Thinking about 3D – design constraints for effective 3D interfaces

A. Evans, J.Agenjo, Eduard Gonzales, Sergi Gonzales, M.Dematei, M.Romeo, J.Blat

Universitat Pompeu Fabra, Barcelona, Spain.
alun.evans@upf.edu Fax : 93 542 25 17

Abstract

Understanding the design of 3D graphical user interfaces is not as simple as doing so for their 2D counterparts. While 3D naturally has an extra dimension in which to display graphics, and thus has the potential to increase the accessible area in which information can be presented to the user, 3D also presents entirely new ways to impede user performance and increase disorientation. As a result, 3D interfaces have had mixed success since their introduction. In this paper, we critically discuss the evolution of the 3D interface, in both the academic and commercial fields. This leads us highlight the four most relevant human computer interaction (HCI) design constraints for the effective implementation of 3D interfaces. We then present an example application designed with these constraints in mind, and focused on the specific use case of visualising simple metadata in post-production.

Keywords: 3D, GUI, interfaces, design, graphics, metadata

1. Introduction

The ways in which humans interact with computers has seen surprisingly little change for several decades. Despite the public interest with 3D and 'pseudo-3D' graphical user interfaces, as reflected in the fanciful interfaces of large-budget movies, the standard keyboard-mouse-screen interface is still the default choice for almost all human-computer interaction. Yet, the rise in power of desktop computing has allowed the proliferation of 3D design effects such as raised buttons, shadows, reflections and see-through windows. While it is easy to dismiss these effects as "eye-candy" or merely an advanced visual presentation, other work [1] suggests that positive experiences will improve user performance. This is reflected in the commercial sector, where several popular products and applications[2][3][4] demonstrate that 'increased productivity' is not necessarily the only indicator of performance.

Understanding the design of 3D interface techniques ('3D' in this sense referring to interactive 3D effects seen on screen, as opposed stereoscopic 3D) is not as simple as their 2D counterparts. While 3D naturally has an extra dimension in which to display graphics, and thus has the potential to increase the accessible area in which information can be presented to the user, 3D also presents entirely new ways to impede user performance and increase disorientation. Additionally, development complexity increases when using a 3rd dimension, requiring a large performance gain to justify the added development time and expense. The 2D windows, icons, menus and pointer metaphor (WIMP) [5] has dominated human-computer interaction for decades, and for good reason.

The history of 3D user interfaces has seen a boom-and-bust cycle not dissimilar to rise, fall, and subsequent repeat rise of stereoscopic 3D. The initial excitement of a new technology lead to disappointment with the lack of exploitation, which is now gradually returning to an increased optimism (lead by the commercial sector) that the technology has a future.

In this paper, we review the evolution of the 3D interface, and analyse it in order to extract information that can be used to guide future work. This leads us to the principal contribution of this paper: the proposal of a design backbone for 3D interfaces, manifested by highlighting the *four most relevant human computer interaction design constraints* that should be followed when dealing with 3D HCI. The paper then presents an example of how these constraints can be used, via the design and implementation of a 3D interface for a specific use case - the rapid pre-visualisation of large volumes of industrial post-production image data.

2. Related Work

3D user interfaces first gained popularity in the first 3D boom of the early 90s, when mass-market dedicated GPUs allowed genuine 3D processing onto the desktop. After this initial boom, interest in 3D interfaces largely died away to near zero, and the concept still suffers from negative first impressions [6]. There are three likely reasons for this near-death of the 3D interfaces:

i) Dedicated GPUs allowed 3D graphics to be displayed on desktop hardware, but nevertheless 3D applications still consumed valuable CPU and memory resources. Thus, such applications invariably drew heavily on system resources in general, and slowed down the performance of the entire machine.

ii) Very little thought was put into the design of initial 3D user interfaces beyond "making it look nice"[6]. More emphasis was placed on visual design and attractiveness, and little to no effort was made in thinking of new ways of how the depth could provide more information to the user.

iii) The combinations of reasons (i) and (ii) above meant that the business world – dominated by productivity – ignored 3D interfaces. This lead to a vicious circle, where the few 3D

interfaces that were created were generally the result of amateur collaborative effort, with an understandable lack of professional polish.

2.2 Academic work

The initial boom in 3D interfaces resulted in much academic interest[7][8][9][10]. However, by 2001, the academic evidence appeared to both inform and reflect the increasingly negative view of using 3D interfaces for everyday tasks. An example of this was the work by Cockburn and McKenzie[11], who demonstrated that there was no advantage to using a 3D for certain specific file-management task, using a 3D document management interface called Data Mountain[7].

Yet, due to both the advent of increased processing power and, more importantly, more thought being applied to the functional design of 3D interfaces, between 2005 and 2010, interest began to return to the field. In 2004, Bowman et al.[5] published the first textbook on 3D interface design, providing the beginnings of a coherent academic basis to 3D interface design. This provided the seed for several academic research groups to begin to design and test results-driven 3D interfaces with the goal of actually increasing productivity, rather than using the 3D for visual effect[12][13].

