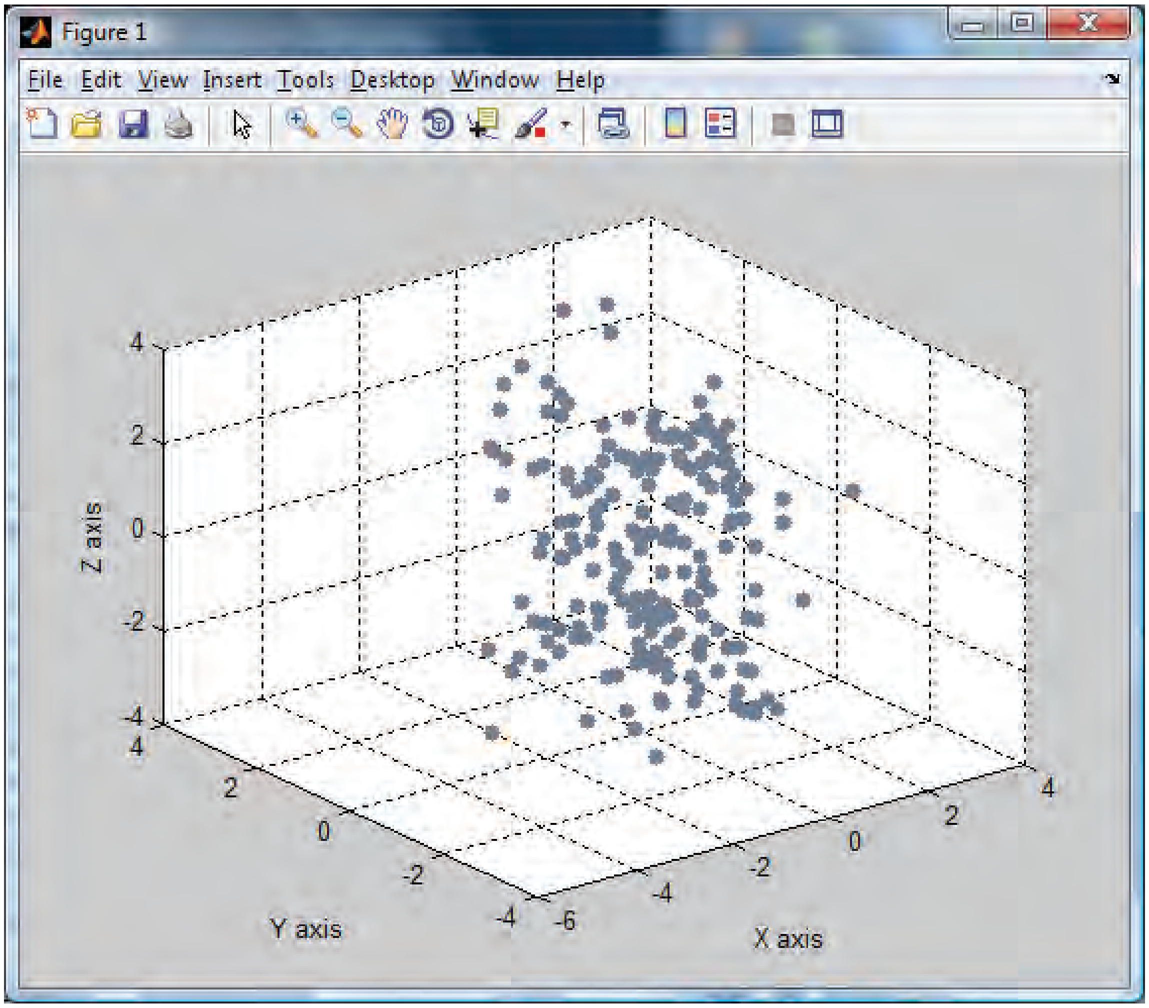


Figure 1: Mono colour 3D projection of a data cluster

The first image, in Figure 1, is a standard scatter plot of 3D data using MATLAB's *scatter3* plotting function. This plot has very few depth or other spatial cues to assist our interpretation of the data. This represents the typical starting point for an analyst, where little is known about the structure or content of the data.

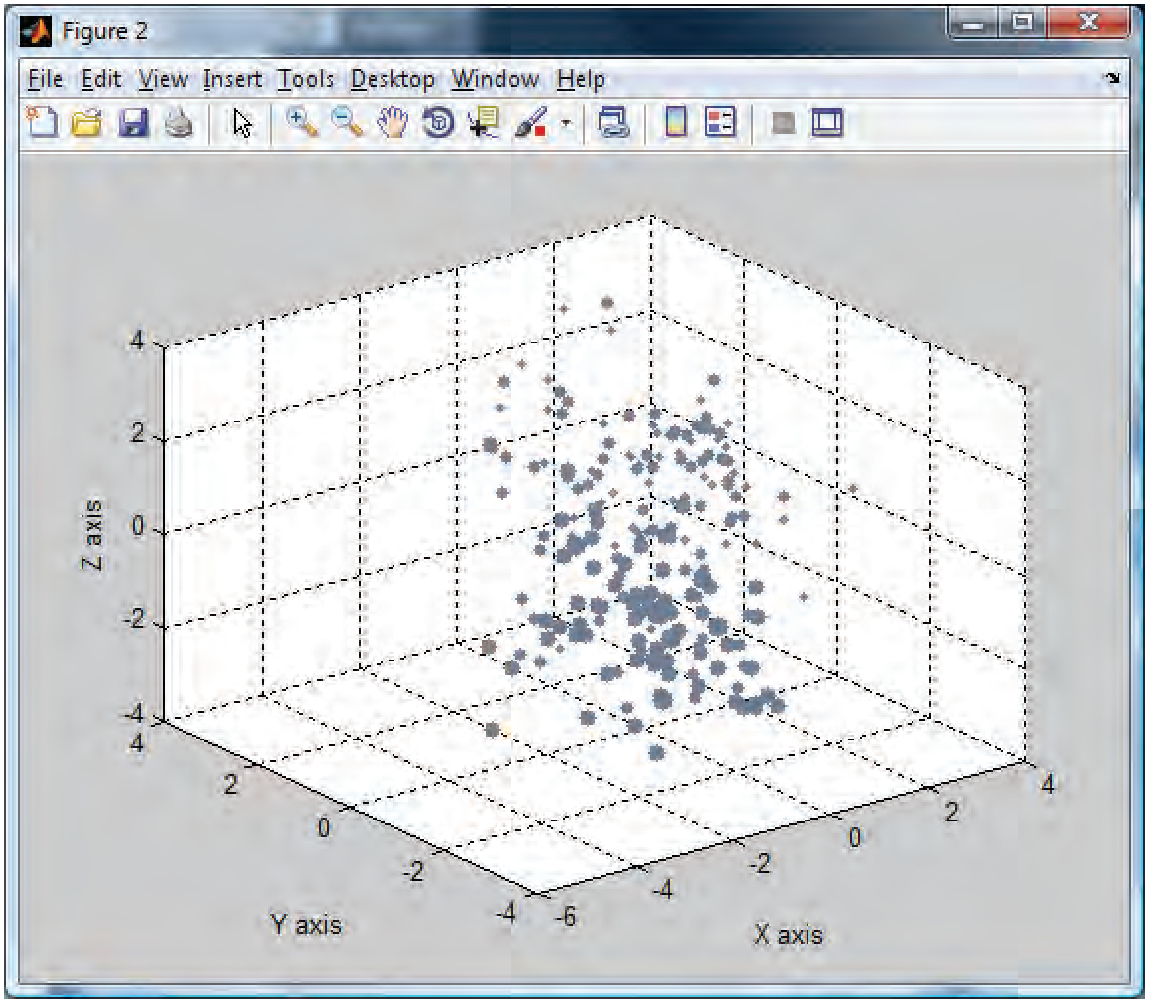


Figure 2: Size encoded depth information for 3D projection of a data cluster

The second image, in Figure 2, has encoded the depth information as variations in the relative sizes of the data points. This process involves computing the distances between each point and the observation position (camera position in MATLAB) and scaling these distances into a discrete range of point sizes. In this case the point sizes range from 60 for the nearest points to 5 for the points furthest from the observer. The use of relative sizes as a function of depth does not create a true sense of depth, however it does significantly enhance our ability to interpret the image. It is now clear that the data is scattered through a volume in the space.

During data analysis our focus is on the data and the processes or transformations that we use to manipulate the data into a more usable form, with the result that the data presentations we choose often represent that data centric focus. In this case the display is making the relationship between the observer and the data explicit in the display. The data point sizes are not a data attribute per se; they represent an observer-to-data relationship.

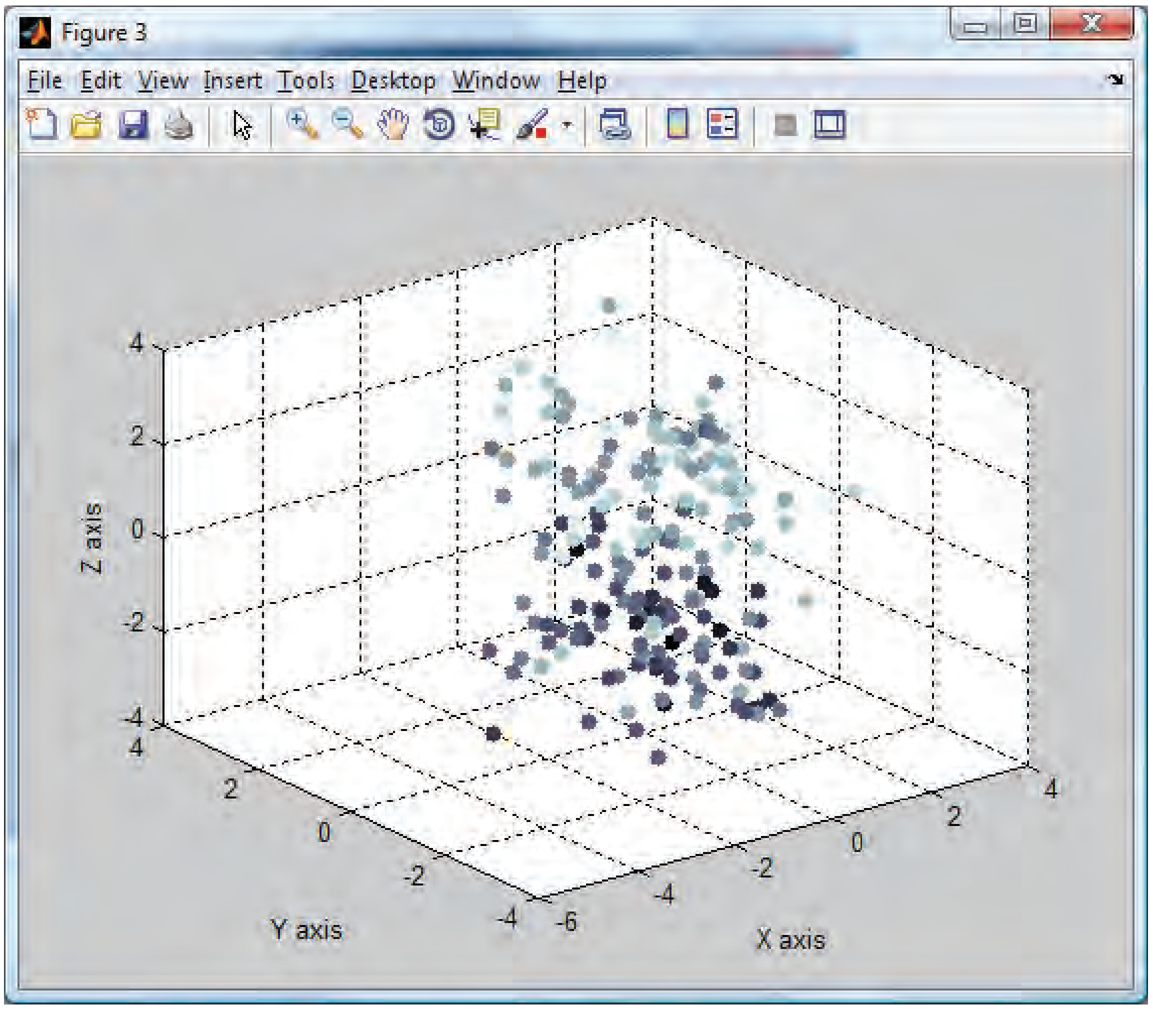


Figure 3: Colour encoded depth information for 3D projection of a data cluster

The third image, in Figure 3, uses only colour, or in this case shading, to encode depth cues. The lighter the colour the more distant the point is from the observation position, which mimics real life with the loss of colour saturation with distance. The use of colour alone seems ineffective for natural depth cues although it can be helpful for reasoning about depth position while examining the image. It is possible that relative size cues are more important than loss of colour saturation and, therefore, we interpret this image as similar objects at a similar relative distance because of their equal size. Also note that if colour is used for depth encoding, the user must take control of the plotting function. In this case, the MATLAB *scatter3* plot routine does not know the significance of the colour encoding and some light background points were plotted on top of dark foreground points thus violating the natural depth occlusion that our brains expect to see. Preliminary testing suggests that this can be corrected by pre-sorting the data into depth order before plotting. This will be addressed in more detail in section 4.2.3.

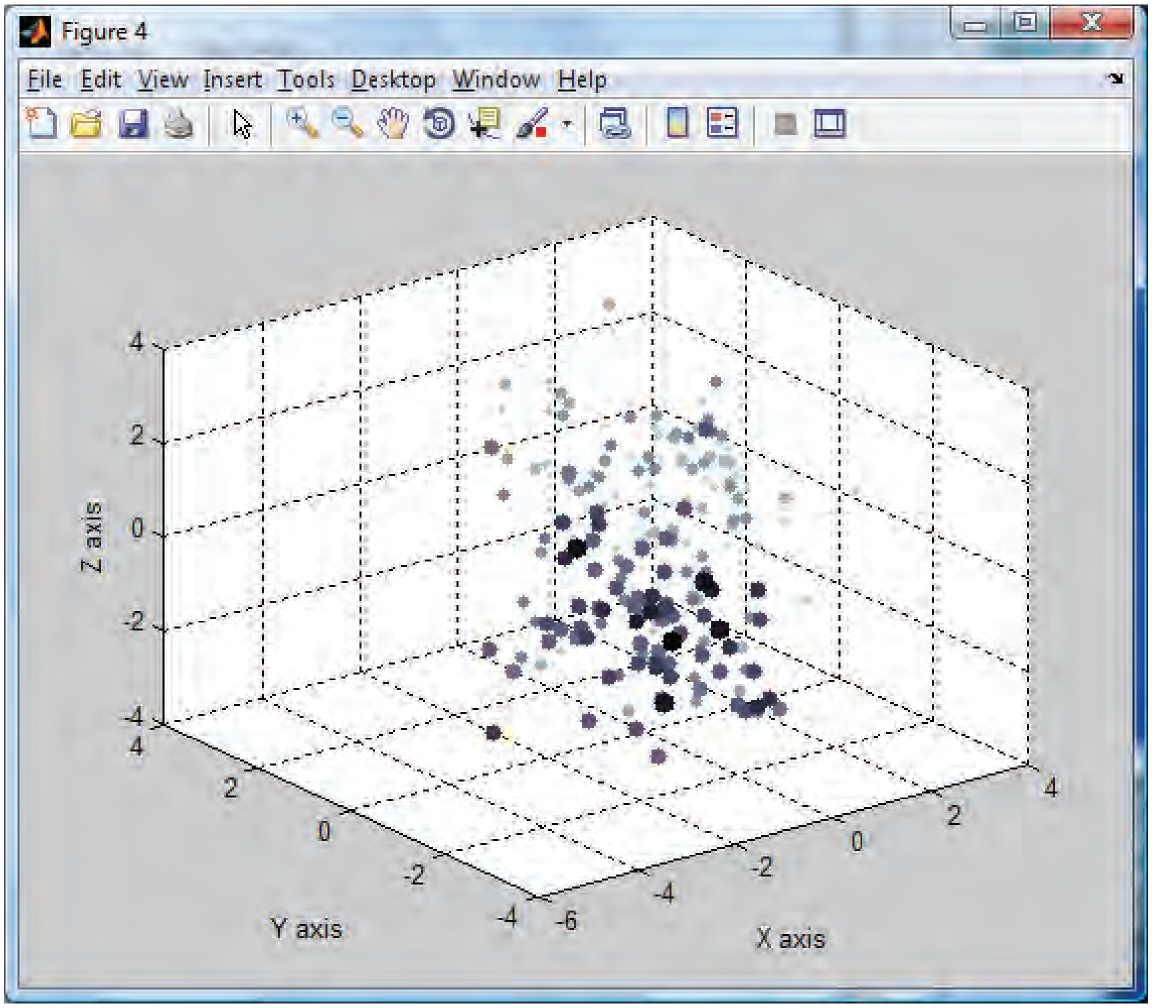


Figure 4: Colour & Size encoded depth information for 3D projection

The last image, in Figure 4, illustrates the use of both relative size and distance fog to encode the depth information. In this case the foreground points are both larger and darker. Although these depth cues are insufficient to create a natural perception of depth, they do lend themselves to a more rapid interpretation of the distribution of the data in the three dimensions. In the first figure of this set, which has no depth encoding, the data could actually be distributed over a plane that is being viewed from above; however, the depth encoding in the other three images clearly shows that the data is distributed over a volume.

4.2.2 Motion Parallax

Motion parallax is a very strong depth cue which can provide an almost real perception of depth. Unfortunately, motion parallax can only be used with dynamic presentation media so it is not possible to capture it in this paper document. The technique that is examined herein is sometimes referred to as "Jitter 3D" and often makes use of the animation feature available in the GIF image specification. Jitter 3D is a technique wherein the perspective of the user is shifted left and right to produce motion parallax effects in the scene. The jitter motion can contain several observation positions or as few as two. With only two frames the effect is similar to blinking alternate eyes and while producing the desired motion parallax; it can be difficult to view if foreground or background objects jump through large angles. Adding extra frames to produce smooth movement can greatly enhance the viewing experience and produce more natural depth perception. A significant advantage of the technique is that depth perception can be experienced without any special software, hardware, or viewing equipment.

Jitter 3D is not limited to producing 3D images in animated GIF files; it can be applied to graphic displays for analysis workstations. When 3D projection graphics are used to display 3D or higher dimensional data, jitter can be applied to shift the observer position over a small arc to produce motion parallax effects in the graphic. The induced depth perception can be quite dramatic if the jitter is smooth and natural. The limitation to this technique is the time required to update the graphic display with the next perspective. If the dataset is very large, the redraw time may be too long to permit smooth motion. In this case one option is to capture the frames to a short animation and then play back the jitter at a more natural speed. Once a frame-based animation sequence is captured (or if the graphic refresh is fast enough), other options become available to the analyst. A control called a jog/shuttle wheel is often used in video editing to rapidly move forward, backward, or jitter in one location in the film sequence. This tool could be useful to an analyst by allowing them interactive control over the motion viewing. The control combines the ability to move to a new perspective with the ability to jitter the perspective to induce motion parallax while generally maintaining the perspective position. The analyst could orbit or pan across the data and when any aspect catches their attention they could linger and maintain the motion to induce depth perception by jittering.

Figures 5 and 6 show two frames from a Jitter 3D sequence that was used to display radar ESM data. This subset of data is from the simulated IP (intra-pulse) data set, and displays bearing and RF versus Time. The true emitter ID tags were used to colour the data points. (Note that some colours may have been used for more than one ID). The motion parallax is generated by orbiting the MATLAB camera position about a line parallel to the Z axis and passing through the target point for the camera, which was approximately the center of the data field. Multiple frames were generated for onscreen display and these two frames were selected for the report. These frames were not adjacent frames in the jitter sequence since adjacent frames normally have small angular differences for smooth motion, however, those small differences would be hard to see in this static presentation.

The radar ESM example has reasonably distinct clusters of well separated data that would likely have been recognizable even without the colour coding by emitter ID. To demonstrate the effectiveness of motion parallax to detect structure in data clouds another example is included in the report. Figures 7 and 8 show two views of the same data cloud, the first is mono colour to represent data with unknown structure, while the second has been colour encoded to illustrate the internal structure for the reader. The animation included with this report starts with the non encoded data display in a static view. This represents the typical view an analyst would have. The display is then jittered to induce motion parallax and depth perception.

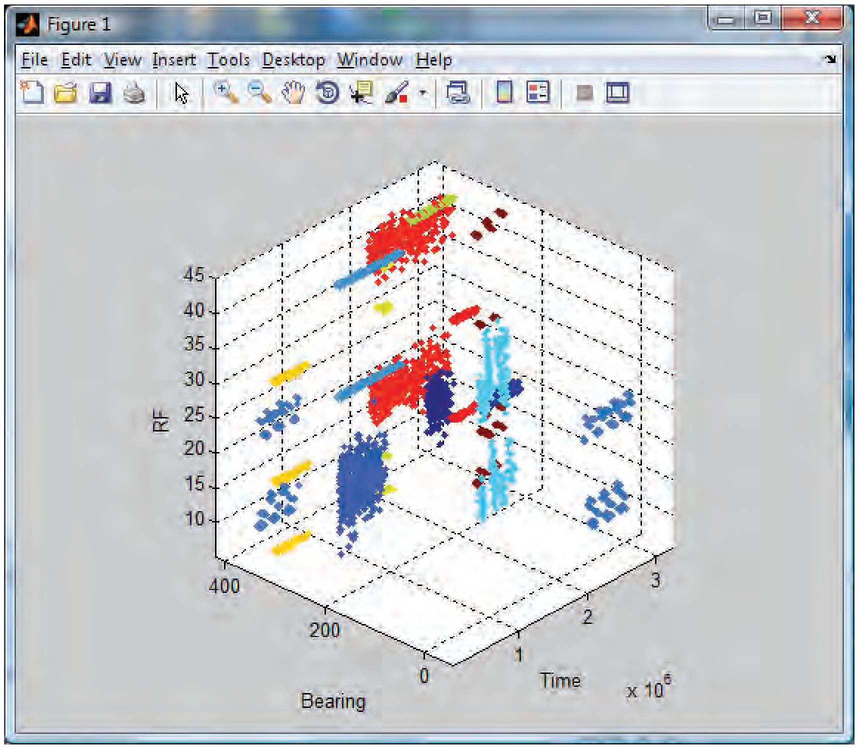


Figure 5: First frame of a Jitter 3D image sequence for Radar data

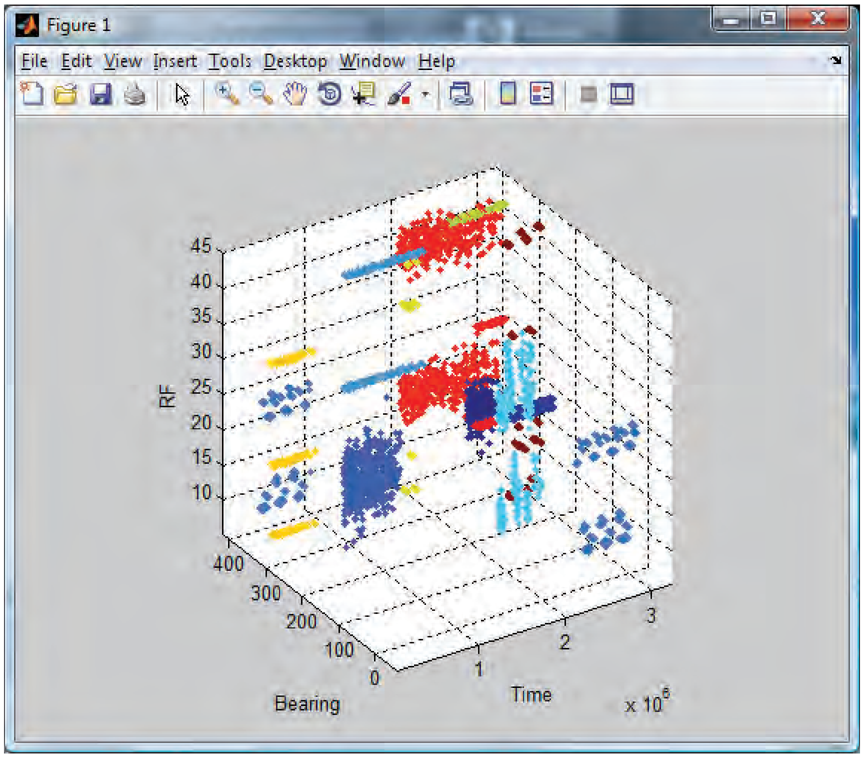


Figure 6: Second frame of a Jitter 3D image sequence for Radar data

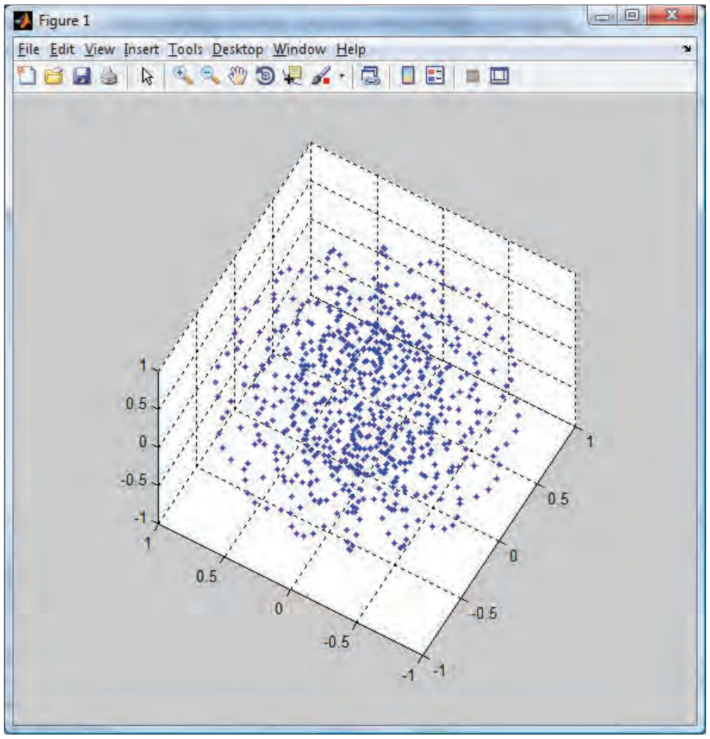


Figure 7: Mono colour data cloud

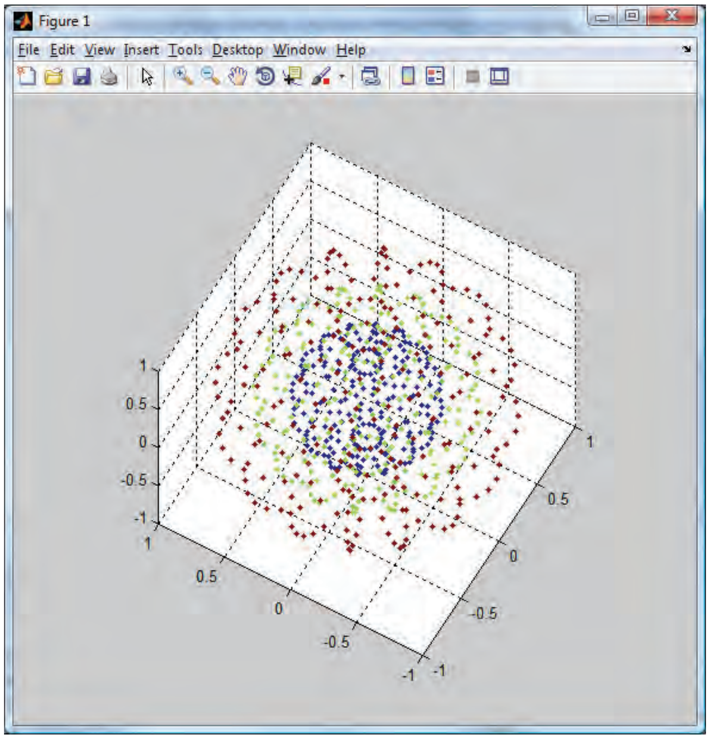


Figure 8: Data cloud structure enhanced by colour

With motion parallax the internal structure becomes evident even without the knowledge required to colour the structure. Adjacent structural features move together making them immediately apparent. This kind of visual perception is what the human brain has evolved to perform: the detection of objects (structures that move together) in a cluttered background. Human survival depends on it and we are exceedingly good at it when compared to machine visual recognition.

A MATLAB jitter function was written and included with this report; the header section of that function is displayed below and the complete function is listed in the Appendix A.2. This function can be called with a handle to a graph axis or without, in which case it uses *gca* (the current axis handle). The function will cause the camera location to perform a small back and forth orbit about the current camera target which will cause the motion parallax effect. This function is only effective for displays using 3D projection, such as *scatter3*, *plot3*, *surf*, etc.

```
function varargout = camera_jitter_3d(varargin)
        % Jitter camera position about Z-axis to create motion parallax for 3D
        %
        % Inputs: nothing or an axes handle
        %       camera_jitter_3d() - jitter the current (gca) axes view
        %
        %       camera_jitter_3d(axes-handle) - jitter the provided axes view
        %
        % Outputs (optional):  the axes handle
```

4.2.3 Occlusion of Background Objects

All 2D displays can contain occlusion of data points by other data points. When the data is 3D or higher dimensionality the occlusion can be exploited as an additional display dimension that shows relative ranking of distance from the observation point.

Occlusion of background points by foreground points is a very strong depth ranking cue, which cannot be exploited if all points are mono colour. Without colour separation the overlapping points merge into a uniform blob. If colour is not already being used to encode some attribute of the data then it is recommended that it be used to encode depth information with respect to the observation point for the graph image.

Whenever multiple colours are used with three or higher dimensional data, the display order should be controlled to display the correct occlusion of the points, even if the colour is not related to encoding depth information. Occlusion for depth ranking effectively adds an additional dimension to the display if correctly presented. As noted in the previous section this may involve pre-sorting the data into depth order with respect to the observation position before plotting the data. The following figures illustrate the importance of correct depth occlusion. This data is colour coded to represent two distinct classes of data. In this case the colour is not encoding the distance between the data point and the observation point.

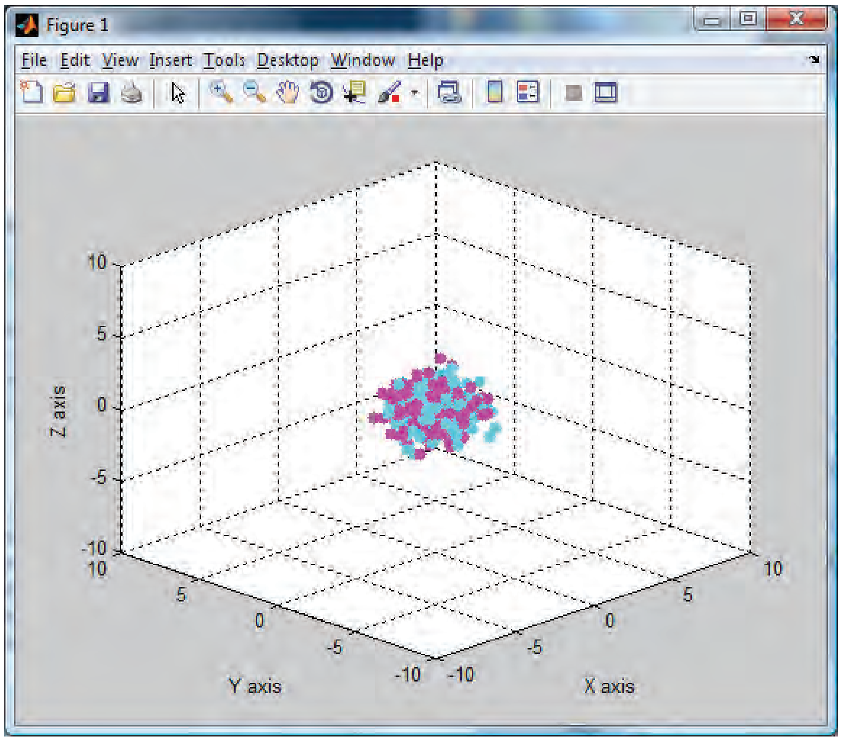


Figure 9: Unordered 3D projection display of two data classes

In the first figure, Figure 9, the data from the two classes appears to be mixed in a common cloud. This display was created by MATLAB *scatter3* used in a typical fashion. The next display, Figure 10, uses the same MATLAB display functions however the data was pre-sorted to order it by distance from the observer position (camera position in MATLAB). Only the order of the data points in the list where changed. In this case one cluster seems to dominate the display. In the third image, Figure 11, the camera view point is rotated to reveal the two distinct clusters, each containing 200 points.

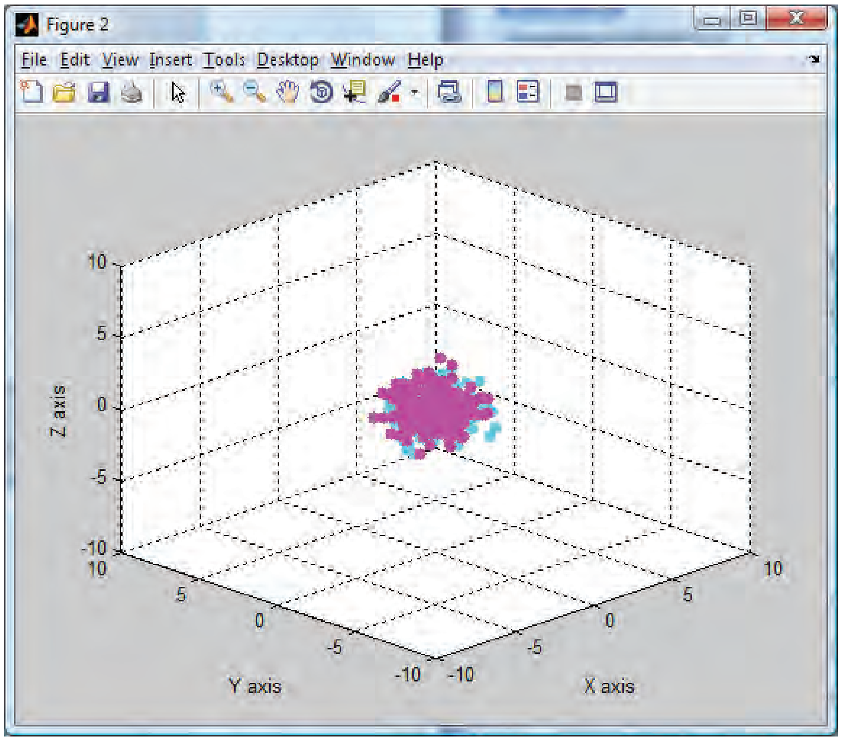


Figure 10: Depth Ordered 3D display of two data classes

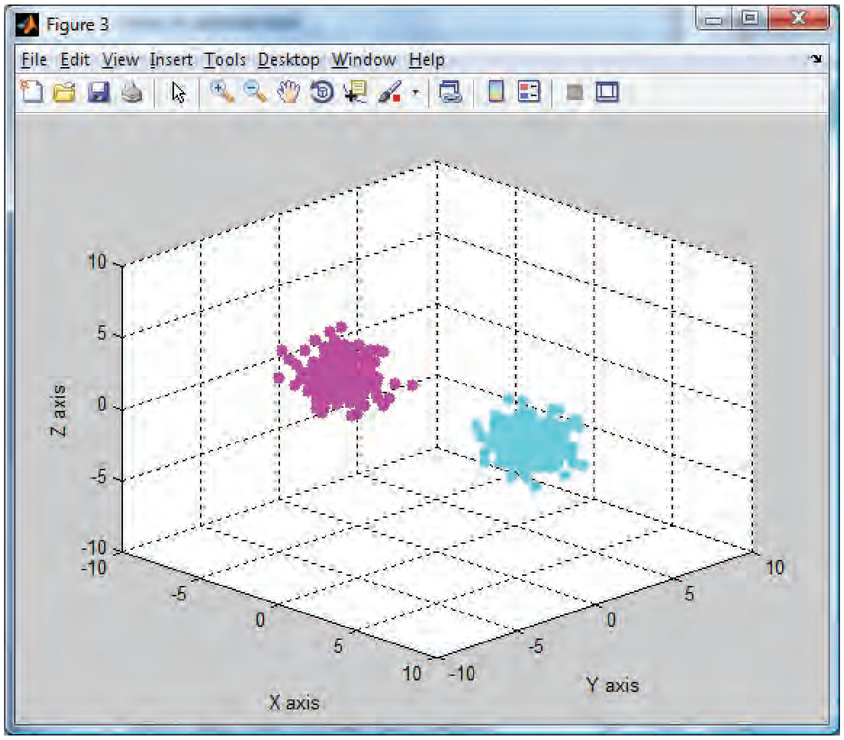


Figure 11: Rotating the view point shows the two data classes are separate

This is an important demonstration of the significant effect that point occlusion and viewing position can have on our interpretation of high dimensional data. A MATLAB function was written and included with this report to sort 3D data into depth order from the perspective of the camera position. The header section of that function is shown below and the complete function is included in the Appendix A.5.

```
function [ sData, distClr, ptSz ] = cameraDistanceSort( data, camPos, maxPtSz )
        %Sort data be distance from the camera position for depth perception
        %   data - the display data the first 3 columns are treated as X,Y,Z
        %   camPos - the [ X Y Z ] position of the camera
        %   maxPtSz - integer maximum point size (minimum 30)
        %   sData - the input data with rows sorted by distance form the camPos
        %   distClr - encodes distance from camPos in a range of 1-64
        %           (column vector equal in size to the rows of data for colour mapping)
        %   ptSz - encodes the size of the display point for each data point
        %           (column vector equal in size to the rows of data)
```

The function inputs include the *data* matrix, the camera position vector *camPo,* and a scalar value *maxPtSz* for the maximum point size. The *data* matrix can be any dimensionality of three or higher, where the first three columns represent the X, Y, and Z values that will be displayed. Individual data points are represented as the rows of the *data* matrix. The function returns the *sData* matrix which is the original *data* matrix with the rows sorted into depth order from the perspective of the camera position. The returned *distClr* column vector has the same number of rows as the *sData* matrix and it contains the depth distance from the camera position encoded into a range of 1 to 64. This vector is provided for colour coding the data by distance from the camera, however, it can be ignored if colour is being used for other purposes. The last return argument is also a column vector *ptSz* which is again the same length as the number of rows in *sData*. This vector is provided to allow encoding the display point size to represent distance from the camera; the values range from 5 to *maxPtSz*.

The return arguments could be used in different combinations for graphing, for example:

```
scatter3( sData(:,1), sData(:,2), sData(:,3), ptSz, distClr, ‘filled’);

scatter3( sData(:,1), sData(:,2), sData(:,3), 30, distClr, ‘filled’);

scatter3( sData(:,1), sData(:,2), sData(:,3), ptSz, myColours, ‘filled’);
```

5 Stereoscopic 3D Presentation

5.1 Introduction

True 3D imaging technologies were examined for easy of use, realistic depth perception, and required support technology. Three technologies were selected for examination:

- Anaglyph Stereogram
- Side-by-side Stereogram
- Interlaced Stereogram with LCD shutter glasses

The hardware and supporting software for each of these technologies was acquired and tested. MATLAB software was used to process ESM/ELINT data to produce the required data for many of the visualizations.

5.2 Anaglyphs

Anaglyph stereograms use color encoding of the right and left eye images so that both images can occupy the same display space on the page or screen. The image encoding must use colours that are chromatically opposite so that the images can be separated by glasses with matching coloured lens. Although the red-cyan anaglyph is the most common, it is far from the only possible colour combination. The Wikipedia site[5] lists ten anaglyph colour combinations that have or are being used. PROVIEW[6] red-cyan glasses with optical quality, cast acrylic lens, were selected for testing in this project.

Figure 12 contains a red-cyan anaglyph created from radar intercept data using the *ANAGLYPH* program which is available free from the MATLAB file exchange service. When the anaglyph image is viewed with the red-cyan glasses the 3D effect has not particularly good. Ghosting was also observed with this anaglyph. It is possible that there is a defect in the *ANAGLYPH* program since there appears to be some vertical error in the registration of the two images as evidenced by the vertical displacement red-cyan images of the Z axis scale. The red-cyan images were expected to have more horizontal displacement differences than vertical.

[5] http://en.wikipedia.org/wiki/Anaglyph_image#Dual_purpose.2C_2D_or_3D_.22compatible_anaglyph.22_technique {2010/10/30}

[6] http://www.rainbowsymphony.com/pro3d.html {2010/10/30}

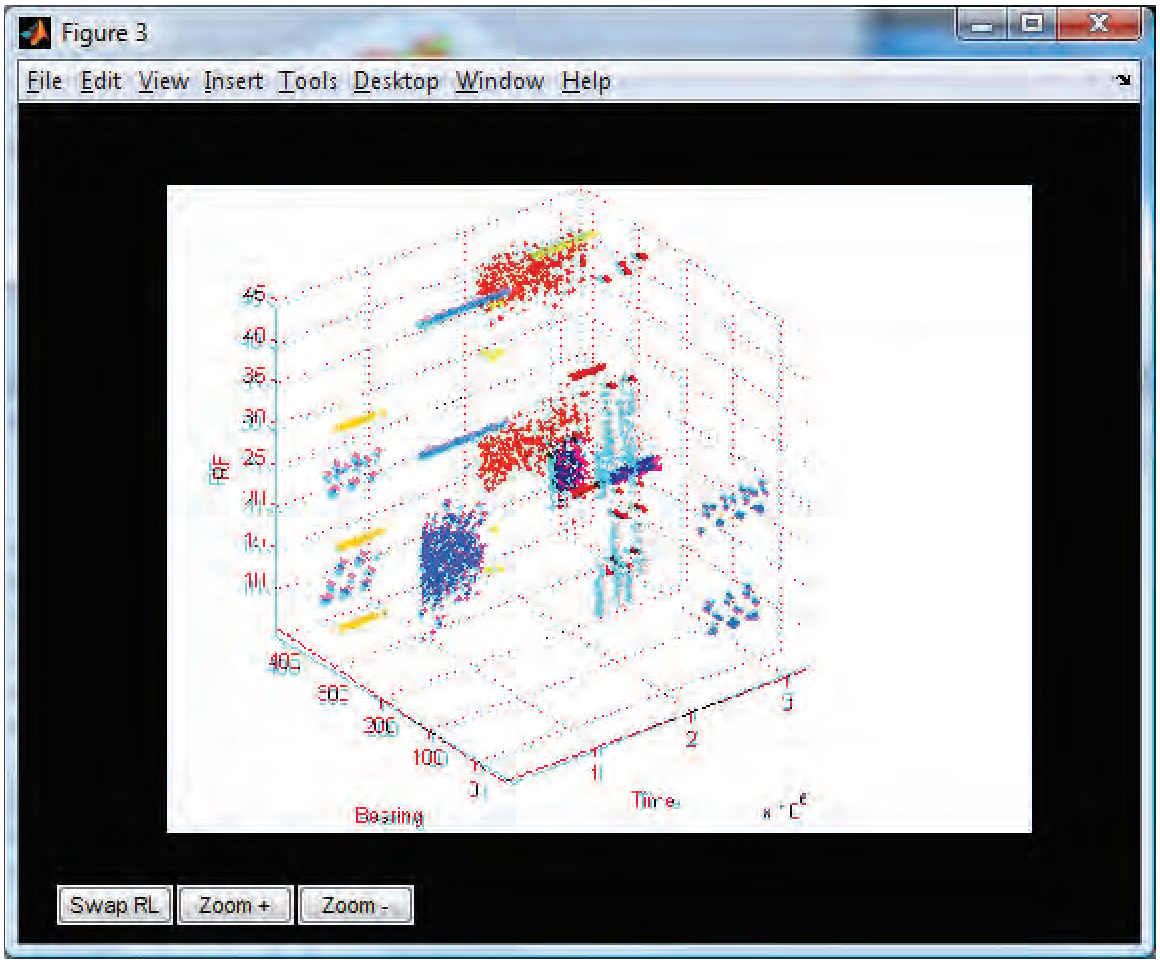


Figure 12: Anaglyph image created by ANAGLYPH by I.G. Marino

For comparison the same scatter plot was captured as a stereo pair and transformed to an anaglyph stereogram by the PokeScope Stereoscopic Software Pro Edition ver.2.6[7]. This software can accept discrete left and right images, or a single image that contains a stereo pair. The image pairs can be edited to correct cropping, rotation, or keystone effects in one or both images. Note that this software may not be supported much longer; the software purchased during this tasking had a 2004 copyright and was not compatible with Windows Vista, however it would work under XP. Refer to Section 5.5 for new software.

A large number of stereograms are available for viewing on the site PHEREO[8] and each stereogram can be viewed in many stereo formats including anaglyph format, which facilities comparing the available stereogram techniques. Examination of the colour anaglyphs on that site indicated that the colour information appears to be encoded only in the right (cyan) channel, with the left (red) channel being a monochrome image. It was also noted that the red channel (left) of the glasses bled in information from the right channel which caused ghosting on the image. The

[7] http://www.berezin.com/3d/pokescope.htm {2010/10/30}
[8] http://phereo.com/ {2010/10/30}

red-cyan glasses used for testing did not produce clean 3D effects with these images either. It should be noted that the red-cyan glasses were good quality PROVIEW hard lens design; not the cheap cardboard glasses normally included with 3D books.

5.3 Stereo Pairs

A stereo pair is a stereogram consisting of two images representing two perspectives of the same object, each presented to a different eye of the viewer. The minor deviation in the perspective of the two images is arranged to be equal to the perspectives that both eyes would naturally receive in binocular vision. The spatial separation of the images should be such that the viewer's eyes are relaxed and pointing straight forward when viewing items at infinity in the images. If the images are large, the viewing device can compensate by folding the image paths down to standard eye spacing.

Stereo pairs can be arranged side by side or above and below. The above and below stereo pairs are rare and generally only used for images with unusually long aspect ratios. Above and below stereo pairs always require a viewing device and side by side stereo pairs will normally employ a viewing device. The cross-eyed side by side stereo pair has the left and the right images reversed and is designed for unaided viewing, however, the cross-eyed viewing technique is neither easy nor comfortable. Stereo pair stereograms with stereoscopic viewers were invented prior to the 1900's and stereo pair stereograms have played a significant role in air reconnaissance since World War I.

The principal advantage of a stereo pair stereogram over an anaglyph stereogram is that the stereo pair images can be high resolution with full spectrum colour. Since the stereo pair present a dedicated image for each eye there is no bleeding of information from one image into the other. Stereo pairs do not have the ghost images around the edges of objects that can occur in anaglyph stereograms due to imperfect anaglyph encoding during production or imperfect filtering during viewing.

Figure 13 below shows a stereo pair stereogram created from radar intercept data using the *Stereoview* program which is available free from the MATLAB file exchange service. This figure represents the same data as was used for the anaglyph stereogram in Figure 12. The stereo pair stereogram illustrates a second advantage of stereo pair stereograms over anaglyph stereograms, namely that users without viewing devices can still easily view each image as uncluttered 2D images of the data. By contrast when viewing an anaglyph stereogram without filter glasses it is difficult to see the details in the image.

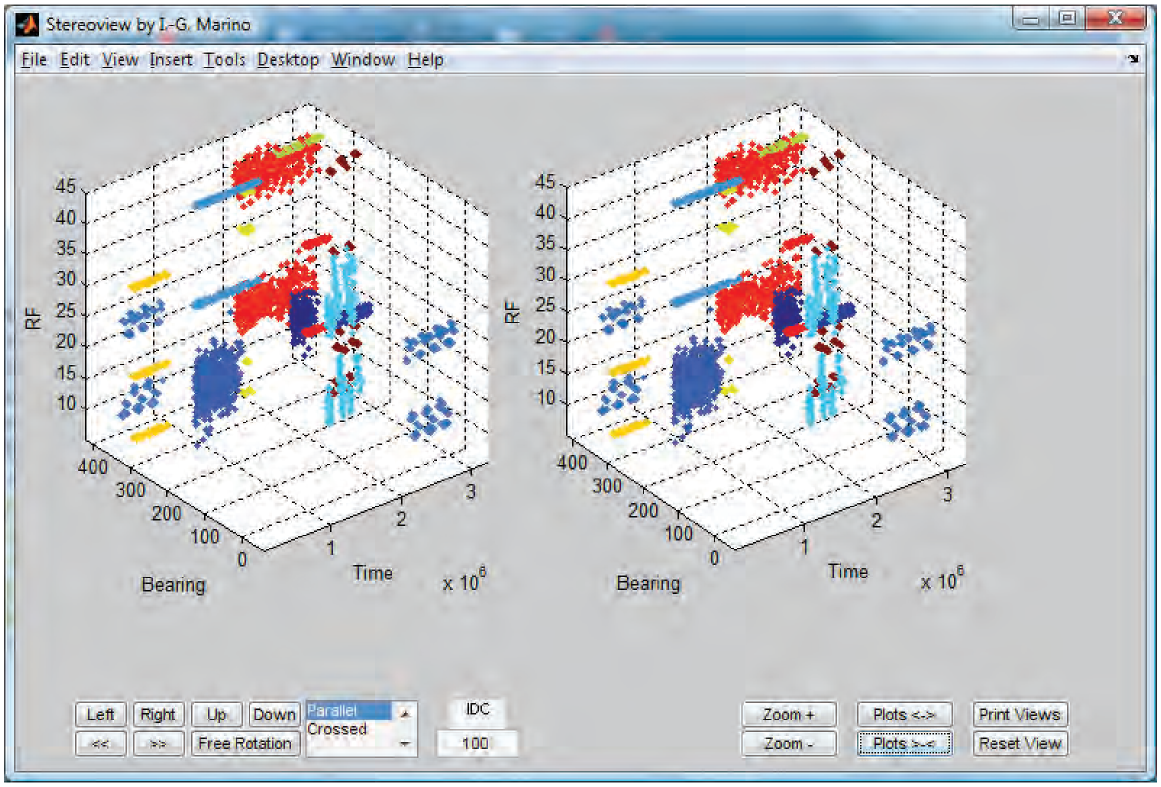


Figure 13: Stereoview a MATLAB file exchange program by I.G. Marino

The principal drawback of stereo pairs is that the viewing devices tend to be more expensive than those required for anaglyph stereograms. The complexity and the expense of the stereo pair viewer depend on the technology used to present the stereo pair. For static image stereo pairs the viewer can be very modestly priced and still have good performance. The View-Master, which contained stereo image pairs around the perimeter of a disk, was introduced in 1939 for viewing tourist sites in 3D and was later adopted by the US military during World War II for training in aircraft recognition and other tasks. This project used the PokeScope[9] shown in Figure 14 below, which costs approximately $50 and works with static images or dynamic images, on paper or on a computer screen.

[9] http://www.pokescope.com/ {2010/10/30}

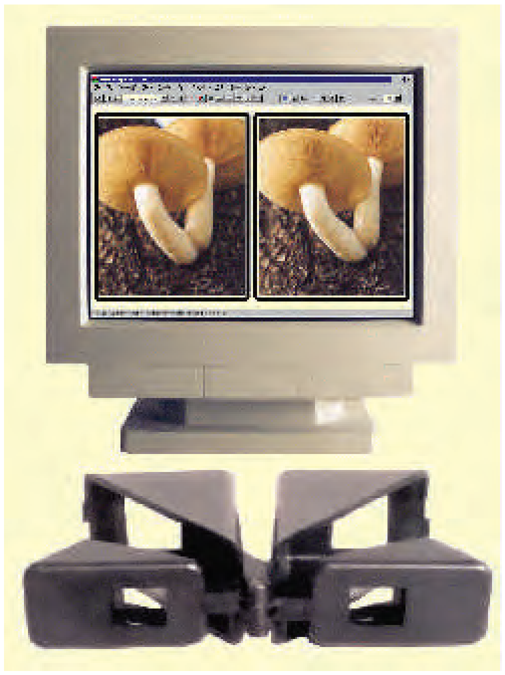


Figure 14: PokeScope stereogram viewer for side-by-side Stereo Pairs

A direct A-B comparison of anaglyph versus stereo pair viewing is possible on the PHEREO web site since it can display any of the stereograms in any one of several formats, including anaglyphs and stereo pairs. The performance of the PokeScope viewer was compared to that of the red-cyan PROVIEW glasses. Unlike the anaglyph glasses the PokeScope presented a clear and realistic 3D view that was free of any ghosting whatsoever. Even a cheaper pair of folding prismatic glasses presented a clear realistic 3D view of the stereo pair images that was far superior to the anaglyph glasses performance.

The requirement to have objects at infinity appear to both eyes to be straight ahead without significant divergence means that side by side images cannot be very large if the viewing device is to be compact, like the PokeScope. This effect was clearly noted when the stereo pair images were enlarged to full screen and the image separation was significantly greater than normal eye separation. Neither the PokeScope nor the prismatic glasses had any capability to fold the viewing path down to normal eye separation distances so the viewer was forced to move back from the screen to reduce the angular divergence. This was not a problem for the PokeScope which accommodates some angular adjustment, however, the prismatic glasses have no angular adjustment and they include magnifying lens for close working distances so they could not be used from longer distances. Moving back from the display nullifies the effect of enlarging the image so there is little advantage to doing so unless the purpose is to allow multiple people to view the image at the same time.

5.4 Interlaced Stereo Displays

An alternative to the normal stereo pair viewing, which does support larger images, is available to dynamic media where both images can share the same display space and have the viewing device separate the images. The images can be interlaced in time and separated by shutter glasses that alternately open and close for each eye, or the images can be cross polarized and separated by glasses with matching crossed polarized lens. Both approaches will have somewhat less bright images, however, the cross polarization approach may suffer more bleeding of information between channels. If vertical and horizontal polarization is used, the viewer must take care to keep the glasses correctly oriented to the display, otherwise significant bleeding can occur between left and right channels.

The LDC shutter glasses are unaffected by the orientation of the glasses with respect to the screen; however it should be noted that normally the 3D depth effect being simulated on the screen assumes the viewers eyes are in the horizontal plane and it will not compensate for head motion. If the shutter glasses are equipped with a head tracking system the 3D effect could be made to track the viewers head movements. In real life, tilting one's head (roll axis) causes the plane for the viewing parallax to also roll, which changes the observed parallax to include more components of vertical parallax and fewer components of horizontal parallax. With most basic shutter glasses systems there is no head tracking and the plane of parallax remains horizontal at all times.

A background investigation of 3D visualization techniques and a survey of available consumer grade 3D equipment were conducted and hardware for a simple 3D computer display system was obtained. The selected 3D display system is based on a Samsung Syncmaster 2233RZ 120Hz LCD monitor and a pair of nVidia LCD shutter glasses which are pictured in Figure 15 below. The PC that will drive the system requires a 3D capable graphics processor unit (GPU) and an upgraded power supply to handle the power requirements of the GPU. A GeForce 9800GT GPU was selected as a mid-range performance GPU which is based on nVidia's Unified Architecture and certified for nVidia 3D compatibility. Note the software drivers for the nVidia 3D glasses require Windows Vista and will not work with XP or earlier versions of Windows. Also note that the 120Hz monitor must be connected to the GPU card with the dual link DVI cable that came with the monitor since it actually has more pins in the DVI connector than a standard DVI cable and although it operates as a standard monitor with the standard cable, it will not function in the 3D mode without the dual link cable.

Installation of the GPU card was a straight swap of graphics cards. Next the GPU drivers were installed and then the nVidia 3D glasses software was installed. Running the 3D setup and the 3D demo confirmed that everything was working correctly. The 3D depth effects on the demo are very good and the viewing is very relaxed and natural; there is no need to work at seeing the effect and there is no delay waiting for the 3D to "pop" into view like there can be with stereo pair and anaglyph images.

The nVidia 3D shutter glasses require the application to switch the GPU into 3D display mode before the glasses will function. Since the MATLAB graphics and the MATLAB code developed under this tasking do not interface directly to the GPU, the glasses were not used directly from MATLAB. Instead the MATLAB data graph was captured as a stereo pair image and saved as left and right JPG images files. Then the 3D image viewer utility that came with the GPU software was used to present the stereo pair images as an interlaced display for the 3D glasses. This multi-step process was acceptable for hardware evaluation, however, it would not be acceptable for use in an analyst workstation. Luckily there are several options available to integrate the analyst workstation software with the GPU hardware, however, they involve writing interface code to the available 3D graphics libraries (e.g., OpenGL or Java 3D or DirectX).

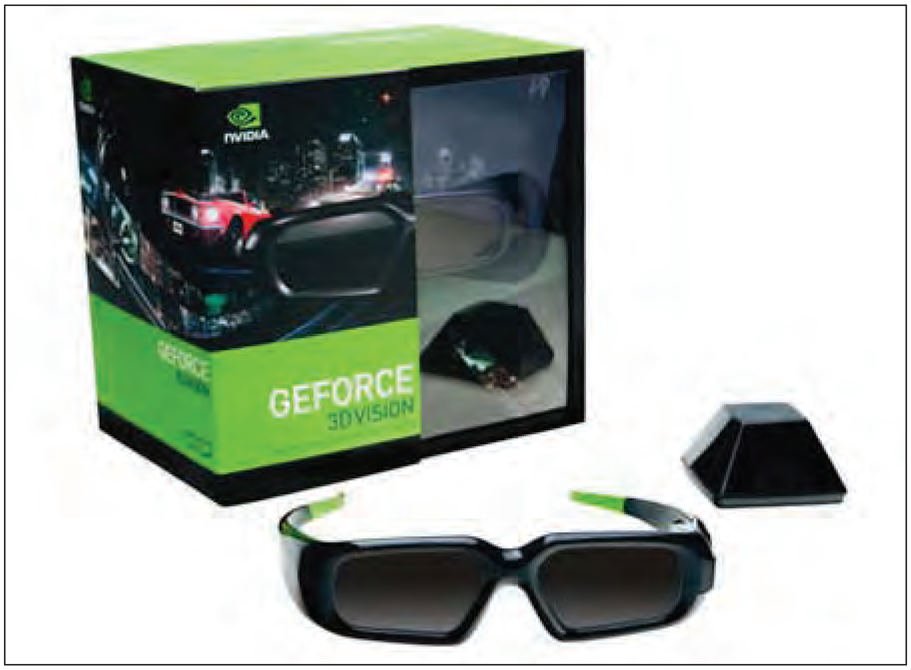


Figure 15: nVidia 3D Shutter Glasses with Infrared Controller for Synchronization

The separate left and right images of the nDim dataset, from Figure 13, were displayed as interlaced 3D for the shutter glasses using the nVidia Stereoscopic 3D Viewer utility program. The 3D depth effect was visible but not as strong as it was for the images in the nVidia demonstration software. This may have to do with angular separation that was used to produce the stereo pair images in MATLAB. In addition, the initial tests had ghosting of the X and Y axes labels, indicating incorrect alignment of the left and right images.

A stereo image editor called StereoPhoto Maker[10] was downloaded and used to fix the alignment of left and right images for the nDim graph. The program features an automatic fix process for alignment of stereo images which rotated each image slightly in addition to some other small adjustments. The resulting corrected stereo images were again viewed with the shutter glasses and this time there was no significant ghosting of the axes labels. The StereoPhoto Maker has several other useful editing functions and it is able to import and export multiple formats, making it a useful tool to have and a surprisingly capable tool, considering that it is freeware.

Compared to an anaglyph or stereo pair stereogram, the nVidia shutter glasses can produce 3D depth effects that are the most natural for the viewer and induce very little eye strain. The shutter glasses have the added benefit that non 3D images and the keyboard all appear normal while the

[10] http://stereo.jpn.org/eng/stphmkr/index.html {2010/10/30}

glasses are operating, which means they can be left on and still easily operate the workstation controls.

5.5 Software for Stereo Imaging

As previously noted in Section 5.2 the PokeScope Stereoscopic Software Pro Edition ver.2.6[11] that was purchased with the PokeScope side by side stereo pair viewer, appears to be out of date with no apparent replacement in sight. The software does operate on Windows XP but will not install on Windows Vista, and Vista is required to support the nVidia 3D shutter glasses, therefore the PokeScope Stereoscopic Software is not recommended.

StereoPhoto Maker[12] is a freeware stereo image editor that appears to provide all the functionality of the PokeScope Stereoscopic Software and more. The automatic optimum re-alignment of left and right images of a stereo pair is a particularly useful feature. This software can also import and export several file formats common to stereo imaging, including the JPS format for side by side stereo pair JPEG files that the nVidia Stereoscopic 3D Viewer can accept as input. Several other stereo utility programs including a StereoMovie Maker[13] are available from the same site[14].

If the analyst workstation is to support stereo image display as a native part of the workstation tools then the analyst software must be interfaced to the Application Program Interface (API) for the graphics processing system. When the MATLAB system displays a 3D perspective scatter plot, the rendering of the data into a 2D image is normally performed within the MATLAB code. MATLAB computes the view from the camera position and target position and incorporates all the effects for lighting and texture to render the 2D image that is sent to the GPU for display. To use the 3D shutter glasses or polarized glasses to view a stereo image, the data must be sent to the GPU as a 3D model that the GPU renders and, therefore, the GPU has the 3D model information required to render the correct perspective for each eye and to send each image at the correct time (for shutter glasses) or to the correct projector (for projection for polarized glasses).

Passing a 3D data model to the GPU and letting the GPU compute the perspective views, the occlusion of background items, and other effects of depth on size and texture; means that many of the depth cues discussed in Section 4.2 come almost for free. In particular the erroneous occlusion of foreground points by background points (refer: Section 4.2.3) should not occur if the GPU renders the 3D data model. Manipulations such as rotating the data or flying through the data should all execute significantly faster on the GPU.

Work is currently underway[15] by many independent parties to integrate MATLAB and other mathematical or numerical software packages with the GPU API, however, this work is not main stream yet. We may still be a couple years from seeing stereo image visualization as a standard feature of MATLAB.

[11] http://www.berezin.com/3d/pokescope.htm {2010/10/30}
[12] http://stereo.jpn.org/eng/stphmkr/index.html {2010/10/30}
[13] http://stereo.jpn.org/eng/stvmkr/index.html {2010/10/30}
[14] http://stereo.jpn.org/eng/index.html {2010/10/30}
[15] http://www.nvidia.com/object/numerical-packages.html {2010/10/30}

The primary support libraries for 3D and stereo imaging that are significant for the development of custom software for analyst workstations are the following:

- Open Graphics Library (OpenGL[16])
- Microsoft DirectX[17]
- Java 3D[18]

DirectX currently dominates the 3D game market whereas OpenGL dominates the high end professional workstations. Each has aspirations to dominate both fields in the future. Java 3D is not a competitor, instead it runs atop either OpenGL or Direct3D, and it provides an object-oriented abstraction of the API's as a Java class library. Until MATLAB is fully integrated with OpenGL or DirectX, Java 3D may be the easiest route to add 3D display support for shutter glasses, since MATLAB is fully integrated with Java.

Note that not all 3D capable GPU cards can support both OpenGL and DirectX; some cards are dedicated to one API. Although the GeForce 9800GT used for these tests has modest capability and price, it did support both DirectX 10 and OpenGL 2.1. This is an older GPU and at the time of publication the current version of OpenGL was 4.1[19] and the current version of DirectX was 11[20].

[16] http://www.opengl.org/ {2010/10/30}
[17] http://www.microsoft.com/games/en-US/aboutGFW/pages/directx.aspx {2010/10/30}
[18] https://java3d.dev.java.net/ {2010/10/30}
[19] http://www.opengl.org/documentation/current_version/ {2010/10/30}
[20] http://www.microsoft.com/games/en-us/aboutgfw/pages/directx.aspx {2010/10/30}

6 Hyper Dimentional Data

6.1 Introduction

Hyper dimensional, or high dimensional, data is a term used to refer to data having more than 3 dimensions. Dimensions can be spatial or non-spatial in nature. Everyone is familiar with the three spatial dimensions that we live in, width, height and depth (typically X, Y, Z for displays, or X, Z, Y for mathematics). Data dimensions are not limited to spatial attributes, for example, most people have heard the expression that time is the fourth dimension. Data dimensions can represent any data attribute that varies over a range of values, which can be continuous, non-continuous, or discrete valued. A survey of people that records sex, age, marital status, number of children, education, income, assets and total debit, would be an example of hyper dimensional data (8 dimensions).

ESM and ELINT signal analysis normally involves hyper dimensional data. The signal attributes of a standard radar intercept can include radio frequency (RF), pulse width (PW), signal to noise ratio (S/N), RF bandwidth (BW), and polarization (Pol). To these radar pulse attributes we can add attributes of the measurement time of arrival (TOA), line of bearing (LOB), and elevation (El). All eight of these dimensions are measureable for each radar pulse. Once the pulses are grouped into pulse trains, many more attributes become available to describe the signal train.

ESM and ELINT intercept receivers that sample and analyze the waveforms of individual pulses are performing intra-pulse (IP) analysis. IP analysis can be used to distinguish between emissions from members of the same type of radar as well as radars of different types. In many systems the IP analysis will result in a characterization vector with 10-30+ values in each vector. A 16 element characterization vector will add 16 more dimensions to the 8 that we already had for each pulse. In order to associate emissions together, or to distinguish between emissions, using the IP characterization vector, we must form clusters and then separate the measurements in the hyper dimensional space. Hyper dimensional data visualization techniques are applicable to both manual signal analysis and manual verification of automated signal processing.

6.2 Test Data

CANEWS AN/SLQ-501, the Canadian Naval Electronic Warfare System (CANEWS) is the Electronic Support Measures (ESM) system used by the Canadian Navy. CANEWS 2 was designed and developed as the Canadian replacement for the original CANEWS system. As part of the development and testing of the CANEWS 2 prototype, a recording system was developed to record the intercepted radar signals that the CANEWS and CANEWS 2 ADM systems received. The recording system captured the digital pulse descriptor words that describe each intercepted radar pulse. An **xwave** pamphlet describes the recording system:

> ESM Recording, Analysis and Playback System (ERAPS) was developed to test various aspects of CANEWS. ERAPS provides the facilities and tools to monitor and evaluate a CANEWS test-bed performance in a real-time environment. This system provides recording and playback facilities in a laboratory environment. ERAPS can be used to record digital radar data produced by CANEWS, as well as replay recorded or simulated

digital-pulse data back into the CANEWS test-bed. This tool is particularly useful in evaluating ESM algorithm performance.[21]

These recorded signals were critical to the testing and evolution of the CANEWS 2 design. The recordings contained all the signal artefacts that a real radar ESM environment can contain: measurement errors, signal reflections, multipath interference, anomalous propagation (i.e., ducting) and more. Unlike live signals from a test range, the recorded signals were exactly repeatable, making them invaluable for testing and evaluation of enhancements.

Another critical aspect of having recorded real data was that the recordings could be manually analyzed to classify the signals that appear on the recording and then the digital pulse descriptor words for the pulses could be augmented with a ground truth identification tag to assist in any future testing of the system. The effort devoted to the analysis of recorded radar intercepts to provide quality testing data, represents about 3-6 person years of effort.

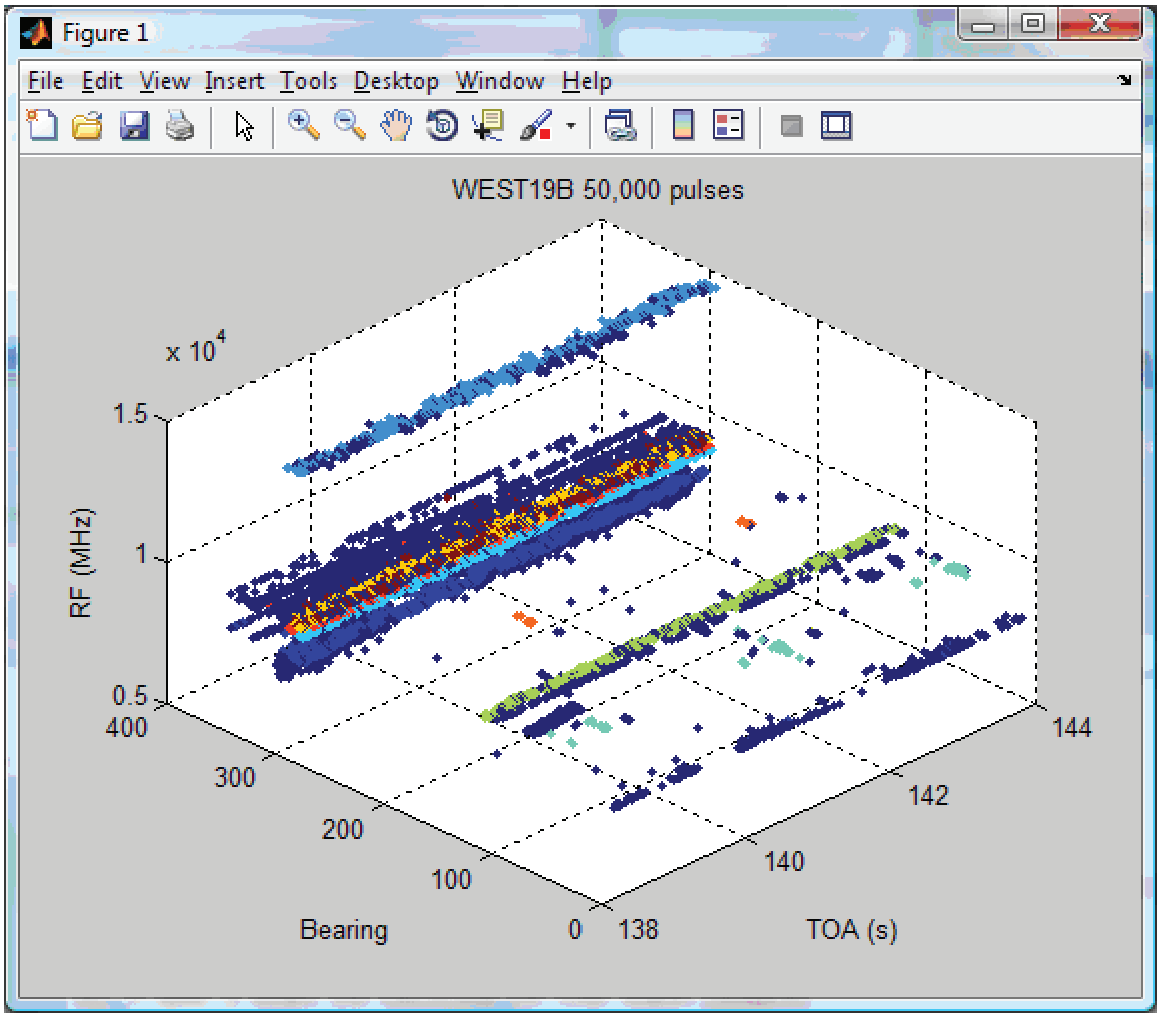


Figure 16: WEST19B recorded radar intercepts

[21] http://www.xwave.com/files/credentials/electronic_warfare.pdf {2010/10/30}

WEST19B is a radar intercept recording dataset; it is not a simulation. The recording was made during a NATO sea exercise involving ships and aircraft from Canada, the US, and Europe. WEST19B is one of the datasets that has the added ground truth emitter IDs and that was used in the development and testing of the CANEWS 2 laboratory prototype.

Figure 16 shows the first 50,000 pulses from the WEST19B dataset which includes a total of 1,073,360 pulses. It is clear from the figure that the dataset has moderately high pulse density; in fact it has approximately 8000 pulses per second.

Portions of WEST19B were used in some of the testing performed under this tasking, however, the WEST19B dataset only includes basic radar signal parameter values so it does not provide us with a large number of data dimensions. Radar intercepts with measured intra-pulse features would provide the necessary high dimensionality, however intra-pulse techniques are normally classified and for this study, actual intra-pulse data is unnecessary. Unclassified intra-pulse data was simulated for use in this study.

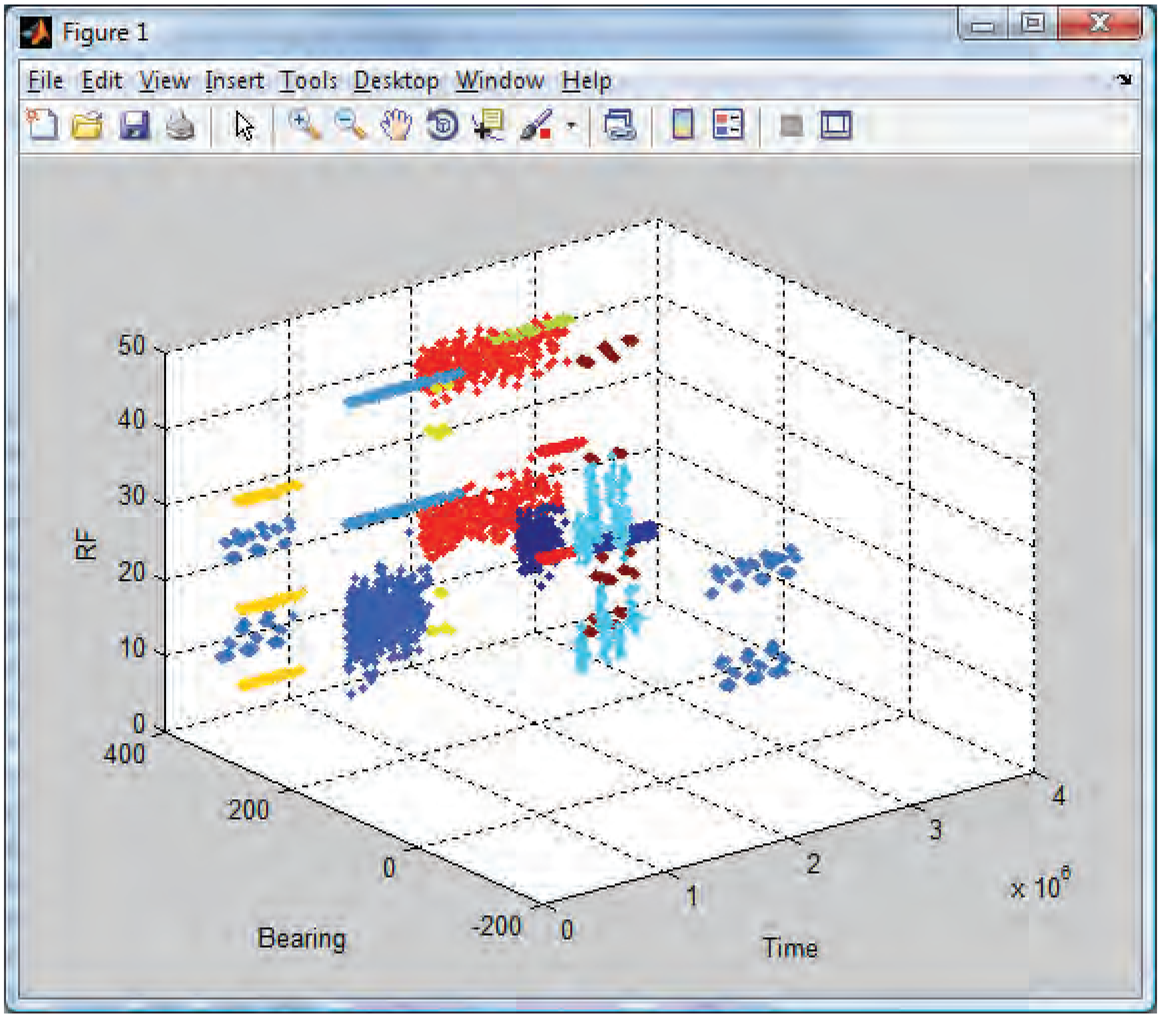


Figure 17: nDim 5,ooo simulated radar pulses

Figure 17 displays the 5000 pulse simulated dataset that includes unclassified intra-pulse features (IP features not shown in this figure) and will be referred to as the nDim dataset (N-Dimensional). The figure shows the same basic radar signal parameters as were displayed for the WEST19B dataset. Another view of the nDim dataset in Figure 18 shows a plot of bearing, RF, and pulse width (PW).

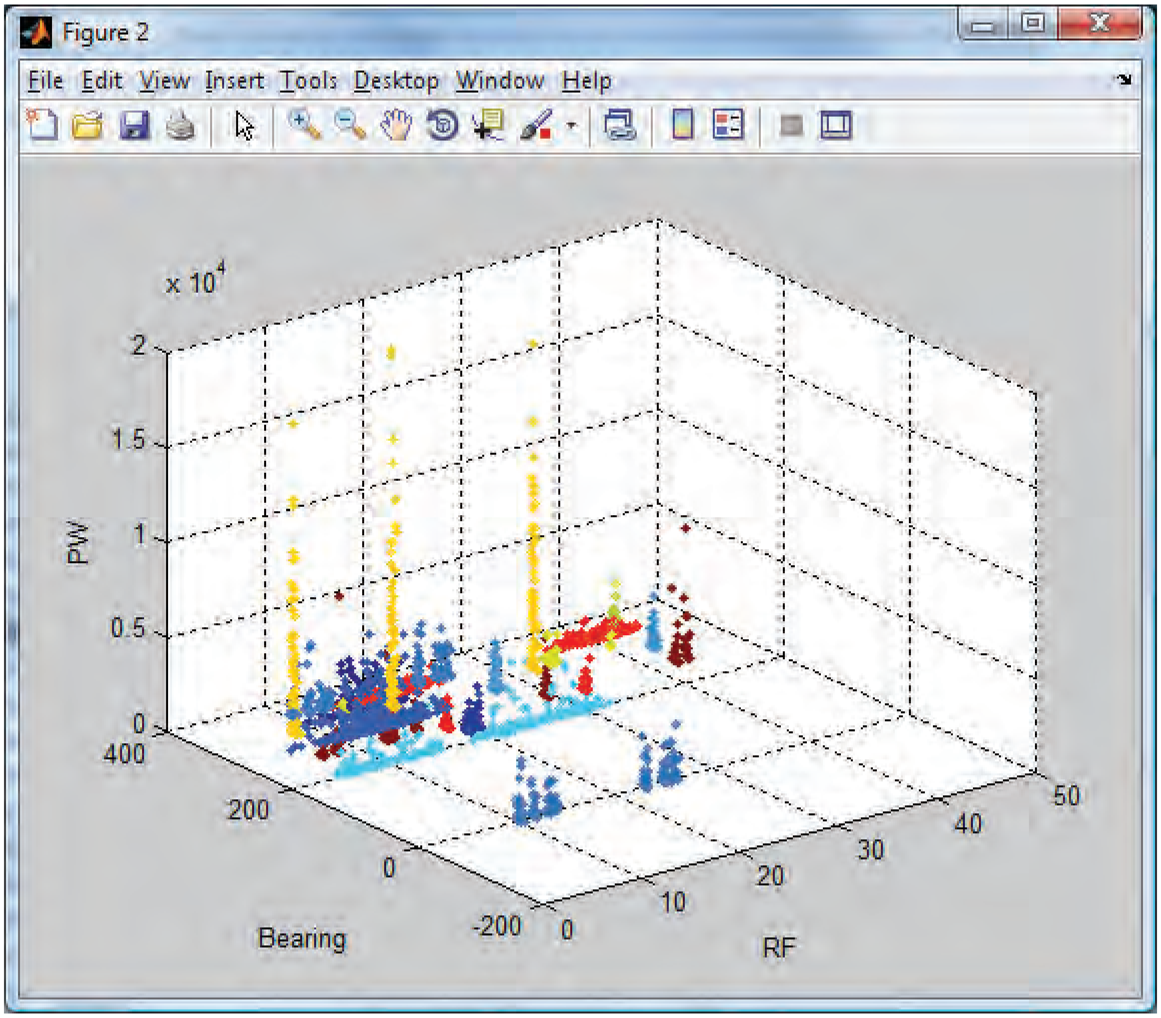


Figure 18: nDim simulated radar intercept parameters

Each pulse of the nDim simulated data includes 24 intra-pulse coefficient values that will be used for the hyper dimensional studies. Although these coefficients are only simulated values they are near enough to true IP values to serve our purposes.

7 Visualizing Hyper Dimentional Data

7.1 Parallel Coordinate Graphs

The Parallel Coordinates plot is one of the few data visualization techniques that do not involve dimensional reduction or dimensionally limited views; it is capable of displaying all of the dimensions from the hyper dimensional data set. In a Parallel Coordinates plot the N axes are arranged as parallel lines, usually equally spaced and scaled so that the data in each dimension occupies the same vertical display space. A point in the N dimensional data space is represented as a polygonal line in the parallel coordinates space. In Figure 19, the polygonal line in the plot represents the point (0.6, 1.6, -0.8, 1.2) in the 4D data space.

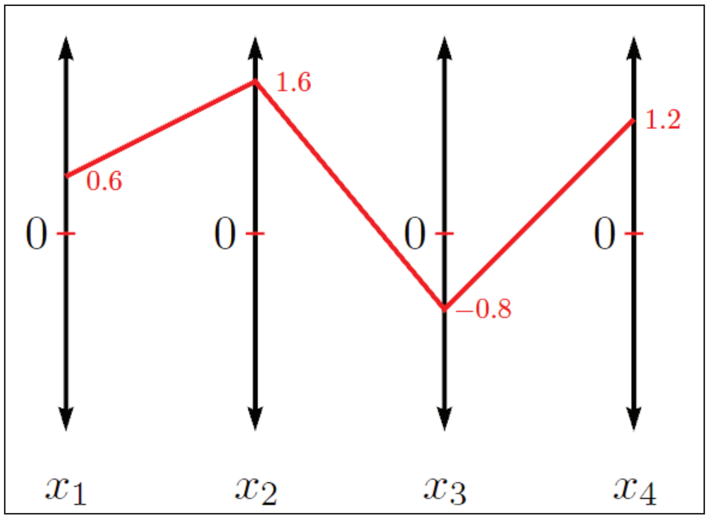


Figure 19: Parallel Coordinates plot of a single point

MATLAB supports parallel coordinates as a standard graphic display technique and Figure 20 shows the MATLAB view of the same (0.6, 1.6, -0.8, 1.2) data point. The MATLAB version uses a single vertical axis on the left so it is important that all dimensions be scaled either before plotting or by one of the techniques provided by MATLAB. Using the MATLAB provided techniques has the advantage that MATLAB handles the reverse transformation to data space values when the cursor position is read. The down side to using the provided MATLAB scaling is that each plot is scaled for the data it contains, which can make comparison to other graphs more difficult, and in some cases the scaling can exaggerate the noise or variation in sample values by scaling it to fill the coordinate axis. Generally, if the data set contains many distinct classes or groups of data then MATLAB scaling may be useful; however, if the data set contains samples from a single class or family, then the MATLAB scaling will magnify the variation between samples until the plot occupies most of each coordinate axis.

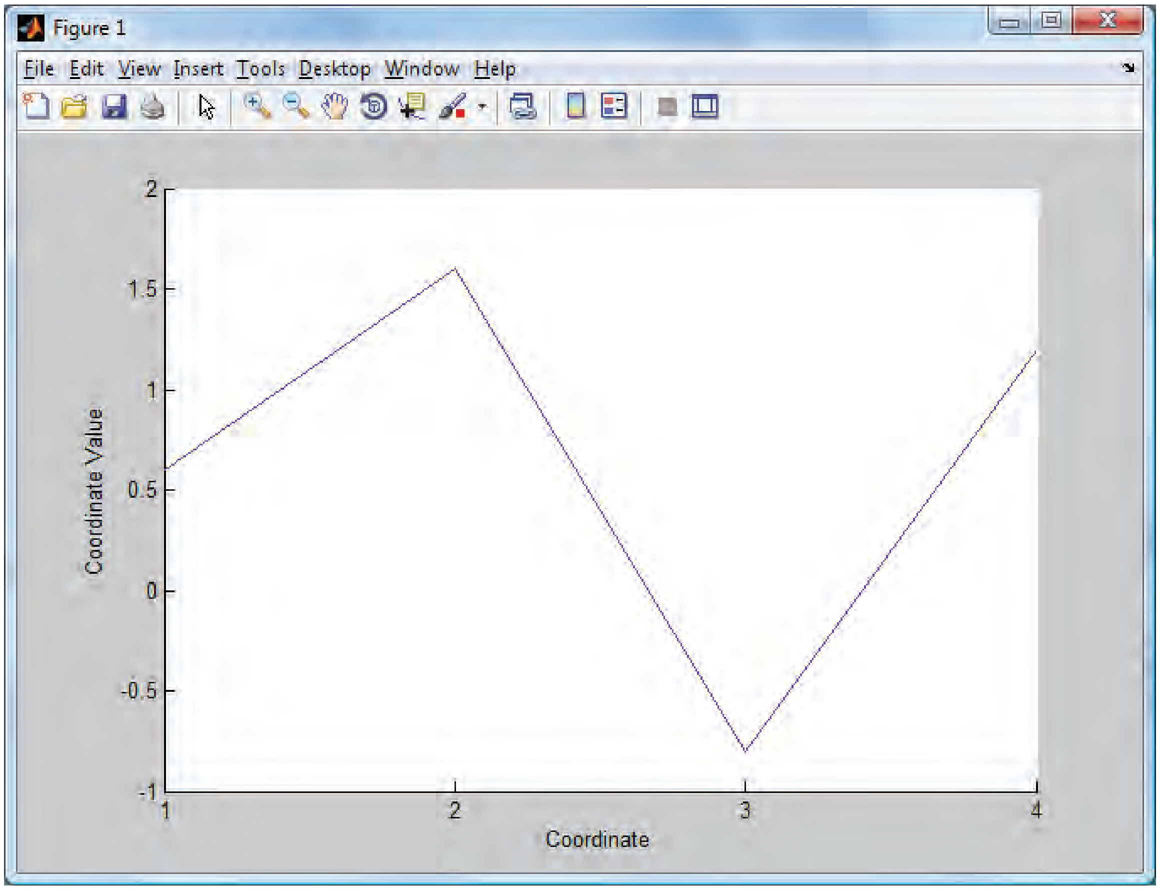


Figure 20: MATLAB Parallel Coordinates Graph Representation

Parallel coordinates systems are not intuitive and require practice to interpret but they are an important tool for high dimensional data. Since each polygonal line represents a single point in the data space, the intersection of two lines on parallel coordinates space represents a pair of points and, therefore, defines a line in data space.

In general, groups of lines that cluster near a single value on any axis will indicate structure or constraints in the data. In Figure 21 many polygonal lines intersect at the same point on the third axis, which can be interpreted as many lines on a hyper plane, which is normal to the third axis.

Parallel coordinate plots can be used for both analysis and for visual feedback during navigation through the data space. The location of the viewpoint can be displayed as a polygonal line and the viewpoint could be modified by dragging the polygonal coordinates of the viewpoint to the desired display region based on the data display in the polygonal coordinates plot. The displayed data can be selected, coloured, or filtered by selecting segments of different axes in the parallel coordinate plot.

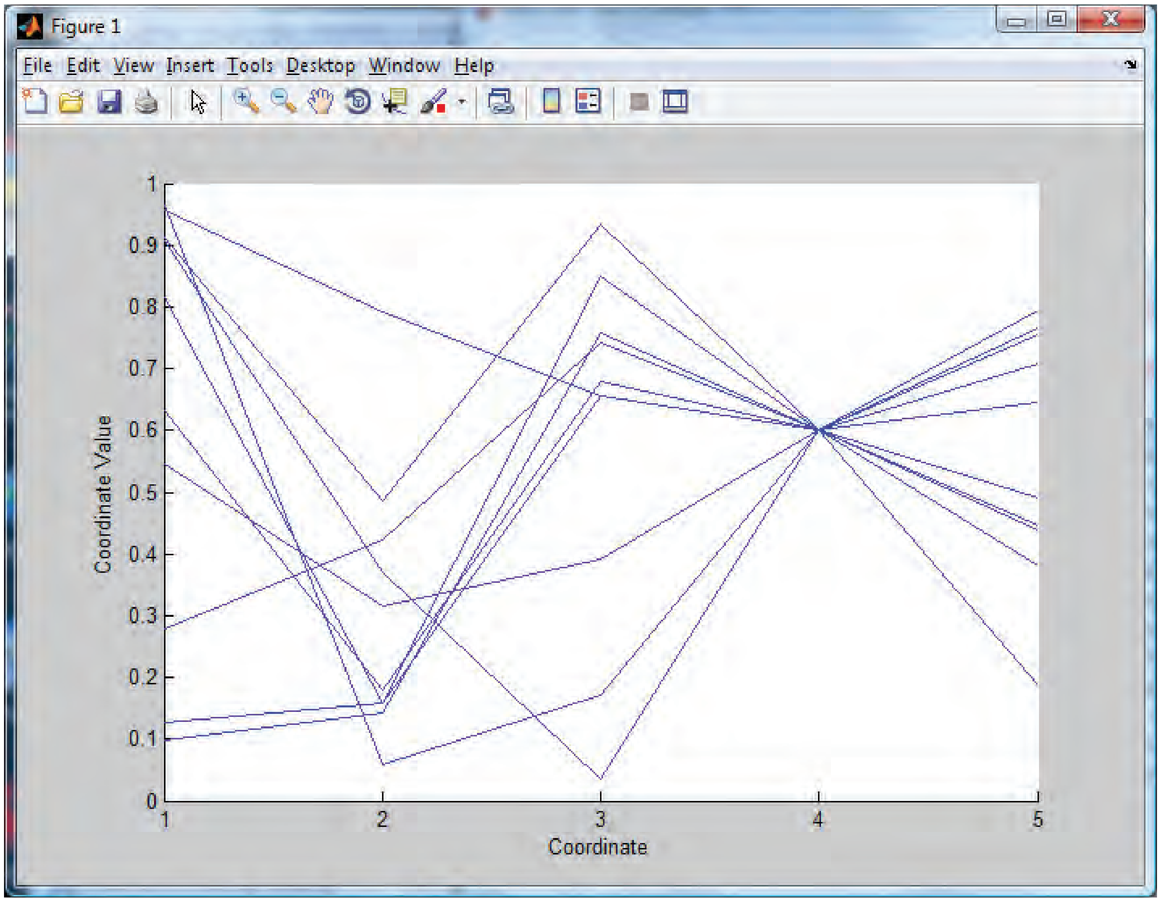


Figure 21: Interpreting a Parallel Coordinates Graph

7.2 Signal Analysis using Parallel Coordinates Graphs

The parallel coordinates view is well suited to the analysis of intra-pulse (IP) features of radar ESM intercepts since many IP measurement approaches produce a large number of coefficient values. It is not unusual to have 16 to 64 or more IP coefficients for each sampled radar pulse. In the following figures the simulated IP samples from the nDim dataset have 24 coefficients, each ranging in value from 0 to 1.0, which represents a 24D hyper dimensional dataset.

Figure 22 depicts a set of 24 separate radar pulses from Emitter 85, which have been sampled to produce 24 IP coefficient values for each pulse. This pulse dataset has a high degree of similarity in the IP values across all of the pulses in the dataset. The pattern is characterized by a few distinct peaks at coordinates 6, 11, 14, and 16. It is not uncommon for there to be a wider variation in the IP sample values which leads to a less distinct pattern. Figure 23 shows 48 sampled pulses from Emitter 22, and it illustrates a case where there is a significant variation in coefficient values between pulse samples. The variation can be caused by inherent variations in the pulse itself, or it can be produced by the measurement conditions, such as low signal to noise ratio.

The IP coefficient values for Emitter 22 displays two interesting aspects to the variations. The first and expected variation is the random looking values that are spread across all coordinates, except for 11 and 12 which seem to force a more consistent behaviour. The other interesting feature occurs at coordinate 10 where there appears to be two distinct favoured values.

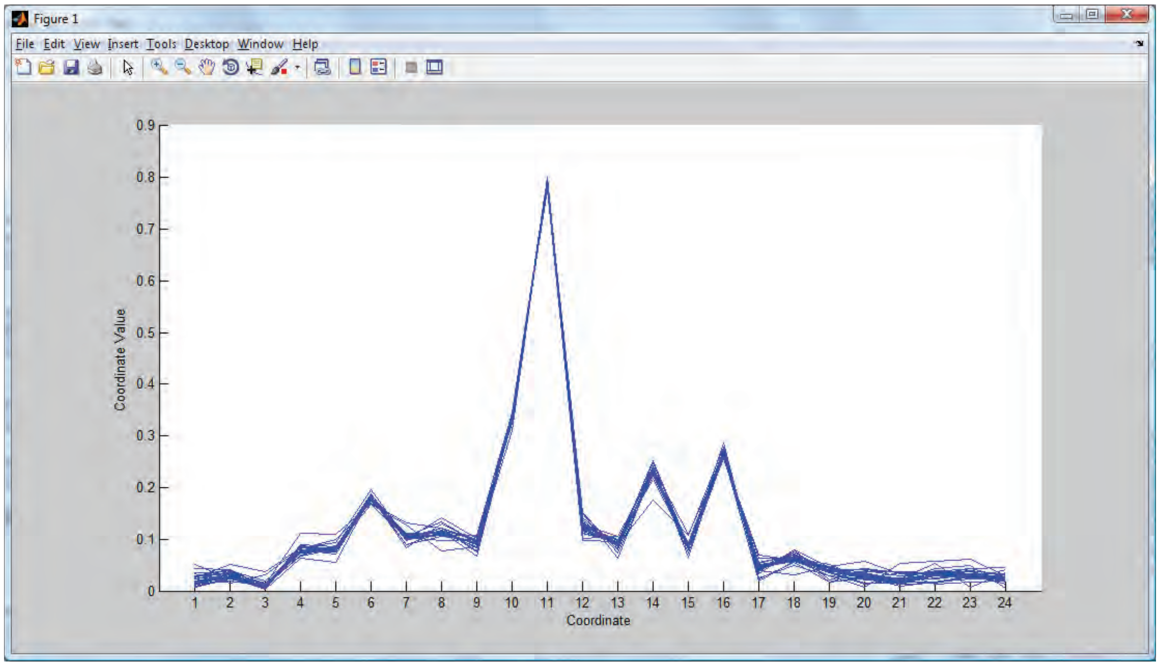


Figure 22: The 24 IP coefficient dimensions for 24 pulses from Emitter 85

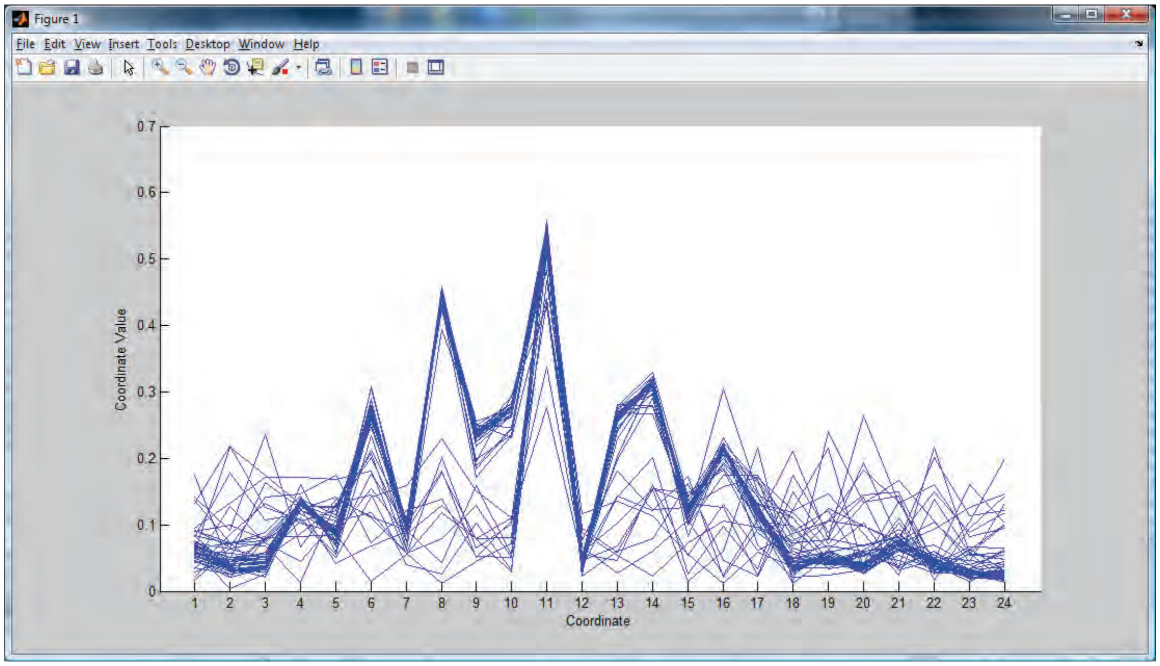


Figure 23: The 24 IP coefficient dimensions for 48 pulses from Emitter 22

The use of a parallel coordinates display allowed us to very rapidly assess the general nature of the pattern of IP coefficient values for Emitter 22, and to note a few particular coordinates that warrant further investigation. If, instead of a parallel coordinates view, we had chosen to use 3D projection plots of the coefficients it would have taken 2024 graphs to view all 3D-tuples that are possible in a 24D hyper dimensional dataset.

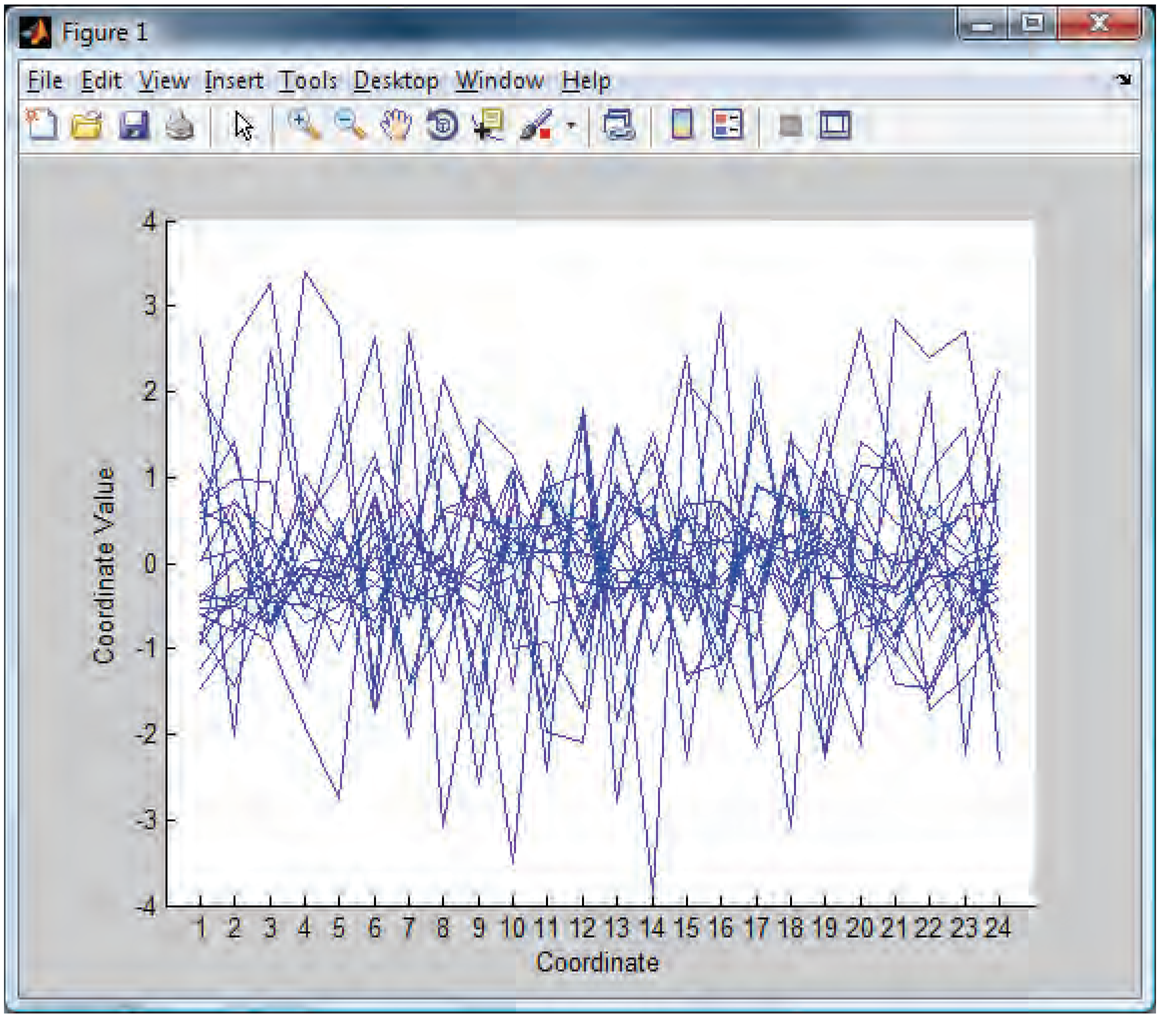


Figure 24: "Standardized" display of Emitter 85 emphasizes noise

As previously mentioned, MATLAB provides functions to scale the coordinates of the parallel coordinates graph to a common Y-axis range. Figure 24 shows the same data as Figure 22 for Emitter 85, except that it was plotted using the MATLAB "*Standardized*" function, which scales all coordinates to a mean of 0 and a standard deviation of 1.0. Since this dataset contains all pulses from a single emitter and the samples are very similar in coefficient values, the process of forcing a standard deviation of 1.0 causes the graph to expand to where it looks like random behaviour.

All parallel coordinate graphs for IP coefficients in this section are displayed in their base range of 0 to 1.0, without any scaling. If we were to include other parameters, such as RF or PW, in the same parallel coordinate graph we would need to scale them into the same 0 to 1 range.

In Figure 25 we can see multiple samples for three distinct emitter families all plotted together on one parallel coordinates graph. In this figure each sample is colour coded to the true source emitter; either Emitter 22, 72 or 85. The patterns for Emitters 22 and 85, that we examined previously, can be see in blue and red. The new Emitter 72 is shown in green and it clearly has its own distinct pattern that exhibits moderate variation from pulse to pulse.

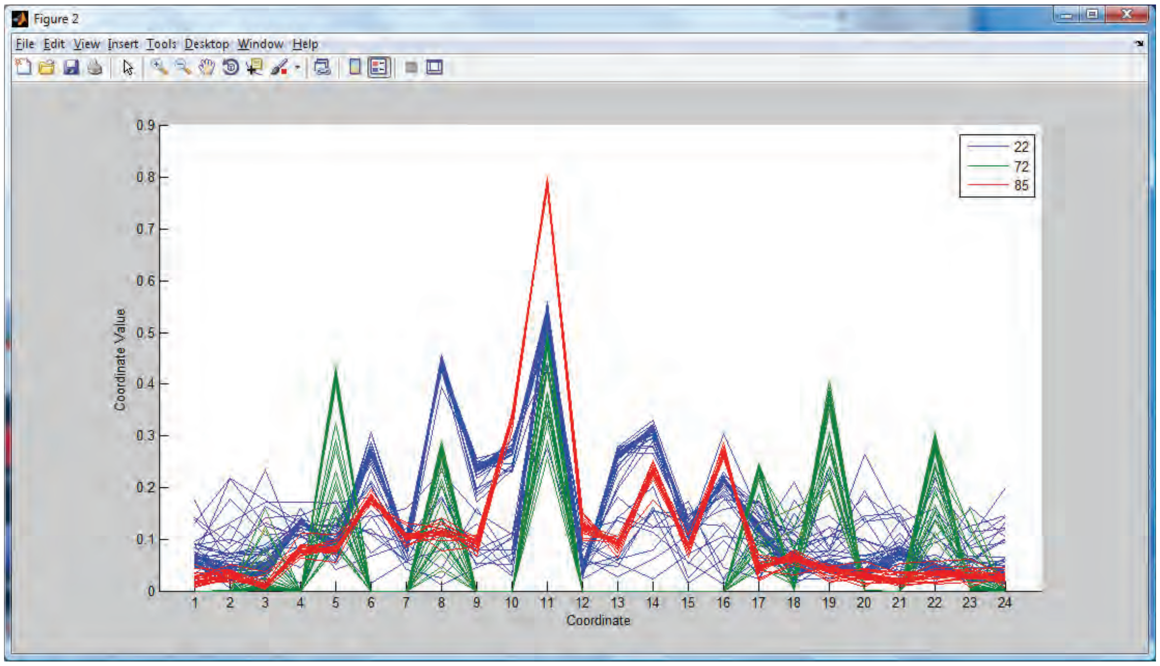


Figure 25: The 24 IP coefficient dimensions for three simulated radar classes

Figure 25 uses ground truth knowledge of the source emitter to colour the samples, however, this graph could also be used to present the results of automated pulse de-interleaving by IP correlation. The de-interleaving process would colour the samples by the pulse train they were sorted into and the graph would allow an analyst to monitor the process.

Unlike the colour coded image in Figure 25, the ELINT analyst will often not have the a priori information to associate the individual pulses with the associated emitter; in those cases the analyst may have to form the associates from the data. Figure 26 depicts a first prototype parallel coordinates browser which could be developed as a tool for ELINT analysis of high dimensional data, and IP analysis in particular. The browser incorporates a set of band pass filters, one for each dimensional coordinate (24 in this case). Each filter represents a pass band consisting of a high and a low value for that coordinate. Any lines lying outside the defined range can be visually subdued or visually suppressed.

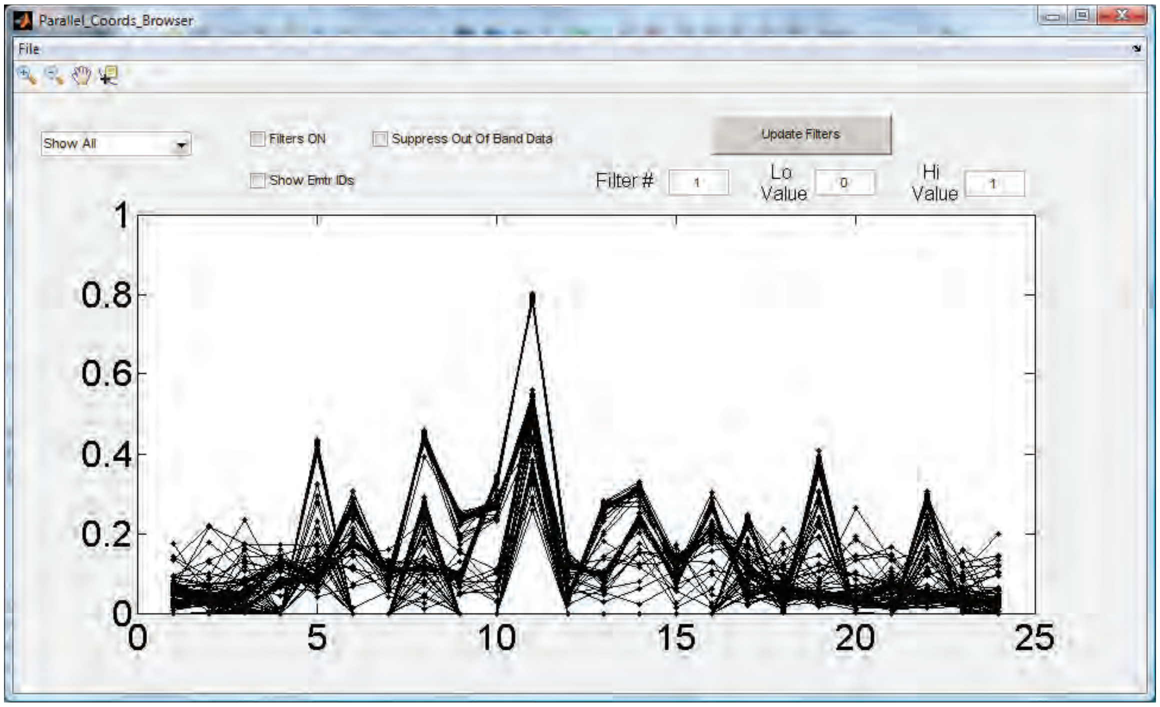
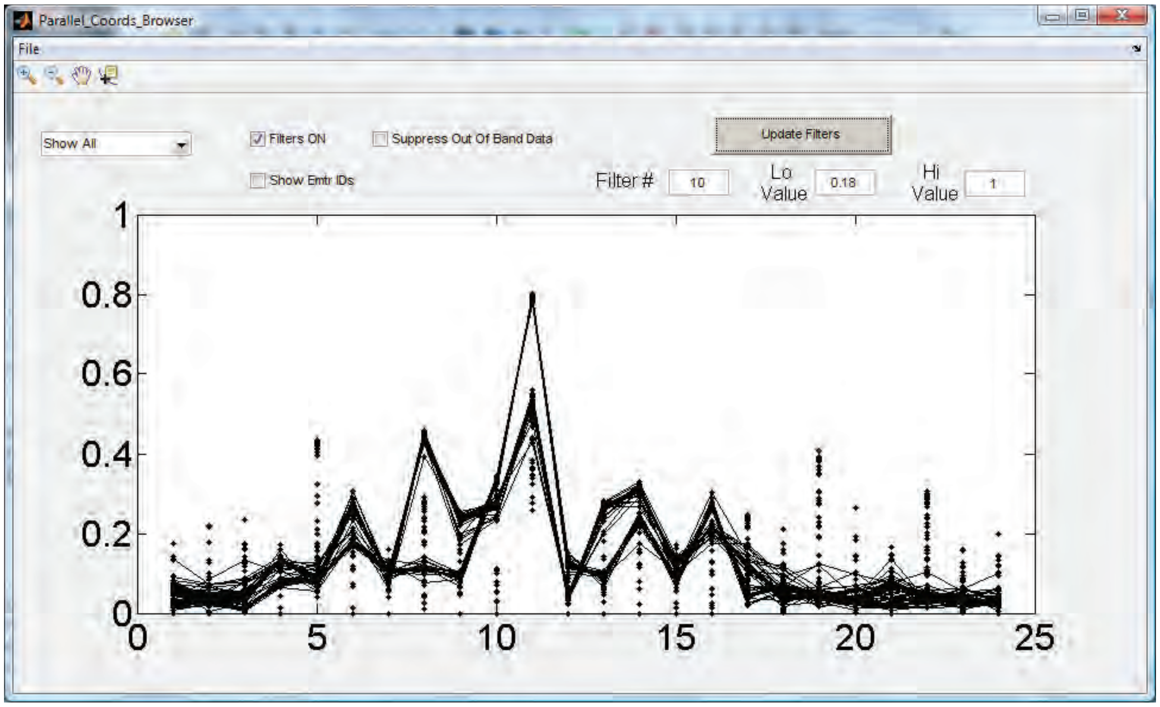


Figure 26: First prototype Parallel Coordinates Browser

Looking at the data in Figure 26, it can be seen that coordinate 10 appears to have two distinct clusters of data. The upper cluster could be separated by a filter set to pass data above 0.18. So the analyst turns the filters on and enters the filter information. When that has been done we get Figure: 27. All lines for out of band data points have been removed from the display and only the coordinate crossing points remain for reference; however these can also be removed by checking the box "Suppress Out Of Band Data", as shown in Figure 28.

Looking at the remaining data in Figure 28 we can see what appear to be two distinct clusters of values for coordinate 8 which could be easily partitioned. This time we will pass all values below 0.25 for coordinate 8, and the result appears in Figure 29. We now have a well behaved subset of the data with no obvious way to partition it further.

Since this data set actually includes ground truth emitter IDs we can verify our processing success by selecting "Show Emtr IDs" and deselecting "Suppress Out Of Band Data". By only subduing and not suppressing the out of band data we will be able to see how many data points for the family that we selected actually lie outside the band pass values. Figure 30 shows the ground truth values for the dataset.



Filters ON [✓]

Suppress Out Of Band Data [ ]

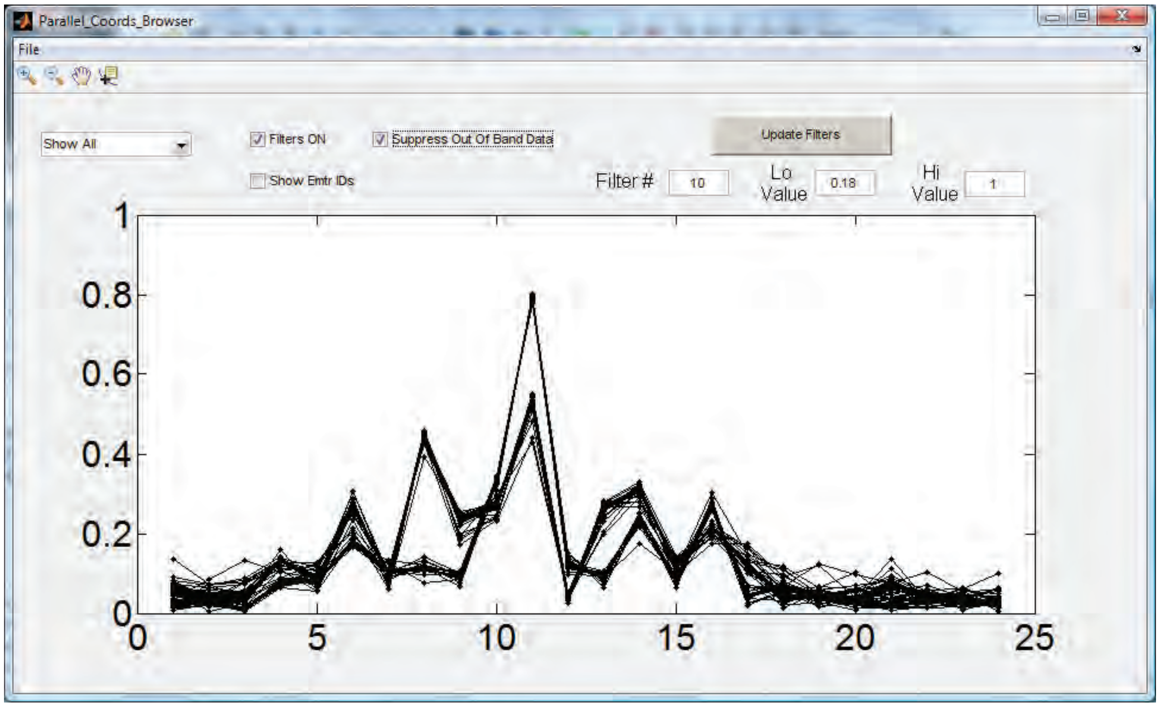
Show Emtr IDs [ ]

Filter # 10

Lo Value 0.18

Hi Value 1

Figure 27: Selecting data with band pass filters



Filters ON [✓]

Suppress Out Of Band Data [✓]

Show Emtr IDs [ ]

Filter # 10

Lo Value 0.18

Hi Value 1

Figure 28: Suppressing data outside of the filter band

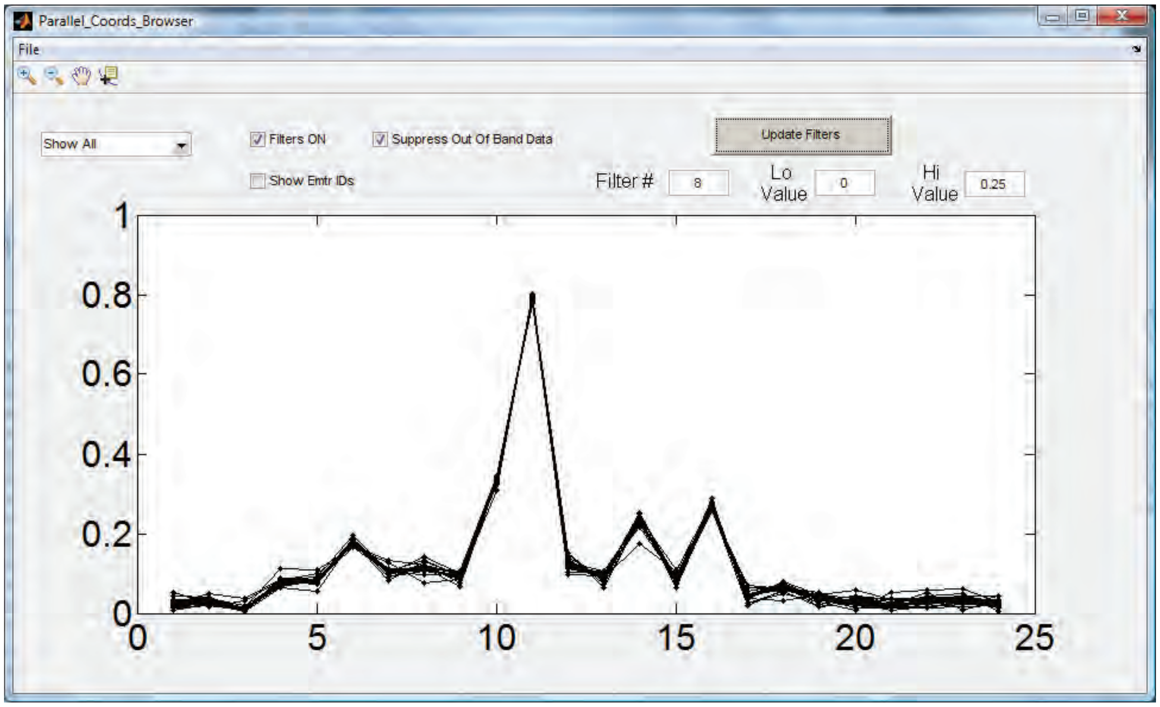


Figure 29: Selected data after two filters applied

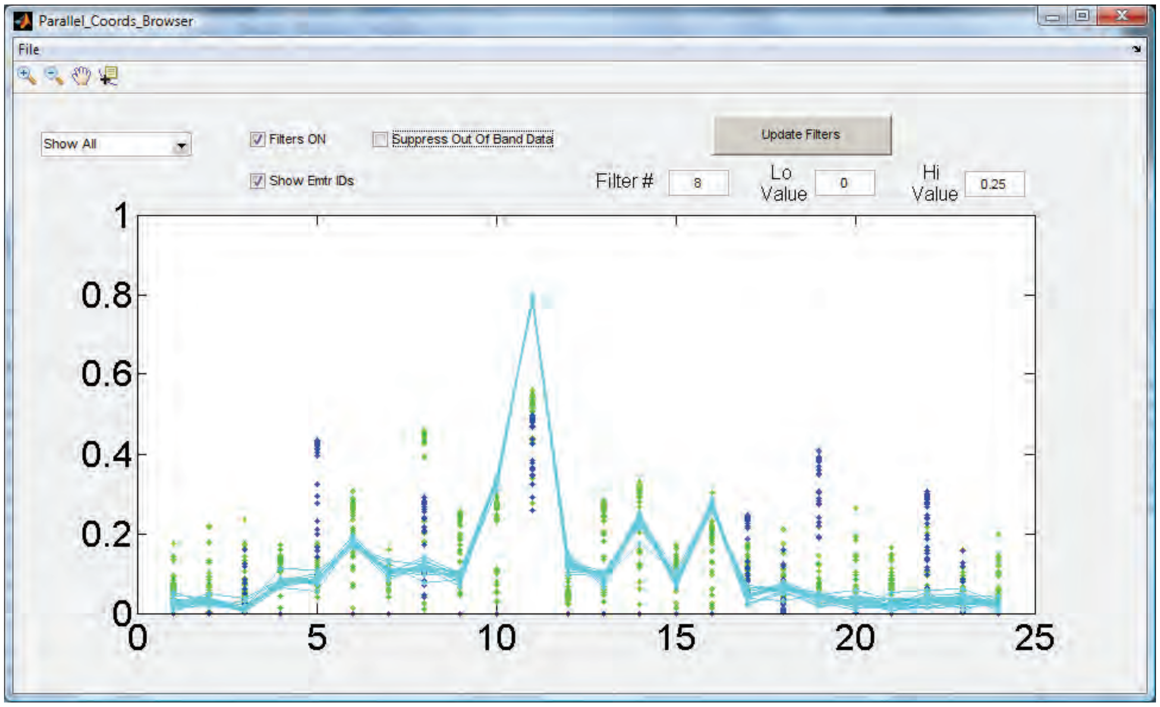


Figure 30: Selected data compared to ground truth ID information

You will note that all of the cyan coloured data points are connected by lines indicating that all are within the band base values and none lie outside. So the band pass filter values have captured all the data for one emitter family. Also note that there are no blue or green coloured data points connected by lines (the other two emitter classes in the data set), therefore no data points from a different emitter made it through the band pass filter even though only two coordinates were used for filtering.

Most de-interleaving or clustering problems will not be this easy and clean, however, the power of this approach is obvious even if it only identifies the core data members of each distinct family or cluster. Combining this technique for generating core clusters, with a nearest neighbour algorithm to collect the outliers, may prove to be very effective.

The very simple prototype Parallel Coordinates Browser used for this analysis could be significantly improved by the addition of a few more control features. First, the native zoom feature provided by MATLAB always zooms in both X and Y together. For the Parallel Coordinates Browser it would be more effective if the zoom feature could be restricted to zooming only the horizontal axis to allow the operator to focus in on a subset of the dimensions. A related second enhancement would allow the operator to change the order of presentation of the dimensions along the X axis, as this would facilitate grouping the dimensions of interest together for the analysis.

7.3 Parallel Coordinates View for Navigating Hyper Dimensional Datasets

The other application of parallel coordinates graphs that we are interested in is the possible use as a navigation control for exploring hyper dimensional datasets. Since the parallel coordinates view simultaneously displays the entire hyper dimensional data space, it can easily show the current observer location for any 3D projection of the space. Using the MATLAB model the 3D projection view has a 3D location for the observer (the camera) and a 3D location for the viewing point (the target). Both of these points could be displayed on the appropriate coordinate axis of the parallel coordinates view. The observer position or the view point could be shifted by simply dragging the handles marking the position to a new coordinate value. Or traditional controls to pan, orbit, dolly in or out, could be employed with the parallel coordinates view depicting the resulting new location.

A 24D IP dataset has 2024 ways to choose a 3D space for viewing, and a not uncommon 64D IP dataset would have a staggering 41,664 ways to choose a 3D space to view. Clearly we need a technique to focus our exploration of this vast search space. How can we exploit the parallel coordinates view to direct our search?

Recall that for the clustering analysis we focused on coordinates with areas where it appeared the data could be partitioned by the band pass filter. Given that these coordinates are exhibiting some recognizable structure where there is separation in the clusters then we should select our 3D coordinate tuple from the coordinates that exhibit the most structure or separation.

Re-examining Figure 26, we note significant structure in coordinates 9, 11, 13, and 19. Figure 31 shows a 3D scatter plot of coordinates 9, 11, and 13. Figure 32 shows a 3D scatter plot of coordinates 11, 13, and 19. Both 3D scatter plots show nominally distinct and separable clusters of data.

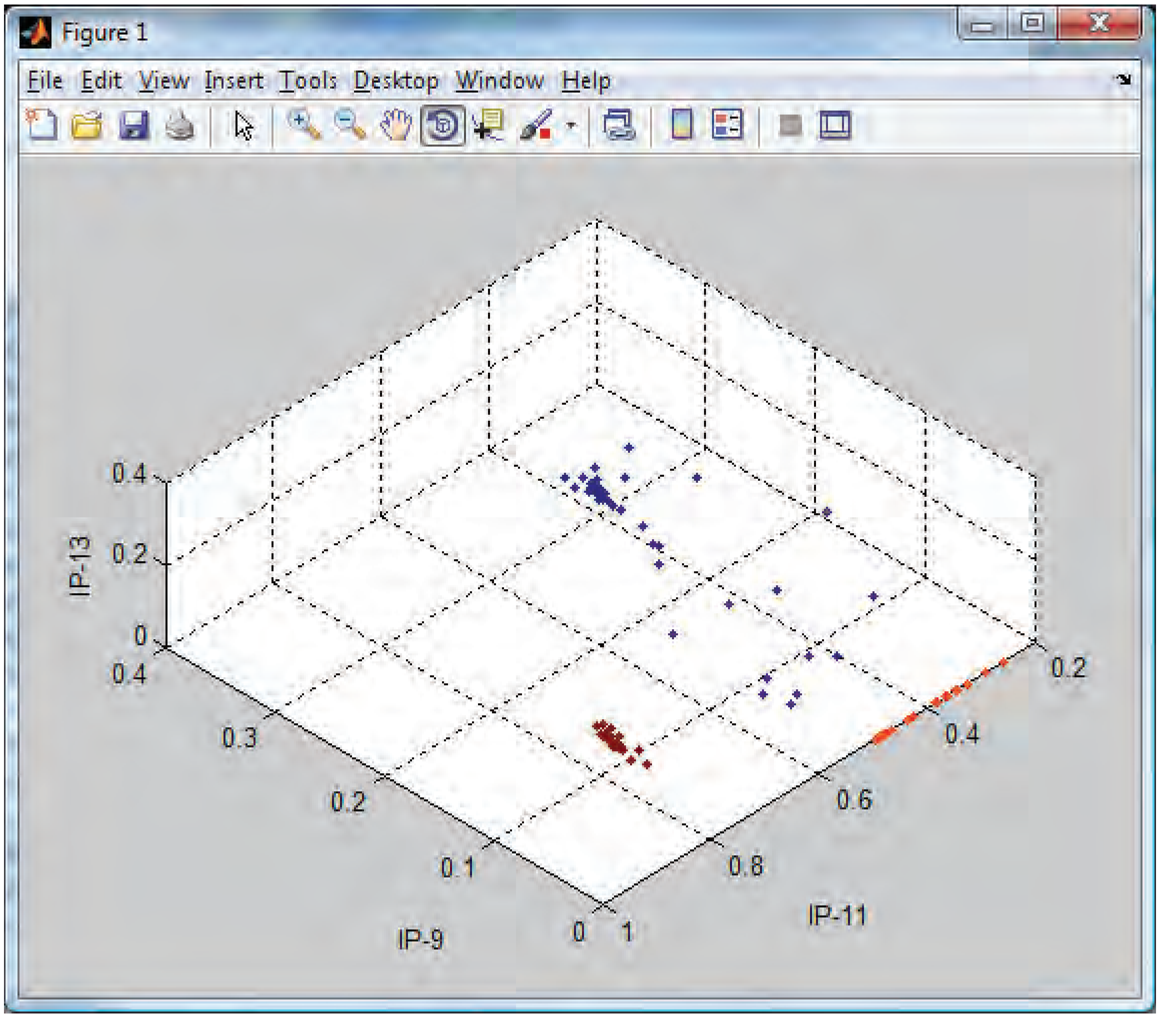


Figure 31: Scatter plot of coordinates 9, 11, and 13

How can we tell if these views are actually good views of the dataset and how do we know that choosing some other combination would not be as good or better? If this approach is actually guiding us to profitable views then selecting 3D views from the parallel coordinate axes with the least structure should produce poorly clustered views.

Examining Figure 26 once again, we can see that coordinates 3, 19, 21, 23, and 24 (among others) have less distinct structure and little opportunity for partitioning the dataset. Figure 33 contains a 3D scatter plot for coordinates 3, 23, and 24. It clearly has less distinct, more diffuse clusters with a higher degree of overlap. Similarly, Figure 34 contains a 3D scatter plot for coordinates 19, 21, and 23; it also exhibits very diffuse and overlapping clusters.

It appears that the parallel coordinates view can provide good visual clues to focus our search in the hyper dimensional data space. Even more benefit can be obtained if we integrate the graphs into a common toolset. If the parallel coordinates view is tied to one or more 3D scatter plots such that a selection in any view is displayed in all views, then we would have a much more powerful analysis tool.

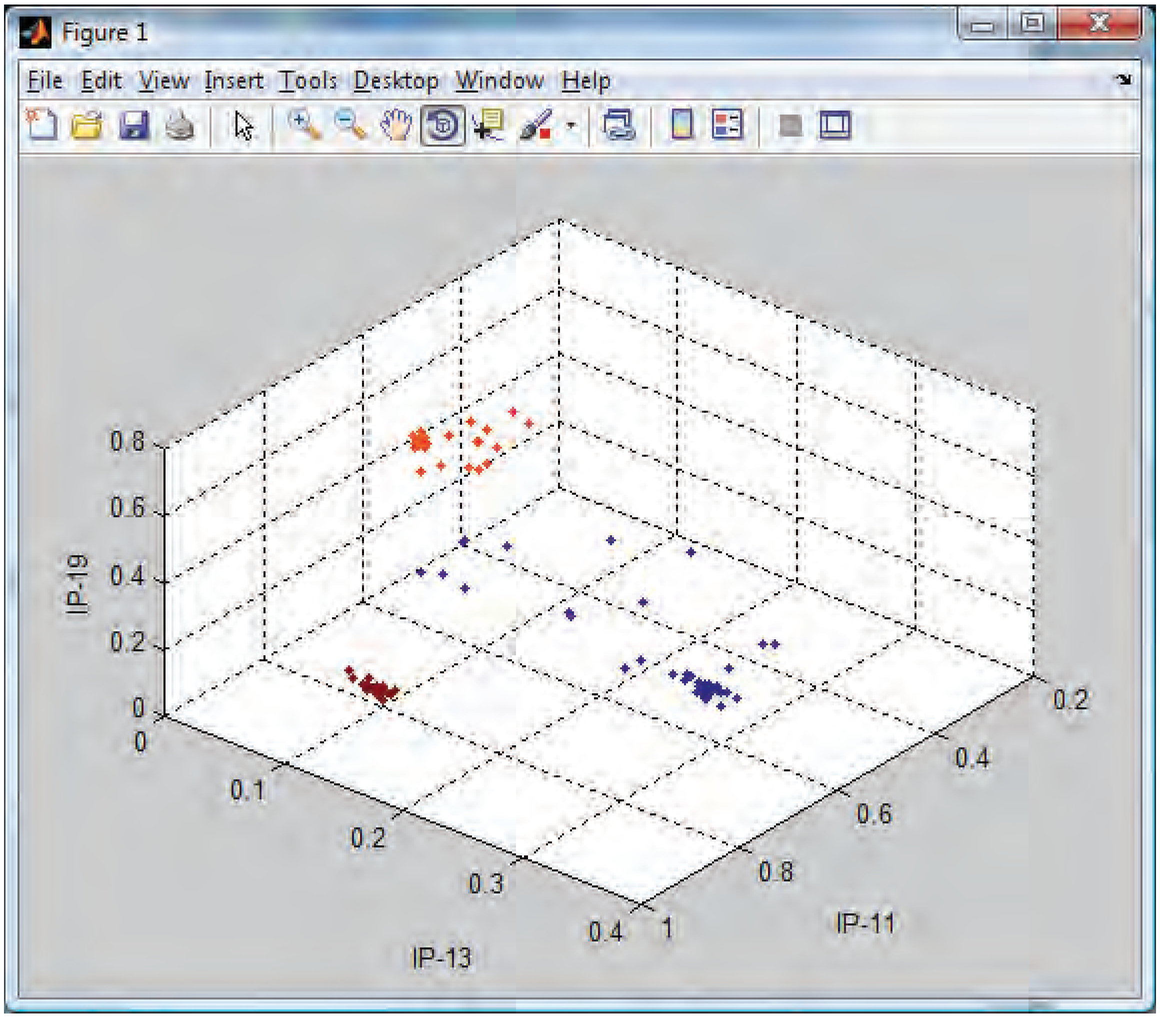


Figure 32: Scatter plot of coordinates 11, 13, and 19

For example, selecting a single line in the parallel coordinates view will select a single point in each 3D scatter plot. However, selecting a 3D point in one scatter view will select a set of lines in the parallel coordinates view, since only three of the coordinates are fixed and all other coordinates are free to take on any value. Then there is the question of how should the selection of a point on one 3D scatter plot be reflected in another 3D scatter plot that may share 2, 1, or no coordinates in common? One approach would be to use the parallel coordinates view as the anchor for all views. Selecting a point in any 3D scatter plot would select a set of lines in the parallel coordinates view that would then be projected as one or more points in each 3D scatter plot views. In this way all views are coordinated.

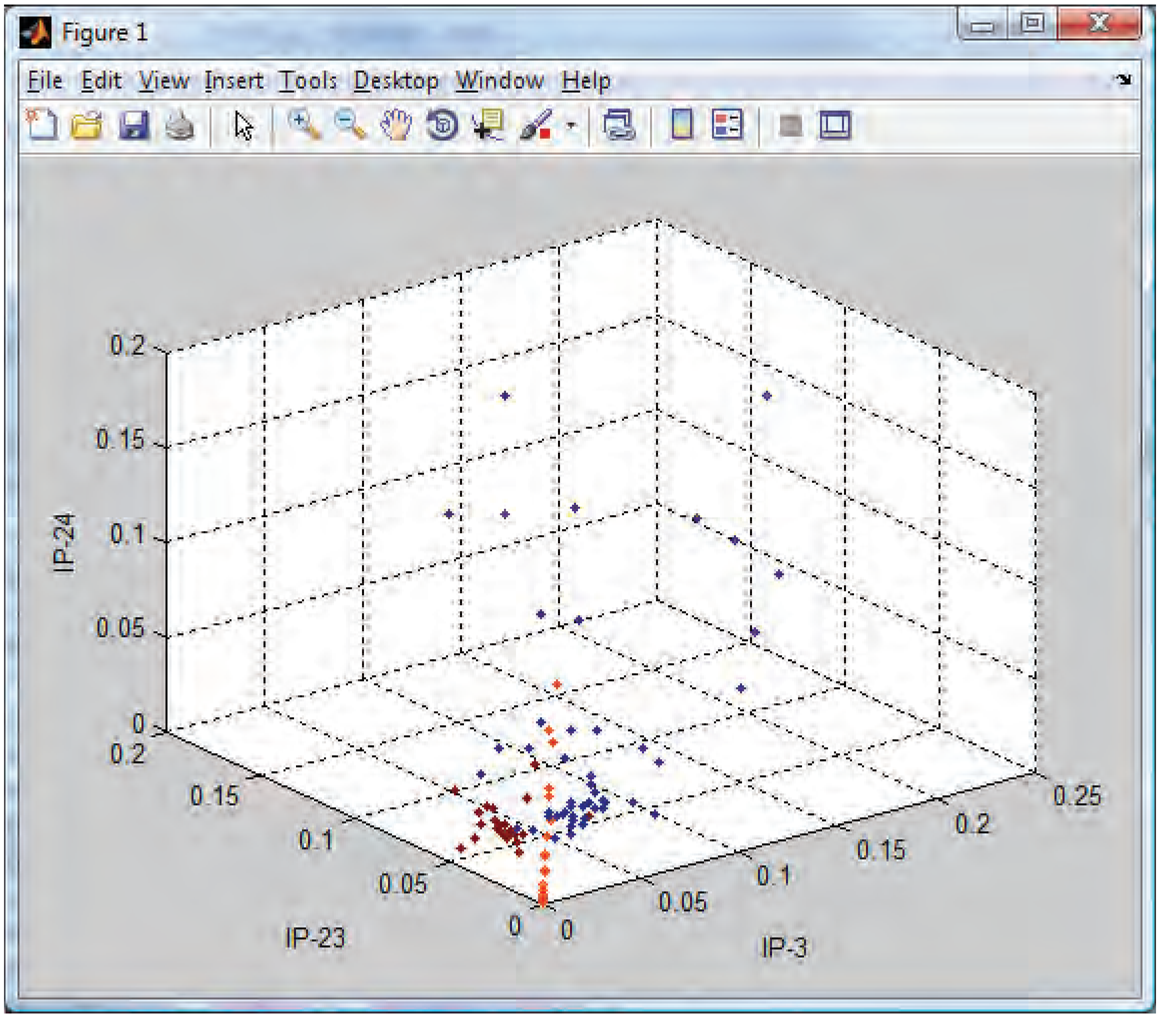


Figure 33: Scatter plot of coordinates 3, 23, and 24

Parallel Coordinates views can also help with the difficult task of selecting a 3D volume from a 3D projection view, where the cursor is restricted to operating in the 2D screen coordinates. One approach would be to use the cursor to select a single point to act as the prototype for the volume selection. The selected point would appear on the parallel coordinates view as a set of lines constrained to pass through 3 coordinate values. In the parallel coordinates view the operator would widen the pass band for each of the three coordinates while watching the projected selection effect on the 3D view. Each of the 3 dimensions could be easily manipulated without any ambiguity caused by mapping a 2D cursor onto a 3D projection view.

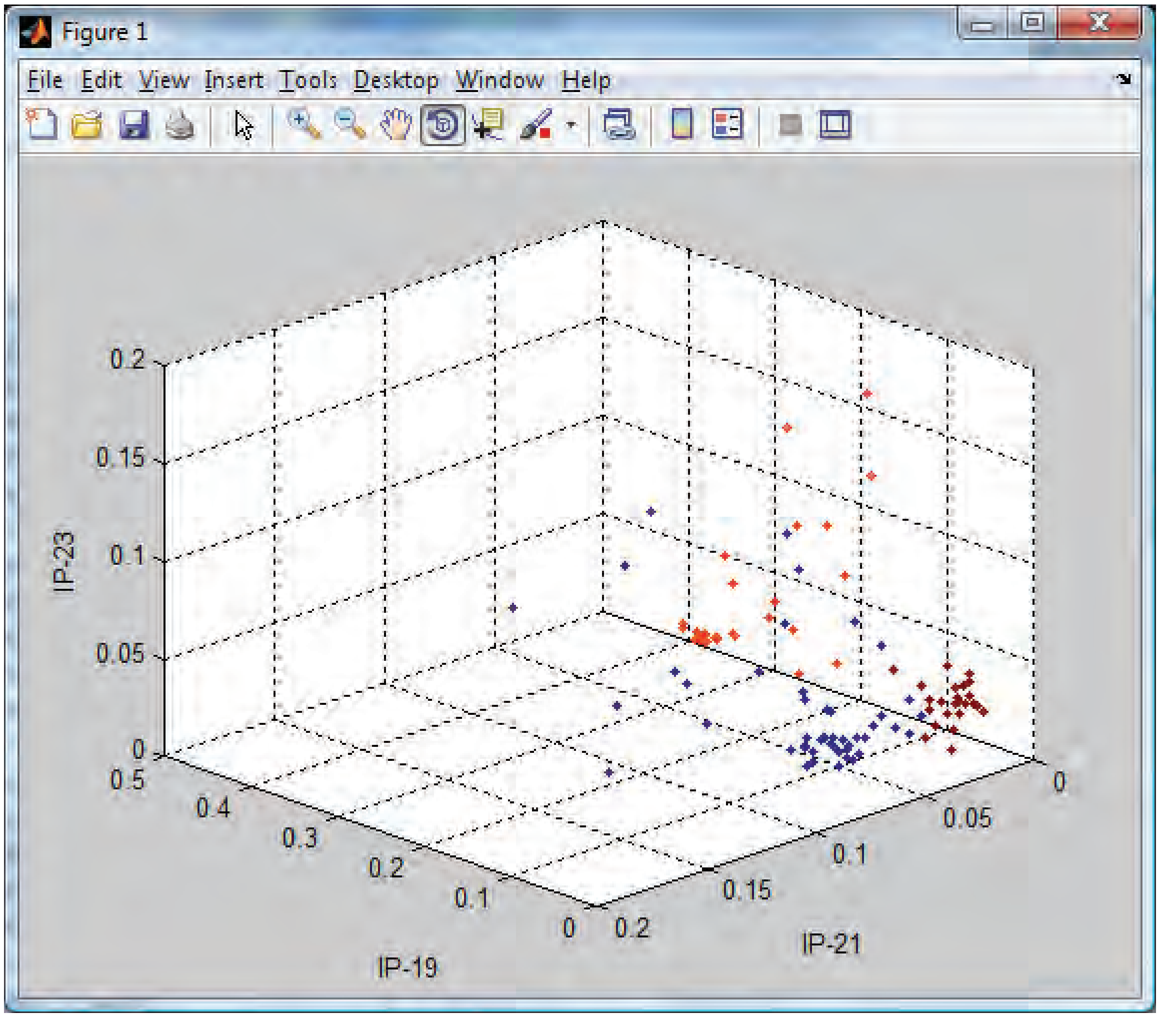


Figure 34: Scatter plot of coordinates 19, 21, and 23

7.4 Glyph Plots for Viewing 4 to 12 Dimensions

Another technique for viewing data of moderately high dimensionality is a glyph plot that forms a star shaped plot with typically 4 to 12 equally distributed axes. A glyph plot is like a polar plot version of a parallel coordinates plot. In a glyph plot all of the axes are arranged radially around a common origin, and, similar to a parallel coordinates plot, a line represents a point by connecting each of the coordinate values. In the case of the glyph plot, the coordinate axes are normally truncated at the coordinate value to create a closed figure, and comparison analysis basically becomes image recognition.

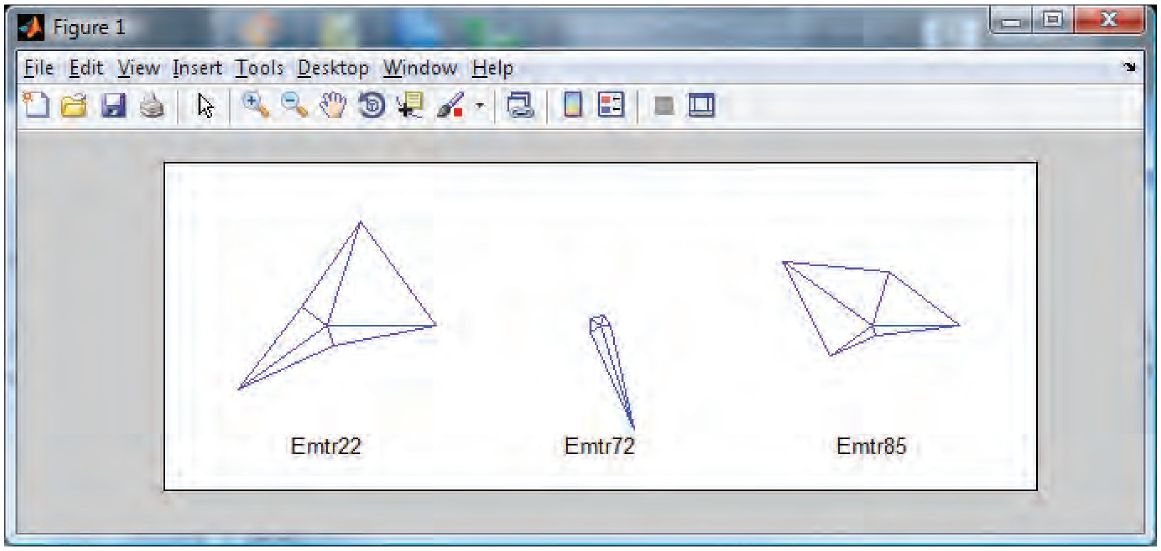


Figure 35: Average coordinate values for the three emitter types

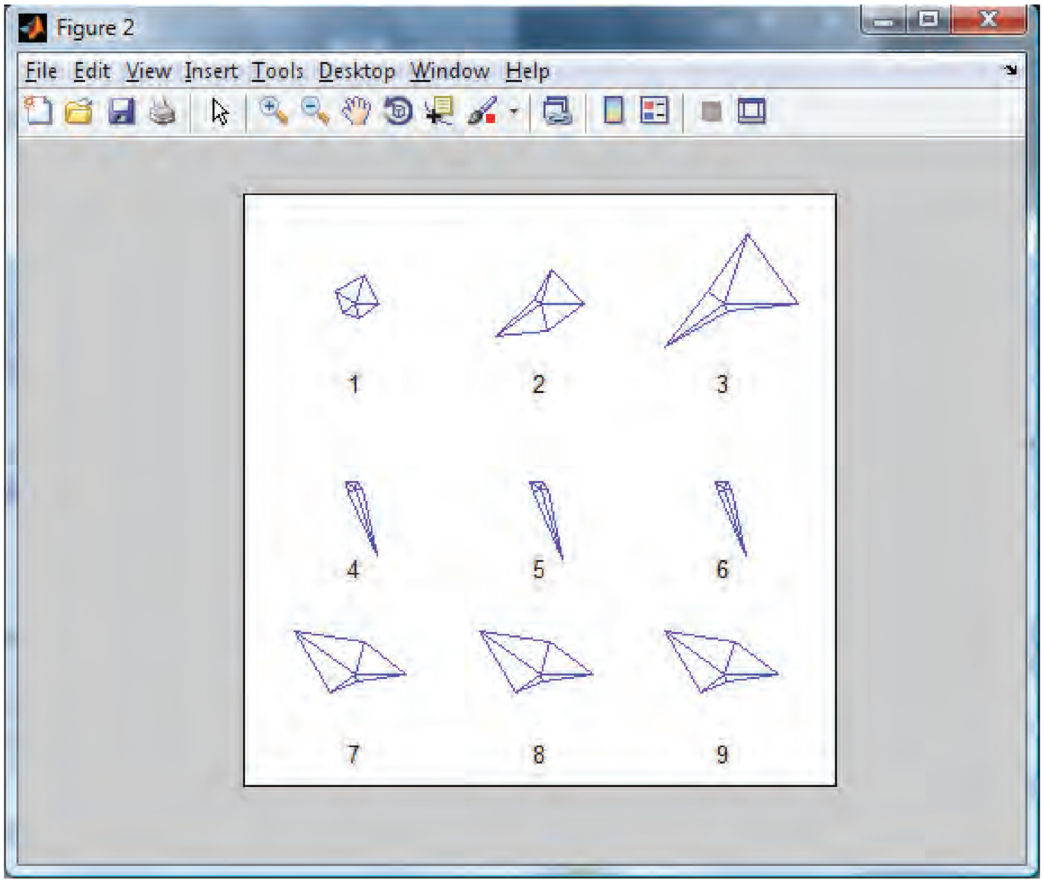


Figure 36: Individual samples from each emitter type

From Figure 26 five coordinates, 6, 9, 11, 13, and 19, with strong structural features were selected for a glyph plot. For each emitter type, the sample values for each of these coordinates were averaged and then plotted as a glyph in Figure 35. It is clear that the glyphs for each emitter family appears to be a distinct shape; however, these are average values and individual samples can exhibit significant variation from the average values. Figure 36 shows three samples from each of the three emitter types; samples 1-3 are Emitter 22, samples 4-6 are Emitter 72, and samples 7-9 are Emitter 85. Samples 3 through 9 are quickly matched to the associated emitter family glyph by visual inspection; however, samples 1 and 2 are not obvious. Sample 1, in particular, would be very difficult to classify as it is significantly different and could be either Emitter 22 or 85.

Glyph star plots would have limited usefulness for hyper dimensional datasets with more than about twelve dimensions. Glyphs appear to be most effective as a presentation technique if the glyph is simple enough to allow our natural image and facial recognition abilities to compare the glyphs. Once the complexity rises to the point where we have to make the conscious and deliberate effort to resolve which axis the spike or notch occurs on, then we loose the natural image recognition that glyphs are designed to exploit. Four to eight dimensions appears to be the optimum range for glyph plots

7.5 Projection onto 2 or 3 Dimensions

Visualizing hyper dimensional data in a two or three dimensional image can be accomplished in a number of ways. Some techniques provide only limited views of the data but preserve the dimensional relationships of the portion that is visible. For example, we can select two or three of the N dimensions and display the data as a two or three dimensional image, with the remaining dimensions suppressed. This approach sacrifices the big picture view for a clear and familiar view of a limited portion of the information. A bigger picture view can be obtained by assembling multiple 2D or 3D views formed from different sets of dimensions. A commonplace example of this technique is the standard top, front, and side views typically used in a drafting drawing. This technique can be easily extended to hyper dimensional data. It would require N(N-1)/2 two dimensional views to display all pairs of dimensions for an N-dimensional data set.

A common visualization technique often used for hyper dimensional statistical data is called a matrix of scatter plots. The 2D scatter plots are usually arranged in the rows and columns of a matrix and often a 1D histogram view is added along the diagonal. Sometimes only half of the matrix is populated since each pair of axes appears twice in the matrix, once above the diagonal and once below the diagonal with the order of the dimensions reversed (i.e., XY and YX).

Figure 37 shows an example of a five dimensional matrix of scatter plots. This set of plots was generated from the same 5D IP pulse data used to produce the glyph plots in the previous section. It contains 107 pulse samples of 5 coefficients from three different emitter types: Emitters 22, 72, and 85. Apart from the strong linear behaviour exhibited by a subset of the data, there are not too many obvious features in the scatter plots.

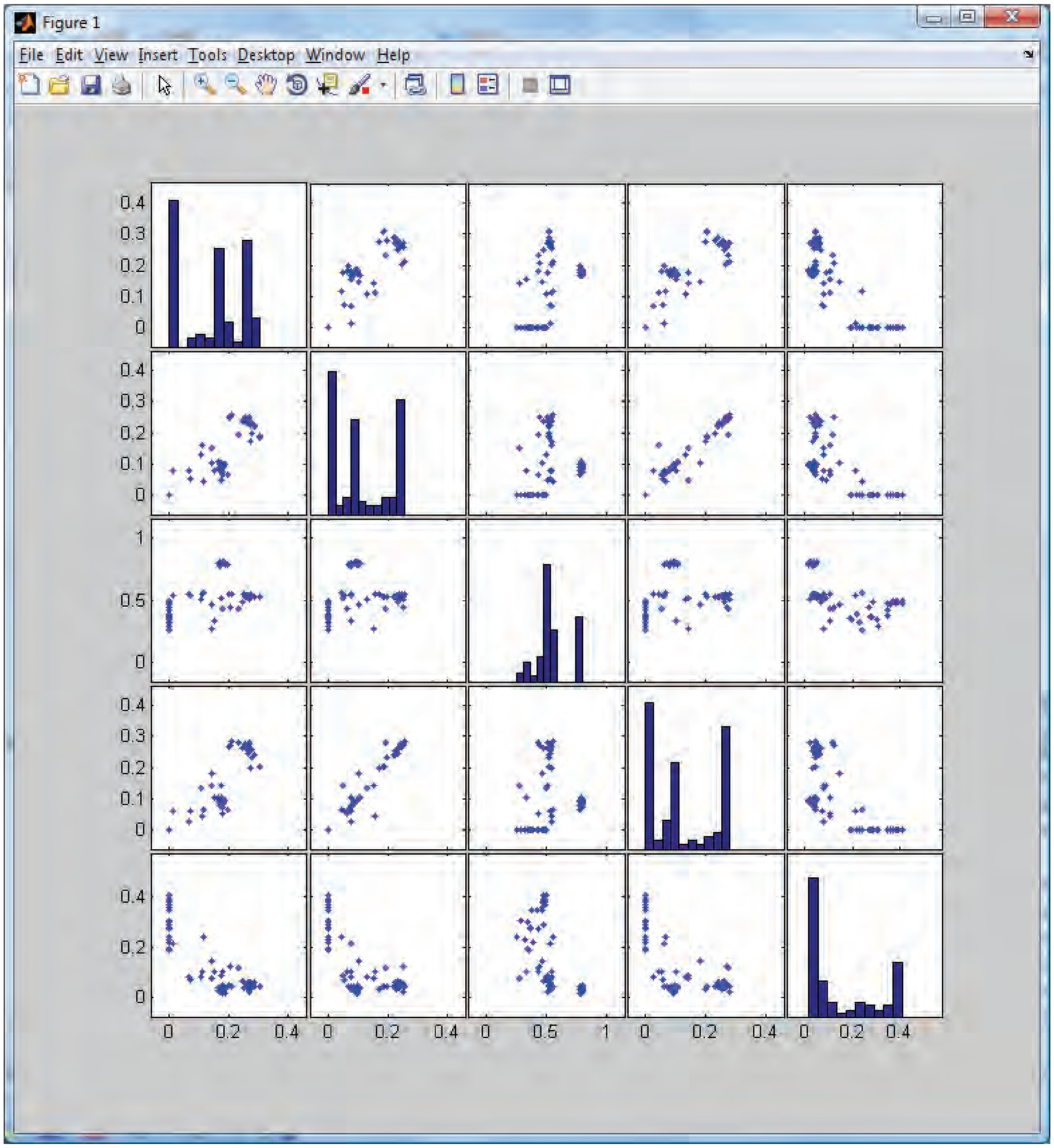


Figure 37: A 5D Matrix of Scatter Plots for three Emitters

Another very innovative technique to present multiple 2D views was devised by D. Asimov. He assembled multiple two dimensional scatter plot views of the hyper dimensional data set into an animation called the Grand Tour [1]. The Grand Tour plays like a movie while the user watches for interesting views of the data. An interesting view is one that provides insight into the relationship between the two dimensions of the displayed plane. It should be understood that these two dimensions are, in general, compositions of all of the dimensions, with the only restriction being that the two chosen dimensions be orthogonal in the 2D plane of the view. In this respect, the selected dimensions are similar to two components of a Principal Component

Analysis in that they are a remapped coordinate system that highlights some structure in the data set.

A Grand Tour provides a very extensive set of views on the data set from nearly every possible perspective; however, the tour can be very long and contain a large number of uninteresting views. A modified tour known as a Projection Pursuit was devised to focus on only the views that are likely interesting. A measure called the pursuit index is defined and views that maximize or minimise the index are collected. An index is usually defined to select views that are maximally different from a normal distribution. The Projection Pursuit creates a collection of snapshot views whereas the Grand Tour is a continuous series of incremental views. With the Grand Tour, the user can maintain orientation and context since the views flow from one to the other, however, the tour can be exceedingly long. With Projection Pursuit, only interesting views are presented, however, the discontinuous jumps between views makes it very difficult for the user to remain oriented.

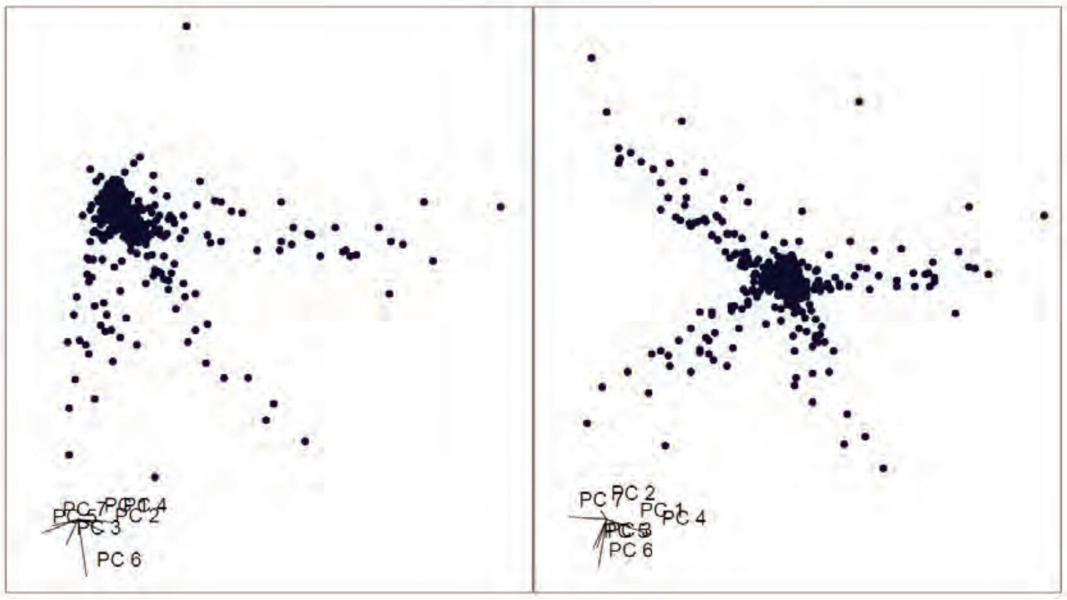


Figure 38: Automated versus Manual Data Exploration [22]

Often the high dimensionality or sheer size of the data set seems to suggest automated search is the only feasible approach; however, Figure 38, above, shows that simply optimising a numeric search metric does not guarantee the most interesting or informative view of the data. In this illustration, particle physics data is represented by principal components in seven dimensions. The left hand image shows the results of a projection pursuit algorithm that maximized the central mass metric at 0.638 to produce the view with three interesting structures visible. However, a small manual change to the principal component contributions produced a new perspective that revealed a fourth arm to the structure. This manual view has a slightly sub-optimal central mass of 0.610 and would not have been found by a strict mechanical search based on maximising the

[22] 7 dimensional particle physics data. Fisherkeller, Friedman, Tukey; 1974, Stanford Linear Accelerator Center, Stanford, CA; Tech.Report SLAC-PUB-1408

search metric. This is a convincing demonstration that human visual analysis still has an important part to play in data analysis, even in cases involving hyper dimensional data.

As part of this project, MATLAB was used to examine a modified hyper dimensional *Tour*, loosely based on the concepts of the original Grand Tour. The modified Tour used a 3D projection scatter plot that was rotated about the target point which is located in the data field. The MATLAB camera position (observer location) was orbited about one or more axes to produce the observed graph rotation. As the camera position moves to a point along one of the axes, the displayed view collapses to a 2D view involving the other two coordinates. Since the data point values for the hidden coordinate have no effect on the displayed position of the point, we can take advantage of this situation to swap the coordinate dimension that is being displayed on the hidden axis. Then as the camera moves off the axis we have a new 3D parameter space that differs from the previous space by one coordinate dimension.

For example, coordinate dimensions 1, 2, and 3 are displayed as the X, Y, and Z axes of a 3D projection plot. The camera position orbits the center of the dataset until the camera is located along the Z axis. At that point in the tour, coordinate dimension 4 replaces 3 on the Z axis and the camera then continues to orbit. As the camera reaches the Y axis, coordinate dimension 5 replaces 2 on the Y axis and the camera moves on. When the camera is on the X axis, coordinate dimension 6 replaces 1 on the X axis and the tour continues. Each time the camera aligns with an axis the hidden coordinate dimension is replaced with another one.

A simple MATLAB function was written to perform the modified Tour as described above and the code is included in Appendix A.12. The modified Tour was used to view simulated radar ESM pulse data that included intra-pulse coefficients for the pulses and therefore represented hyper dimensional data.

The modified Tour approach was considered for adaptation to true stereoscopic 3D, however, the axis swapping technique is not compatible with 3D depth perception. The axis swapping can occur in the 3D projection because there is no depth information when the camera is on one of the axes. While the camera is located along an axis, the data values for that coordinate have no effect on the displayed position of the data point. As long as that is true, the coordinate dimension can be changed and the new data values will not affect the displayed position of the data point. If, however, we introduce stereoscopic 3D, then the foreground/background depth of the data point would suddenly jump if we tried to swap the coordinate dimension for a different one. Instead of a seamless transition to the new coordinate dimension there would be a sudden pop across the entire data field as it jumped to the new depth relationship. Consequently, a 3D version of the modified Tour was not possible.

It should be noted that if the data displayed in the modified Tour has colour coded points and if the 3D projection plot preserves proper point occlusion based on depth order with respect to the observer position, then there can still be a visible glitch in the image when the axis dimension is swapped.

The software for the modified Tour included with this report uses a very simple fixed trajectory for the camera as it moves from one axis to another to step through all the dimensions of the dataset. A serious application of the Touring technique should consider improving the camera trajectory by adding user controls and a more adaptive trajectory. Forward, pause, and rewind controls would be an obvious first enhancement. It would also be advantageous to be able to define key views that should form part of the Tour.

For example, a parallel coordinates view could be used to select points along each coordinate that should be in the Tour given the evident structure at that location. The tour could compute a set of 3D target points made up from combinations of the selected coordinate values and then produce a tour trajectory that passed through all the target points. The Observer position could be set to orbit the target point with the plane of the orbit normal to the trajectory of the tour so that it spirals along the tour path. This tour would be substantially shorter than a formal Grand Tour that is computed to visit all 3D views of the data, but which suffers from extended sections of the tour having no interesting features.

8 Possible Applications

8.1 Problem Domains

Within the context of radar ESM and ELINT analysis there are several problem domains, each with their own distinct characteristics and analysis approaches. Four of these problem domains have been identified as targets for application of data visualization techniques:

- Detection
- Clustering
- Correlation
- Fusion

Signal detection in noise is a problem domain that will forever be an area of research and development. Systems will be pushed to detect signals with lower and lower signal to noise ratios to increase intercept range or to counter stealth attempts to bury the signal in the background noise. Humans have an uncanny ability to detect patterns in noisy environments, often far exceeding the abilities of automated processes.

The clustering or association of data values from ESM and ELINT intercepts is essential to the production of meaningful statistical or deterministic behavioural descriptions for the signal sources. There are many well documented techniques to cluster data values and many of these are applicable to hyper dimensional data, either directly or after dimensional reduction. Although these techniques can operate automatically, there can be situations when the clustering needs to be verified and perhaps modified by an analyst, for example, during ELINT technical analysis of a new signal. Human analysis may be particularly valuable when clusters are partially overlapped in some portion of the data space. Human visual pattern recognition is generally superior to most automated processes and the operator may be able to resolve ambiguous cluster membership based on temporal sequence or signal envelope continuity or other features.

Correlation is defined as a causal, complementary, parallel, or reciprocal relationship, especially a structural, functional, or qualitative correspondence between two comparable entities. Correlation is a key component of ESM and ELINT analysis that comes into play whenever we attempt to relate observed signal behaviour to templates for signal behaviour, or when we compare current observations to previous observations.

Fusion can be loosely defined as the combining or union of data from disparate sources. The focus of correlation is relationships based on the shared attributes between two entities, whereas fusion attempts to create a more complete description by combining component parts that may or may not share attributes. Data fusion is more commonly associated with processing the products of ESM and ELINT systems than with the internal analysis.

8.2 Candidate Visualizations

The following subsections will describe areas where 3D or hyper dimensional visualization techniques were investigated for any advantages they could provide to ESM or ELINT analysis.

8.2.1 Detection

A Falling Raster or Waterfall display has a long history in the ESM, ELINT, and sonar communities. A Falling Raster display has traditionally been a 2D display of spectral frequency on the X-axis and sample time on the Y-axis, with the oldest samples at the top of the display. As samples arrive they are written as new lines directly below the last sample. A Falling Raster can also be used to display a cyclic or folded time axis similar to an oscilloscope trace except that each trace appears as a separate line along the X axis with subsequent lines below it. The technique gets its name from the fact that the image appears to fall from the top of the display, assuming the sample rate is slow enough to see the effect.

Some signal analysis systems have added a third dimension to the Falling Raster by displaying the image as a 3D perspective drawing. Normally the third dimension must be very compressed to prevent it from interfering with the presentation of primary raster pattern. The goal for this project was to examine the Falling Raster techniques to determine if a true 3D display would improve data presentation over what can be accomplished with 3D perspective views. However, since no signal detection datasets were available for testing, this area of study was unfortunately seriously limited.

The intent was to use depth in the 3D display to represent the time axis instead of using the Y axis as is normal for a waterfall display. In a standard 2D waterfall display the data values are displayed as the colour or the intensity of the waterfall raster line, whereas in a 3D perspective waterfall display, like the MATLAB version in Figure 39, the data value is also encoded as a Z value in the 2D projection. The true 3D version of the waterfall would encode the data with colour and as a value on the Y axis. In addition it would display the data as points rather than surfaces, to facilitate looking through the current data values at the past data values behind it.

The data used for the MATLAB example in Figure 39 comes from MATLAB and does not represent the type of data expected for the intended EW application. The 3D waterfall display is intended for data that is much peakier with respect to the raster line and yet still repetitive from one raster line to the next, although the position in the next raster may be shifted left or right. Data like the MATLAB data with values spread over broad areas of the raster line are not suitable for point data displays. Implementation and testing of these ideas for a 3D waterfall display were deferred to a future tasking when test data is available.

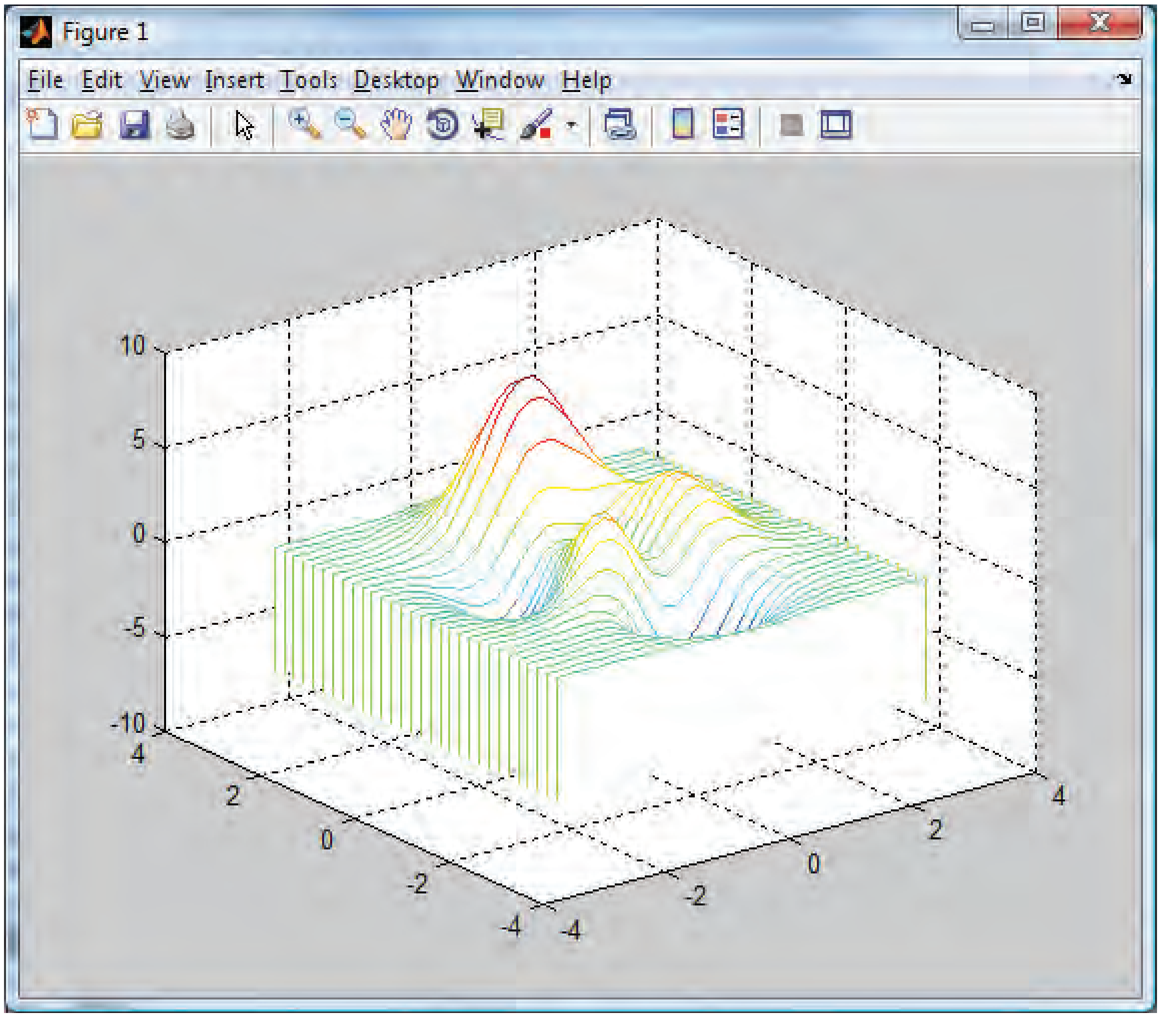


Figure 39: MATLAB waterfall display

8.2.2 Clustering

De-interleaving signal pulse trains is an essential and critical operation for both ELINT and, in particular, wideband ESM intercept processing. The performance of all downstream analysis processes depend on the purity and coherence of the pulse clusters formed by the de-interleaving process. As previously described, pulse descriptor words are typically hyper dimensional and when intra-pulse features are included, the data sets become very high order hyper dimensional.

The 3D imaging and hyper dimensional data visualization techniques reported herein could be applied to pulse de-interleaving clustering. The use of the parallel coordinate browser, or preferably a parallel coordinates browser with integrated 3D scatter plotting, would be an asset to the analyst for either verification of automated clustering, or manual cluster analysis. Including a modified Grand Tour capability into the tool set would increase its flexibility for data exploration and clustering verification.

8.2.3 Correlation

As previously discussed, ESM and ELINT have a requirement to correlate observed signal behaviour to templates for signal behaviour, or to compare current observations to previous observations. This correlation operation is at the core of both radar type identification and Specific Emitter Identification (SEI) and is, therefore, critical to the operational value of ESM/ELINT intercept receivers.

The visualization techniques that could be useful for visual correlation of two observational data sets are likely similar to those used for visualizing the clustering of the data sets. Although the goals and process may be different, the tools may be shared or similar. On the other hand, the correlation of observed data sets to templates of signal behaviour, or the correlation of signal behavioural descriptions to templates of signal behaviour, may require a different set of visualization tools.

The parallel coordinate browser could be used to match simple templates that consist of valid ranges for hyper dimensional datasets. A template that consists of min-max data limits for a set of dimensions could be used to set the pass filters of the parallel coordinate browser with the result that data meeting the template description would be highlighted in all coordinated views of the tool set. Templates having more complex structure would likely require customized tools for verification analysis.

8.2.4 Fusion

Although data fusion is more commonly associated with processing the products of ESM and ELINT systems than with the internal analysis, data visualization techniques were considered for possible applications. An ESM/ELNT problem area has been identified that relates to information fusion or at least fusible attributes.

In general, radar ESM intercept receivers produce ambiguous type identifications for the intercepts. The list of possible type identifications for an intercept may, or may not, include an associated belief value that, if present, allows the list to be ranked. In either case the ambiguous radar type identification leads to additional ambiguity in the inferred platform or installation class for the observed entity.

An intercept with ambiguous radar type identification can lead to an ambiguous situation with respect to whether the entity is hostile, neutral, or friendly. Ambiguity as to the hostility status of the entity can lead to multiple ambiguous interpretations of the tactical situation. One area of automated reasoning that deals with this type of ambiguity is Possible Worlds reasoning, which is a form of modal logic. In Possible Worlds reasoning, ambiguous possible interpretations are grouped into alternate world views where each world contains only interpretations that are logically self consistent. Given any normal situation where there are a number of entities with ambiguous identifications, the number of possible world views can be exceedingly large and difficult to manage.

An intriguing possibility coming out of this investigation of hyper dimensional data is that these visualization techniques may have some application to the Possible Worlds problem. If we treat each ambiguous entity as a dimension and its possible interpretations as the range of values within that dimension, then it might be possible to treat it as a hyper dimensional visualization problem. There would be several difficulties to overcome; one major problem is how to graphically represent or rank non numerical information like radar type identifications. Solutions may come from a different area of hyper dimensional analysis, that being the analysis of subject themes in text documents. Work has been done in that domain to produce similarity metrics for non numeric information, which may have application to our problem domain.

Although a detailed examination of this idea exceeded the scope of this tasking, the approach was explored sufficiently to document the concept. The feasibility of the approach is difficult to estimate at this early stage of investigation.

9 Summary

The technology to support 3D visualization has undergone a massive shift from the high end virtual reality lab to the consumer market, due in large part to the huge market for high end video games. Competition in this market will rapidly drive 3D performance up and cost down. Virtual 3D immersive environments are likely to be available as COTS technology at consumer pricing within this decade. Visual analysis and data mining techniques previously restricted to high end virtual reality labs will soon be supportable on consumer grade computers.

This report describes the investigation of 3D display presentation, 3D representation techniques, and hyper dimensional data presentation techniques that may be applicable to ESM/ELINT analysis. An additional task item was to conduct a survey of virtual reality display software/hardware that is currently available for low cost 3D display systems.

ESM and ELINT signal analysis are inherently complex and involve multi-dimensional analysis. The complexity of the analysis often makes it difficult to represent an interim state of the analysis, or the solution, in the form of text or 2-dimensional graphics. This project investigated the application of 3D graphic presentations to improve user comprehension of interim or final ESM signal analysis results. The investigation considered the following:

- 3D representations presented as 2D graphics;
- true 3D visualization as is currently possible with modest Virtual Reality (VR) systems;
- Hyper Dimensional Visualization techniques.

For static display of 3D data (i.e., printed reports) the two primary technologies are anaglyph and stereo pair stereograms. Anaglyph stereograms are far more common in printed documents than stereo pair stereograms; however, the stereo pair stereogram provides better colour reproduction and superior 3D depth perception. The advantage in favour of the anaglyph stereogram, and the reason for its popularity, is the cost and simplicity of the anaglyph glasses required to view the image. A stereo pair stereogram would normally require prismatic glasses or a viewer that uses mirrors to direct each image to the correct eye.[23] The advantage in favour of the stereo pair is that each image of the pair is a standard 3D perspective view of the data that can be easily interpreted without any viewing devices. For this reason. and for the superior quality of the stereo pair stereogram, this report recommends the use of stereo pair stereograms in printed reports.

For electronic documents and 2D display screens that support animated graphics, motion parallax was found to greatly enhance depth perception from 2D media without external viewing devices. A 3D projection onto a 2D screen can provide significant depth cues to the observer by introducing a small jitter or wiggle in the observer position. This jitter produces strong depth cues by motion parallax and changes to the occlusion of background objects. A short jitter animation can be effectively used for on screen display or embedded in electronic documents as a substitute for a true 3D image. Electronic documents and 2D display screens can obviously also use all of the same techniques that were suggested for static media.

[23] The one exception is the cross-eyed stereo pair which can be viewed directly however the cross-eyed viewing introduces significant eye strain.

Hardware for a simple 3D display system was obtained and tested. The system was based on LCD shutter glasses and a 120Hz LCD monitor. A consumer grade PC with a modest 3D capable GPU and an upgraded power supply were sufficient to support the 3D system. Depth perception with the LCD shutter glasses was very good, with very little eye strain. Good quality 3D display technologies at consumer prices are a very recent development and are currently in a state of flux. The key approaches to watch are Open Graphics Library (OpenGL), Microsoft DirectX, and Java 3D. DirectX currently dominates the 3D game market whereas OpenGL dominates the high end professional workstations. Each has aspirations to dominate both fields in the future. Java 3D is not a competitor, instead it runs atop either OpenGL or Direct3D, and provides an object-oriented abstraction of the API's as a Java class library. Until MATLAB is fully integrated with OpenGL or DirectX, Java 3D may be the easiest route to add 3D display support for shutter glasses, since MATLAB is fully integrated with Java.

The MATLAB software system has graphical support for 3D presentation on 2D media, as well as animations and hyper dimensional visualization. Very effective 3D jitter effects can easily be added to MATLAB 3D graphs to greatly enhance depth perception. MATLAB code to demonstrate these effects can be found in the Annex. MATLAB also supports parallel coordinate graphs for hyper dimensional data and this leads to the development of a basic prototype Parallel Coordinate Browser used to explore radar intra-pulse coefficients. The initial success of the Parallel Coordinate Browser supports continued development of this browser as the key element in a coordinated toolset with linked 3D scatter plots.

Many third party MATLAB functions are available through the file exchange program and several of these have been identified herein as applicable to 3D and hyper dimensional visualization.

References

[1] "The Grand Tour: A Tool for Viewing Multidimensional Data", D. Asimov, DIAM Journal on Scientific and Statistical Computing, vol.61, pp.128-143, 1985.

[2] "High-Dimensional Visualizations"; G. Grinstein, M. Trutschl, U. Cvek; In Proceedings of Workshop on Visual Data Mining, ACM Conference on Knowledge Discovery and Data Mining, pp. 1-14, 2001.

[3] "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations"; B. Shneiderman; Proceedings of the 1996 IEEE Symposium on Visual Languages; pg 336, 1996.

[4] "Stereo and motion parallax cues in human 3D vision: Can they vanish without a trace?"; A.M. Rauschecker, S.G. Solomon, A. Glennerster; Journal of Vision v6, pg1471–1485, 2006.

[5] "Manual Controls for High-Dimensional Data Projection"; D. Cook, A. Buja; Journal of Computational and Graphical Statistics, 6(4), pg464-480, 1997.

[6] "Rolling the Dice: Multidimensional Visual Exploration using Scatterplot Matrix Navigation"; N. Elmqvist, P. Dragicevic, JD. Fekete; In IEEE Transactions on Visualization and Computer Graphics, v14 #6, 2008.

[7] "Interactive Data Exploration with Targeted Projection Pursuit"; J. Faith; Leonardo Electronic Almanac, v16 issue 6-7, 2009.

[8] "Immersive Projection Technology for Visual Data Mining"; E.J. Wegman, J. Symonzik; Journal of Computational and Graphical Statistics, v11 #1, pg1–26, 2002.

[9] "Data Mining and Visualization: Some Strategies"; E.J. Wegman; Bulletin of the International Statistical Institute, Tome LVIII, Book 3, pg223-226, 1999

[10] "Parameterizer and Analysis Software System (PASS)"; Northrop Grumman; <SIGINT software marketing pamphlet>; 2008.

Annex A MATLAB Code Developed

A.1 Function: Camera 3D Pic

This function takes the handle of a 3D perspective graph axes and creates a stereo pair stereograph image by shifting the camera (observer) position and recording the left & right images.

```
function varargout = camera_3d_pic(varargin)
% take 3D side-by-side pic of graph in axes
%
% Inputs: zero or an axes handle
%
% Outputs: struct with 3D image in 'cdata' field
%
%   camera_3d_pic() - 3D pic of the current (gca) axes view
%
%   camera_3d_pic(axes-handle) - 3D pic of the provided axes view
%
%   camera_3d_pic(axes-handle, angle) - 3D pic with binoccular angle
%
% KNOWN BUG: picture may have bleed thru from MATLAB window under figure.
%             move Figure off of all MATLAB windows before executing
%
% example
%   sim_distance_perspective_3d;
%   picStruct = camera_3d_pic(gca, 6);
%   pic3d = picStruct.cdata;
%   image(pic3d);
%   axis off;
%   axis image;
%
% example
%   sim_distance_perspective_3d;
%   picStruct = camera_3d_pic();
%   pic3d = picStruct.cdata;
%   imwrite(pic3d,'c:\test\test-3d-pic.jpg','jpg');

switch nargin
    case 0
        hand = gca;
        angle = 8;
    case 1
        % uses the axes handle passed in
        hand = varargin{1};
        angle = 8;
    case 2
        % uses the axes handle & binoccular angle
        hand = varargin{1};
        angle = varargin{2};
end
```

```
axes(hand);
figHand = getFigureHandle(hand);
figure(figHand); %bring to front
camorbit(hand, (angle/-2),0,'data',[0 0 1])
drawnow
film(1) = getframe(hand);
camorbit(hand, angle,0,'data',[0 0 1])
drawnow
film(2) = getframe(hand);
camorbit(hand, (angle/-2),0,'data',[0 0 1])
drawnow


pic1=film(1).cdata;
pic2=film(2).cdata;
out = make_sterio_pair(pic1, pic2);
out.pic1 = pic1;
out.pic2 = pic2;
if nargout == 1,
    varargout = {out};
end
```

A.2 Function: Camera Jitter 3D

This function takes the handle of a 3D perspective graph axes and will jitter the graph on the screen to produce motion parallax effects to enhance 3D depth perception.

```
function varargout = camera_jitter_3d(varargin)
% Jitter camera position about Z-axis to create motion parallax for 3D
%
% Inputs: nothing or an axes handle
%
% Outputs (optional):  the axes handle
%
%   camera_jitter_3d() - jitter the current (gca) axes view
%
%   camera_jitter_3d(axes-handle) - jitter the provided axes view
%
% Example 1
%   surf(peaks);
%   camera_jitter_3d();
%
% Example 2
%   [x,y,z] = sphere(16);
%   X = [x(:)*.5 x(:)*.75 x(:)];
%   Y = [y(:)*.5 y(:)*.75 y(:)];
%   Z = [z(:)*.5 z(:)*.75 z(:)];
%   S = repmat([1 .75 .5]*10,prod(size(x)),1);
%   C = repmat([1 2 3],prod(size(x)),1);
%   fHdl=figure; hdl=gca;
%   scatter3(hdl, X(:),Y(:),Z(:),S(:),C(:),'filled'), view(-60,60);
%   camera_jitter_3d();

switch nargin
    case 0
        hand = gca;
    case 1
        % uses the axes handle passed in
        hand = varargin{1};
end

figHand = getFigureHandle(hand);
figure(figHand); %bring to front
if figHand ~= hand
    axes(hand);  %if capture is on an axes make it current
end
axis vis3d; % adjust aspect for 3D motion

for i=1:8
    for k=1:8
        camorbit(hand, k/4,0,'data',[0 0 1])
        drawnow
    end
    for k=8:1
        camorbit(hand, k/4,0,'data',[0 0 1])
        drawnow
    end
    for k=1:8
        camorbit(hand, k/-4,0,'data',[0 0 1])
        drawnow
    end
    for k=8:1
        camorbit(hand, k/-4,0,'data',[0 0 1])
        drawnow
    end
end

if nargout == 1,
    varargout = {hand};
end
```

A.3 Function: Camera Jitter 3D Record

This function produces a 2D & 3D jitter animations from the output of the function *camera_record_3d_stripe*.

```
function varargout = camera_jitter_3d_record(varargin)
% Record jitterred camera position to create motion parallax for 3D
% One complete jitter cycle is recorded and can be looped on playback
```

```
%
% Inputs: zero or an axes handle
%
% Outputs (optional):
%    the movie of the jittered axes
%
%   camera_jitter_3d() - jitter the current (gca) axes view
%
%   camera_jitter_3d(hdle) - jitter the provided axes handle
%
%   camera_jitter_3d(hdle,angle) - set the angle of jitter in deg
%
%   camera_jitter_3d(hdle,angle,step) - set the step size in deg
%
% Example:
%   [x,y,z] = sphere(16);
%   X = [x(:)*.5 x(:)*.75 x(:)];
%   Y = [y(:)*.5 y(:)*.75 y(:)];
%   Z = [z(:)*.5 z(:)*.75 z(:)];
%   S = repmat([1 .75 .5]*10,prod(size(x)),1);
%   C = repmat([1 2 3],prod(size(x)),1);
%   fHdl=figure; hdl=gca;
%   scatter3(hdl, X(:),Y(:),Z(:),S(:),C(:),'filled'), view(-60,60);
%   [film2d, film3d] = camera_jitter_3d_record(hdl);
%   figure; movie(film2d,10);
%   figure; movie(film3d,10);
%

switch nargin
    case 0
        hdl = gca;
        angle = 28;
        step = 1;
    case 1
        % uses the axes handle passed in
        hdl = varargin{1};
        angle = 28;
        step = 1;
    case 2
        % uses the axes handle passed in
        hdl = varargin{1};
        angle = varargin{2};
        step = 1;
    case 3
        % uses the axes handle passed in
        hdl = varargin{1};
        angle = varargin{2};
        step = varargin{3};
end

film = camera_record_3d_stripe(hdl, angle, step);
frames = length(film);
frSize = zeros(frames, 2);
```

```
for f=1:frames
  [frSize(f,1) frSize(f,2) z ] = size(film(f).cdata);
end
frameMax = max(frSize);
center = floor(frames/2);
last = center*2;
% assume about 14deg binoccular angle for 3D {+/- 7 degrees}
bino = max(1,floor(14/step));
start = 1 + bino;
stop = last - bino;
sample = film(1).cdata(1,1,:);  %sample background color
f=1;
for i=center:stop
    jFilm(f) = film(i);
    jFilm(f).cdata = reframeImage(jFilm(f).cdata, frameMax, sample);
    picLeft = reframeImage(film(i-bino).cdata, frameMax, sample);
    picRight = reframeImage(film(i+bino).cdata, frameMax, sample);
    film3d(f) = make_sterio_pair(picLeft, picRight);
    f=f+1;
end
for i=stop:-1:start
    jFilm(f) = film(i);
    jFilm(f).cdata = reframeImage(jFilm(f).cdata, frameMax, sample);
    picLeft = reframeImage(film(i-bino).cdata, frameMax, sample);
    picRight = reframeImage(film(i+bino).cdata, frameMax, sample);
    film3d(f) = make_sterio_pair(picLeft, picRight);
    f=f+1;
end
for i=start:center
    jFilm(f) = film(i);
    jFilm(f).cdata = reframeImage(jFilm(f).cdata, frameMax, sample);
    picLeft = reframeImage(film(i-bino).cdata, frameMax, sample);
    picRight = reframeImage(film(i+bino).cdata, frameMax, sample);
    film3d(f) = make_sterio_pair(picLeft, picRight);
    f=f+1;
end

if nargout == 1,
    varargout = {jFilm};
elseif nargout == 2,
    varargout = {jFilm, film3d};
end

% Helper function
function imgOut = reframeImage(imgIn, frameMax, sample)
% reframes imgIn to center of imgOut which is sized to frameMax
frameSz = size(imgIn);
xOff = floor((frameMax(1) - frameSz(1))/2);
yOff = floor((frameMax(2) - frameSz(2))/2);
imgOut = zeros(frameMax(1), frameMax(2), 3, 'uint8');
for i=1:3
    imgOut(:,:,i) = sample(i);
end
```

```
xIndices = (1 + xOff) : (frameSz(1) + xOff);
yIndices = (1 + yOff) : (frameSz(2) + yOff);
imgOut(xIndices, yIndices, :) = imgIn(:,:,:);
```

A.4 Function: Camera Record 3D Stripe

This function takes the handle for a graph axes and record images of the graph as it orbits the camera position around the data. These images can be used to create 2D animations, 3D-jitter GIF files, and 3D stereo pair animations.

```
function varargout = camera_record_3d_stripe(varargin)
% Orbit camera position to capture series of pictures for 3d
%
% Inputs: zero or an axes handle
%
% Outputs: cell array of structs with images in 'cdata'
%
% Format:
%   camera_record_3d_stripe() - default using current (gca) axes
%
%   camera_record_3d_stripe(handle) - default using provided axes
%                                      or figure handle
%
%   camera_record_3d_stripe(hdl, angle) - handle & stripe angle
%
%   camera_record_3d_stripe(hdl, angle, step) - handle, stripe angle &
%                                        step size
% Defaults:
%   axes =  gca
%   stripe angle coverage = 32 deg
%   step size  = 1 deg
%
% GOOD example:
%   [x,y,z] = sphere(16);
%   X = [x(:)*.5 x(:)*.75 x(:)];
%   Y = [y(:)*.5 y(:)*.75 y(:)];
%   Z = [z(:)*.5 z(:)*.75 z(:)];
%   S = repmat([1 .75 .5]*10,prod(size(x)),1);
%   C = repmat([1 2 3],prod(size(x)),1);
%   fHdl=figure; hdl=gca;
%   scatter3(hdl, X(:),Y(:),Z(:),S(:),C(:),'filled'), view(-60,60);
%   film = camera_record_3d_stripe(hdl);
%   figure; movie(film,1);
%
% BAD example ??
%   fHdl=figure; hdl=gca;
%   surf(hdl, peaks);
%   axis vis3d;
%   film = camera_record_3d_stripe(hdl);
%   figure; movie(film,1);

switch nargin
    case 0
        hand = gca;
        angle = 32;
        step = 1;
    case 1
        % uses the axes handle passed in
        hand = varargin{1};
        angle = 32;
        step = 1;
    case 2
        hand = varargin{1};
        angle = varargin{2};
        step = 1;
    case 3
        % uses the axes handle passed in
        hand = varargin{1};
        angle = varargin{2};
        step = varargin{3};
end

figHand = getFigureHandle(hand);
figure(figHand); %bring to front
if figHand ~= hand
    axes(hand);  %if capture is on an axes make it current
end
axis vis3d; % adjust aspect for 3D motion

f=1;
camorbit(hand, angle/-2,0,'data',[0 0 1]);
drawnow;
film(f) = getframe(hand);
f=f+1;
for i=1:step:angle
    camorbit(hand, step,0,'data',[0 0 1]);
    drawnow;
    film(f) = getframe(hand);
    f=f+1;
end

if nargout == 1,
    varargout = {film};
end
```

A.5 Function: Camera Distance Sort

This function sorts 3D point data into display order from back (most distant) to front (closest) to ensure correct occlusion of displayed points. In addition it provides vectors to colour and/or size the display points with respect to distance from the camera (viewer).

```
function [ sData, distClr, ptSz ] = cameraDistanceSort( data, camPos, maxPtSz )
%Sort data be distance from the camera position for depth perception
%   data - the display data the first 3 columns are treated as X,Y,Z
%   camPos - the [ X Y Z ] position of the camera
%   maxPtSz - integer maximum point size (minimum 30)
%   sData - the input data with rows sorted by distance form the camPos
%   distClr - encodes distance from camPos in a range of 1-64
%           (column vector equal in size to rows of data)
%   ptSz - encodes the size of the display point for each data point
%           (column vector equal in size to rows of data)
%

posDat = data(:,1:3);
[s,f] = size(posDat);
dist = distance3D(posDat, camPos);
mn = min(dist);
distMap = dist-(mn - 1);
mx = max(distMap);
scale = 64/mx;
distClr = distMap*scale;
%distMap = 64 - distMap;

sz = max(maxPtSz, 30);
scale = (sz-5)/mx;
ptSz = sz - (distMap*scale);
%Sz = 66 - distMap;

[dSort,dIndex] = sort(distMap);
sData = data(dIndex,:);
distClr = distClr(dIndex,:);
ptSz = ptSz(dIndex,:);
end


function [dist] = distance3D(data, camPos)

deltaX = data(:,1) - camPos(1,1);
deltaY = data(:,2) - camPos(1,2);
deltaZ = data(:,3) - camPos(1,3);

dist = sqrt(deltaX.*deltaX + deltaY.*deltaY + deltaZ.*deltaZ);
end
```

A.6 Demo: Occlusion of Background Points

This code demonstrates correct and incorrect occlusion of background points in scatter plots. It demonstrates the utility of the *cameraDistanceSort* function.

```
% Demo Occlusion of Background Points

offset = zeros(200,4);  % along default camera axis
offset(:,1) = offset(:,1)+3;
offset(:,2) = offset(:,2)+3;
offset(:,3) = offset(:,3)-2.2;
cameraPos = [ -15, -15, 11];

posDat = [randn(200,4)+offset; randn(200,4)-offset];
posDat(1:200,4) = 1; % group 1
posDat(201:400,4) = 2; % group 2
rd = rand(400,1);
[sRd, sIndex] = sort(rd);
posDat = posDat(sIndex, :); % randomize order

% display data without depth sort
hFig = figure;
axes_handle = gca;
mapClr = cool;
colormap(mapClr);
X = posDat(:,1);
Y = posDat(:,2);
Z = posDat(:,3);
ptClr = posDat(:,4);
scatter3(X,Y,Z,45,ptClr,'filled');
camPos(cameraPos);
set(get(axes_handle,'XLabel'),'String','X axis');
set(get(axes_handle,'YLabel'),'String','Y axis');
set(get(axes_handle,'ZLabel'),'String','Z axis');

% display data with depth sort
hFig = figure;
axes_handle = gca;
colormap(mapClr);
posCam = campos(axes_handle);
[ sData, distClr, ptSz ] = cameraDistanceSort( posDat, posCam, 45 );
X = sData(:,1);
Y = sData(:,2);
Z = sData(:,3);
ptClr = sData(:,4);
scatter3(X,Y,Z,45,ptClr,'filled');
camPos(cameraPos);
set(get(axes_handle,'XLabel'),'String','X axis');
set(get(axes_handle,'YLabel'),'String','Y axis');
set(get(axes_handle,'ZLabel'),'String','Z axis');

% rotate view to see structure
hFig = figure;
```

```
axes_handle = gca;
colormap(mapClr);
posCam = campos(axes_handle);
[ sData, distClr, ptSz ] = cameraDistanceSort( posDat, posCam, 45 );
X = sData(:,1);
Y = sData(:,2);
Z = sData(:,3);
ptClr = sData(:,4);
scatter3(X,Y,Z,45,ptClr,'filled');
camPos([ 15, -15, 11]);
set(get(axes_handle,'XLabel'),'String','X axis');
set(get(axes_handle,'YLabel'),'String','Y axis');
set(get(axes_handle,'ZLabel'),'String','Z axis');
```

A.7 Function: Get Figure Handle

This function can take a graph axis handle and return the handle for the figure that contains it.

```
function varargout = getFigureHandle(varargin)
% find the Figure for this handle

switch nargin
    case 0
        handleIn = 0;
    case 1
        % uses the handle passed in
        handleIn = varargin{1};
end

handle = get(handleIn, 'Parent');
if handle == 0
    handleOut = handleIn;
else
    handleOut = getFigureHandle(handle);
end

if nargout == 1,
    varargout = {handleOut};
end
```

A.8 Function: Make Stereo Pair

This function will make a simple stereo pair stereogram image from left and right images.

```
function varargout = make_sterio_pair(varargin)
% make 3D side-by-side stereo image from 2 images
```

```
%
% Inputs: 2 images OR cell array of struct & 2 indices
%                       images stored in field 'cdata'
%
% Outputs:  struct with sterio pair in field 'cdata'
%
%   camera_3d_pic(im1, im2) - 2 images
%
%   camera_3d_pic(array, index1, index2) - struct array & indices
%
% example
%

switch nargin
    case 2
        % uses the axes handle passed in
        pic1 = varargin{1};
        pic2 = varargin{2};
    case 3
        % uses the axes handle & binoccular angle
        strArray = varargin{1};
        index1 = varargin{2};
        index2 = varargin{2};
        pic1 = strArray(index1).cdata;
        pic2 = strArray(index2).cdata;
    otherwise
        '**** Incorrect number of arguments ****'
end

[y1 x1 z1] = size(pic1);
[y2 x2 z2] = size(pic2);
y = min([y1 y2]);
pic = zeros(y,x1+x2,3, 'uint8');
index=1:x1;
pic(1:y,index,:) = pic1(1:y,index,:);
index=(1:x2) + x1;
pic(1:y,index,:) = pic2(1:y,1:x2,:);

out = struct( 'cdata', pic, 'colormap', []);
if nargout == 1,
    varargout = {out};
end
```

A.9 Function: Movie2GIF

This function will convert MATLAB movie frames to an animated GIF file. This code is a small modification of the “image2animation” function by Moshe Lindner which is available on the MATLAB file exchange.

```
function [ output_args ] = movie2gif( movieFrames )
```

```
%Convert movie frames to animated GIF file
%
% Modification of image2animation.m
% by Moshe Lindner , Bar-Ilan University, Israel.
% September 2010 (C)

numFrames = length(movieFrames);
file_path = cd;
[file_name2 file_path2]=uiputfile('*.gif','Save as animated GIF',file_path);

lps=questdlg('How many loops?','Loops','Forever','None','Other','Forever');
switch lps
    case 'Forever'
        loops=65535;
    case 'None'
        loops=1;
    case 'Other'
        loops=inputdlg('Enter number of loops? (must be an integer between 1-65535)        .','Loops');
        loops=str2num(loops{1});
end

delay=inputdlg('What is the delay time? (in seconds)        .','Delay');
delay=str2num(delay{1});
dly=questdlg('Different delay for the first image?','Delay','Yes','No','No');
if strcmp(dly,'Yes')
    delay1=inputdlg('What is the delay time for the first image? (in seconds)        .','Delay');
    delay1=str2num(delay1{1});
else
    delay1=delay;
end
dly=questdlg('Different delay for the last image?','Delay','Yes','No','No');
if strcmp(dly,'Yes')
    delay2=inputdlg('What is the delay time for the last image? (in seconds)        .','Delay');
    delay2=str2num(delay2{1});
else
    delay2=delay;
end

h = waitbar(0,['0% done'],'name','Progress') ;
for i=1:numFrames
    frame = movieFrames(i);
    [M  c_map]= rgb2ind(frame.cdata,256);

    if i==1
        imwrite(M,c_map,[file_path2,file_name2],'gif','LoopCount',loops,'DelayTime',delay1)
    elseif i==numFrames
        imwrite(M,c_map,[file_path2,file_name2],'gif','WriteMode','append','DelayTime',delay2)
    else
        imwrite(M,c_map,[file_path2,file_name2],'gif','WriteMode','append','DelayTime',delay)
    end
    waitbar(i/numFrames,h,[num2str(round(100*i/numFrames)),'% done']) ;
end
close(h);
```

```
msgbox('Finished Successfully!')


end
```

A.10 Function: Parallel Coordinates Browser

The code for the Parallel Coordinates Browser is available on the CD which accompanies this report. The code was not included herein since the GUI code file is not standard text and the m-file for the function definition (header listed below) is not functional without the GUI code file.

```
function varargout = Parallel_Coords_Browser(varargin)
% PARALLEL_COORDS_BROWSER M-file for Parallel_Coords_Browser.fig
%      PARALLEL_COORDS_BROWSER, by itself, creates a new PARALLEL_COORDS_BROWSER or raises the
existing
%      singleton*.
%
%      H = PARALLEL_COORDS_BROWSER returns the handle to a new PARALLEL_COORDS_BROWSER or the
handle to
%      the existing singleton*.
%
%      PARALLEL_COORDS_BROWSER('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in PARALLEL_COORDS_BROWSER.M with the given input arguments.
%
%      PARALLEL_COORDS_BROWSER('Property','Value',...) creates a new PARALLEL_COORDS_BROWSER or
raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Parallel_Coords_Browser_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Parallel_Coords_Browser_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
```

A.11 Demo: Simulate Distance Perspective for 3D

This demo plots 4 graphs which illustrate the effect of the depth cues discussed in Section 4.2.1.

```
% Simulate distance perspective for 3D

hFig = figure;
axes_handle = gca;
mapClr = bone;
colormap(mapClr);
posDat = [randn(100,3)+ones(100,3); randn(100,3)-ones(100,3)];
X = posDat(:,1);
```

```
Y = posDat(:,2);
Z = posDat(:,3);
scatter3(X,Y,Z,'filled');
posCam = campos(axes_handle);
[s,f] = size(posDat);
[ sData, distClr, ptSz ] = cameraDistanceSort( posDat, posCam, 50 )
X = sData(:,1);
Y = sData(:,2);
Z = sData(:,3);
oneClr = X;
oneClr(:,1)=1;
% ============================
scatter3(axes_handle,X,Y,Z,45,oneClr,'filled');
set(get(axes_handle,'XLabel'),'String','X axis');
set(get(axes_handle,'YLabel'),'String','Y axis');
set(get(axes_handle,'ZLabel'),'String','Z axis');
%load distancecolormap;

hFig = figure;
axes_handle = gca;
% colormap(distancecolormap);
colormap(mapClr);
scatter3(axes_handle,X,Y,Z,ptSz,oneClr,'filled');
set(get(axes_handle,'XLabel'),'String','X axis');
set(get(axes_handle,'YLabel'),'String','Y axis');
set(get(axes_handle,'ZLabel'),'String','Z axis');
%load distancecolormap;

hFig = figure;
axes_handle = gca;
% colormap(distancecolormap);
colormap(mapClr);
scatter3(axes_handle,X,Y,Z,45,distClr,'filled');
set(get(axes_handle,'XLabel'),'String','X axis');
set(get(axes_handle,'YLabel'),'String','Y axis');
set(get(axes_handle,'ZLabel'),'String','Z axis');
%load distancecolormap;

hFig = figure;
axes_handle = gca;
% colormap(distancecolormap);
colormap(mapClr);
scatter3(axes_handle,X,Y,Z,ptSz,distClr,'filled');
set(get(axes_handle,'XLabel'),'String','X axis');
set(get(axes_handle,'YLabel'),'String','Y axis');
set(get(axes_handle,'ZLabel'),'String','Z axis');
%scatter3(X,Y,Z,distMap,'filled');
```

A.12 Function: Simple Tour

This function takes hyper dimensional data and plots a 3D perspective scatter plot of the first 3 dimensions then performs a flying tour around the data and swaps one dimension at a time until all dimensions have been displayed (NOTE: not all combinations are displayed). This is a much simplified and much shorter tour then a true “Grand Tour”.

```
function  varargout = simple_tour(varargin)
% Simplified Grand Tour of Hyperdimensional data
%
% Inputs: data, (colour), (labels)
%
% Outputs: none
%
% simple_tour(nDimData );
% simple_tour(nDimData, colourVector );
% simple_tour(nDimData, colourVector, axisLabels );
%
% Example-
% load nDim_TestData;
% simple_tour(nDim(:,20:27), nDim(:, 2) );
%

switch nargin
    case 1
        data = varargin{1};
        sz = size(data);
        color = ones(sz(1),1);
        hasLabels = 0;
    case 2
        data = varargin{1};
        color = varargin{2};
        hasLabels = 0;
    case 3
        data = varargin{1};
        color = varargin{2};
        labels = varargin{3};
        hasLabels = 1;
    otherwise
        '**** Incorrect number of arguments ****'
end

step = 5;
az = 0;
el = 0;
pauseT = 0.4;

sz = size(data);
flds = sz(2);
if flds < 4
    error( 'Not enough dimensions for Tour');
end
```

```
X = data(:,1);
Y = data(:, 2);
Z = data(:, 3);
fHdl=figure; hdl=gca; axis vis3d; view(0,0);
scatter3(hdl, X, Y, Z, 6, color, 'filled');
if hasLabels == 0
    for i=1:flds
        labels(i) = {['C:' num2str(i)]};
    end
end
xL = 1;
yL = 2;
zL = 3;
xlabel(hdl, labels{xL});
ylabel(hdl, labels{yL});
zlabel(hdl, labels{zL});

view(hdl, 0,0);
pause(pauseT);
nxtCoord = 4;
for k = 1:ceil(flds/4)
    for i=step:step:90
        az = i;
        if i > 45
            el = 90-i;
        else
            el = i;
        end
        view(hdl, az,el);
        pause(pauseT);
    end
    %az=90, el=0; swap X axis
    X = data(:, nxtCoord);
    scatter3(hdl, X, Y, Z, 6, color, 'filled');
    xL=nxtCoord;
    xlabel(hdl, labels{xL});
    ylabel(hdl, labels{yL});
    zlabel(hdl, labels{zL});
    nxtCoord = mod(nxtCoord, flds) + 1;
    for i=step:step:90
        az = i + 90;
        el = i;
        view(hdl, az,el);
        pause(pauseT);
    end
    %az=180, el=90; swap Z axis
    Z = data(:, nxtCoord);
    scatter3(hdl, X, Y, Z, 6, color, 'filled');
    zL = nxtCoord;
    xlabel(hdl, labels{xL});
    ylabel(hdl, labels{yL});
    zlabel(hdl, labels{zL});
    nxtCoord = mod(nxtCoord, flds) + 1;
    for i=step:step:90
        az = i + 180;
        el = 90-i;
        view(hdl, az,el);
        pause(pauseT);
    end
    %az=270, el=0; swap X axis
    X = data(:, nxtCoord);
    scatter3(hdl, X, Y, Z, 6, color, 'filled');
    xL= nxtCoord;
    xlabel(hdl, labels{xL});
    ylabel(hdl, labels{yL});
    zlabel(hdl, labels{zL});
    nxtCoord = mod(nxtCoord, flds) + 1;
    for i=step:step:90
        az = i + 270;
        if az>=360
            az = 0;
        end
        if i > 45
            el = 90-i;
        else
            el = i;
        end
        view(hdl, az,el);
        pause(pauseT);
    end
    %az=0, el=0; swap Y axis
    Y = data(:, nxtCoord);
    scatter3(hdl, X, Y, Z, 6, color, 'filled');
    yL = nxtCoord;
    xlabel(hdl, labels{xL});
    ylabel(hdl, labels{yL});
    zlabel(hdl, labels{zL});
    nxtCoord = mod(nxtCoord, flds) + 1;
end


if nargout == 1,
    varargout = {[]};
end
```

Annex B MATLAB File Exchange Programs

B.1 STEREOVIEW

```
function stereoview
% STEREOVIEW  Stereoscopic plotting function
%
%   stereoview grabs the current 3D figure and plots it in two
%   stereoscopical images.
%
%   Once launched, you can:
%   1) Choose the interocular distance coefficient (idc)
%   that is defined as the ratio between the distance from the Camera
%   Target and the interocular distance; a value of 30 works fine.
%
%   2) Choose the viewing mode: 'parallel' or 'crossed'.
%
%           3) Rotate the view. In the Free Rotation mode, use:
%   a, s, d, w to rotate,
%   x to stop.
%
%   4) Start a continuous rotation of the view; x to stop.
%
%   5) Print the two stereoscopical views in separate images. You
%   can change the options for the saved images directly in the .m file
%   at lines 163 and 169.
%
%   Version 4.0 NEWS
%   - stop of the continuous rotation with "x"
%   - can print the two images
%
%   Version 5.0 NEWS
%   - can change the distance between the two
%     views to allow easier visualization
%   - can change zoom
%   - vertical axis bending bug fixed
%
% ----------------------------------------------------------
% (c) 2006  Iari-Gabriel Marino
%          <iarigabriel.marino@fis.unipr.it>
%          http://www.fis.unipr.it/home/marino/
% ----------------------------------------------------------
%   Example 1
%
%   [X,Y,Z] = peaks(30);
%   surfc(X,Y,Z)
%   stereoview
%   ------------------------
%   Example 2
```

```
%
%           daspect([1,1,1])
%           load wind
%           xmin = min(x(:));
%           xmax = max(x(:));
%           ymin = min(y(:));
%           ymax = max(y(:));
%           zmin = min(z(:));
%           daspect([2,2,1])
%           xrange = linspace(xmin,xmax,8);
%           yrange = linspace(ymin,ymax,8);
%           zrange = 3:4:15;
%           [cx cy cz] = meshgrid(xrange,yrange,zrange);
%           hcones = coneplot(x,y,z,u,v,w,cx,cy,cz,5);
%           set(hcones,'FaceColor','red','EdgeColor','none')
%           hold on
%           wind_speed = sqrt(u.^2 + v.^2 + w.^2);
%           hsurfaces = slice(x,y,z,wind_speed,[xmin,xmax],ymax,zmin);
%           set(hsurfaces,'FaceColor','interp','EdgeColor','none')
%           hold off
%           axis tight; view(30,40); axis off
%           camproj perspective; camzoom(1.5)
%           camlight right; lighting phong
%           set(hsurfaces,'AmbientStrength',.6)
%           set(hcones,'DiffuseStrength',.8)
%   stereoview
%   -----------------------
%   Example 3
%
%           [x y z v] = flow;
%           h = contourslice(x,y,z,v,[1:9],[],[0],linspace(-8,2,10));
%           axis([0,10,-3,3,-3,3]); daspect([1,1,1])
%           camva(24); camproj perspective;
%           campos([-3,-15,5])
%           set(gcf,'Color',[.5,.5,.5],'Renderer','zbuffer')
%           set(gca,'Color','black','XColor','white', ...
%       'YColor','white','ZColor','white')
%           box on
%   stereoview
```

B.2 ANAGLYPH

```
function varargout = anaglyph(varargin)
% ANAGLYPH - displays an anaglyph obtained by two stereoscopical images
%
% anaglyph with no arguments, opens a dialog to pick an image which contains
% two stereoscopical images in a single image file.
%
```

```
% anaglyph('filename') displays an anaglyph of the image "filename"
% (with extension), which contains two stereoscopical images in a single
% image file (bmp, jpg, or other supported image type).
%
% anaglyph('filename1', 'filename2') displays an anaglyph from the two images
% "filename1", "filename2" (with extension) one for the right eye and one for
% the left one.
%
% anaglyph(A) displays an anaglyph of the image A, in memory,
% which contains two stereoscopical images in a single image
% file (bmp, jpg, or other supported image type).
%
% anaglyph(A,B) displays an anaglyph from the two images A and B, in memory,
% (with extension) one for the right eye and one for the left one.
%
% Inputs: one or two names of image files or of matlab variables
%
% Outputs (optional):
%    the figure handle to the anaglyph
%
% Notes:
% One can swap the two images during the visualization.
%
% Version 2.0 NEWS
% - now plots also images in memory
% - with no args, opens a dialog to choose the image
% - zoom function
%
% Example:
%    anaglyph('foto3d.bmp')
%
% Other m-files required: none
% Subfunctions: plotAnaglyph, swap
% MAT-files required: none
%
% See also: stereoview
%
% Author: Iari-Gabriel Marino, Ph.D., nonlinear optics
% University of Parma, Dept. of Physics, ITALY
% email address: iarigabriel.marino@fis.unipr.it
% Website: http://www.fis.unipr.it/raman/index2.htm
% Personal: http://www.fis.unipr.it/home/marino/
% November 2006; Last revision: 6-Nov-2006
```

B.3 IMAGE2ANIMATION

```
% This program creates a movie/slideshow from a set of images, and save it as an animated GIF file.
% Notice that the quality an image may decrease due to the GIF format.
```

```
%
% Written by Moshe Lindner , Bar-Ilan University, Israel.
% September 2010 (C)

clear all
[file_name file_path]=uigetfile({'*.jpeg;*.jpg;*.bmp;*.tif;*.tiff;*.png;*.gif','Image Files (JPEG, BMP, TIFF, PNG and
GIF)'},'Select Images','multiselect','on');
file_name=sort(file_name);
[file_name2 file_path2]=uiputfile('*.gif','Save as animated GIF',file_path);
lps=questdlg('How many loops?','Loops','Forever','None','Other','Forever');
switch lps
    case 'Forever'
        loops=65535;
    case 'None'
        loops=1;
    case 'Other'
        loops=inputdlg('Enter number of loops? (must be an integer between 1-65535)        .','Loops');
        loops=str2num(loops{1});
end

delay=inputdlg('What is the delay time? (in seconds)        .','Delay');
delay=str2num(delay{1});
dly=questdlg('Different delay for the first image?','Delay','Yes','No','No');
if strcmp(dly,'Yes')
    delay1=inputdlg('What is the delay time for the first image? (in seconds)        .','Delay');
    delay1=str2num(delay1{1});
else
    delay1=delay;
end
dly=questdlg('Different delay for the last image?','Delay','Yes','No','No');
if strcmp(dly,'Yes')
    delay2=inputdlg('What is the delay time for the last image? (in seconds)        .','Delay');
    delay2=str2num(delay2{1});
else
    delay2=delay;
end

h = waitbar(0,['0% done'],'name','Progress') ;
for i=1:length(file_name)
    if strcmpi('gif',file_name{i}(end-2:end))
        [M  c_map]=imread([file_path,file_name{i}]);
    else
        a=imread([file_path,file_name{i}]);
        [M  c_map]= rgb2ind(a,256);
    end
    if i==1
        imwrite(M,c_map,[file_path2,file_name2],'gif','LoopCount',loops,'DelayTime',delay1)
    elseif i==length(file_name)
        imwrite(M,c_map,[file_path2,file_name2],'gif','WriteMode','append','DelayTime',delay2)
    else
        imwrite(M,c_map,[file_path2,file_name2],'gif','WriteMode','append','DelayTime',delay)
    end
    waitbar(i/length(file_name),h,[num2str(round(100*i/length(file_name))),'% done']) ;
```

```
end
close(h);
msgbox('Finished Successfully!')
```

List of acronyms

3D	3 Dimensional
API	Application Program Interface
BW	bandwidth
CANEWS	Canadian Naval Electronic Warfare System
DOF	Degrees Of Freedom
El	elevation
ELINT	Electronic intelligence
ERAPS	ESM Recording, Analysis and Playback System
ESM	Electronic Support Measures
GPU	Graphics Processor Unit
HUD	Heads-up Display
IP	intra-pulse
LCD	Liquid Crystal Display
LH	left-hand
LOB	line of bearing
OpenGL	Open Graphics Library
PC	Personal Computer
Pol	polarization
PW	pulse width
RF	radio frequency
RH	right-hand
S/N	signal to noise ratio
SEI	Specific Emitter Identification

TOA	time of arrival
VR	Virtual Reality

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.) John Altoft	2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.) UNCLASSIFIED

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

Data Visualization for ESM and ELINT: Visualizing 3D and Hyper Dimensional Data

4. AUTHORS (last name, followed by initials – ranks, titles, etc. not to be used)

John Altoft

5. DATE OF PUBLICATION (Month and year of publication of document.) June 2011	6a. NO. OF PAGES (Total containing information, including Annexes, Appendices, etc.) 98	6b. NO. OF REFS (Total cited in document.) 10

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Contract Report

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

Defence R&D Canada – Ottawa
3701 Carling Avenue
Ottawa, Ontario K1A 0Z4

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.) 11aj02	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.) W7714-050965
10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.) ASI-2010-09-003	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.) DRDC Ottawa CR 2011-084

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

Unlimited

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.))

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

The last decade has witnessed a massive shift in 3D (3 Dimensional) device technology from high end virtual reality labs into the consumer market. Up until a decade ago, a virtual reality headset with the computer to drive it would likely have cost hundreds of thousands of dollars. Today a capable PC (personal computer) equipped with a 3D graphics processor unit (GPU), 120Hz refresh rate LCD (Liquid Crystal Display) monitor, and a pair of LCD shutter glasses, costs around $2000 dollars.

This project investigated 3D display technology and data visualization techniques for application to the ESM (Electronic Support Measures) and ELINT (electronic intelligence) domains. ESM and ELINT signal analysis are inherently complex and involve multi-dimensional analysis. The application of 3D graphic presentation to improve user comprehension of interim or final ESM signal analysis results was examined. This investigation examined both 3D representations presented in 2D and also true 3D visualization as is currently possible with modest Virtual Reality (VR) display systems.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

ESM; ELINT; 3D; visualization