Pu et al. [14] pursue such an approach, with the goal of creating a 3D interface for a specific use case (in the case, navigation of a 3D CAD repository). Their results suggest that users were able to browse and search in a more efficient manner when using the 3D version of their browser.

Leal et al.[15] approach the process from the point of view of the 'user experience'. Revisiting 3D file browsing and evaluating three separate approaches, their conclusions show that, of the three different interfaces that they created, two performed no better than 2D interfaces, and one performed significantly worse. However, another interesting result of theirs was that the reported user experience was more 'enjoyable'.

Baumgärtner et al. [16] appear to accept the limitations of 3D and attempt to fuse 2D and 3D in a hybrid approach, claiming an intuitive learning process for non-expert users. This approach is interesting as it is carried out from a desire to genuinely improve the user interface, and not lever 3D effects into a situation where they are not required.

One of the principal conclusions in many of these studies is the difficulty in obtaining unbiased data. Any critical evaluation requires comparison with previous systems and interfaces, however it is highly likely the majority of test-users will already be accustomed to existing interfaces, making learning bias against the new 3D interface and almost unavoidable issue.

2.3 Commercial/Industry

A further measure of the success of 3D interfaces can be obtained by analysing the impact that they have had within the commercial sector. The types of product that have been developed can broadly be split into three categories; those designed for workstation *desktop and file management*; *visual data* interfaces that use 3D to present data in a visually attractive way; and *applied data* interfaces, which use 3D to provide the user with additional information about a set of data.

In the *desktop and file management* category are grouped the many efforts made to replace the standard desktop with some form of 3D interface. Figure 1 shows screenshots from examples such as Tactile3D[17] and 3DNA[18], two of the many 3D file management systems either available, discontinued, or in development.

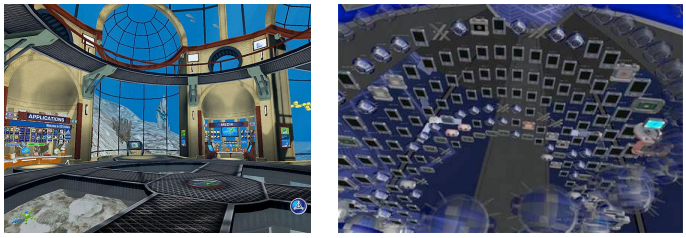
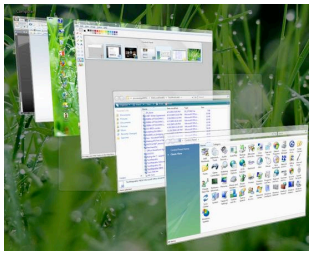


Figure 1: Screenshots of Desktop replacement services

Very few, if any, of these interfaces have gained any real traction or popularity, possibly for the reasons provided by Cockburn et al.[11] mentioned above.

Furthermore, when the first 3D management interfaces appeared, another considerable problem for users lay in the overuse of computing resources. Many of the applications used considerable resources and that meant that all other processes (word-processing, browsing, gaming etc.) were negatively affected. A recent (2008) example of this is AT&T's Pogo, a prototype web-browser which never made it to market, as private beta tests indicated that its hardware requirements were so high that it made the software almost unuseable[19].

A more practical use of 3D within this category are those interfaces that manage the workstation desktop, as opposed to making carrying out direct operations on files. These 3D interfaces are usually integrated within the standard desktop environment, rather than directly replacing it. Popular examples of 3D desktop management interfaces are Microsoft Windows Aero[20], and the Beryl[21] feature of several Linux desktops (See Figure 2).



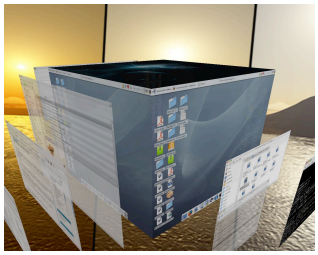


Figure 2: Microsoft Windows Aero and Linux Beryl

The key feature of these desktop management interfaces is that they are not intended to replace existing features (i.e. they are not forced upon the user), rather they are designed to augment the management of items on the desktop (mimicking, at least in principal, the 2D-3D hybrid approach of Baumgärtner et al.[16]). A further important point is that, at the time of their introduction (for example, the Aero interface was first released with Windows Vista in 2006/7) the processing power of computer hardware had increased to a level whereby the features could be enabled without noticeably deteriorating the performance of the system[21] (with the exception of graphically intensive processes such as video games).

3D interfaces have not been confined solely to desktop and file management interfaces. *Visual data* applications present data within a 3D environment, but use the 3D primarily to improve the visual effect. The 3D aspect doesn't necessarily provide any extra information about the data, but does allow it to be viewed in a different, attractive manner. In that sense, visual 3D data applications are similar to early desktop replacement applications (in that they make no real effort to directly improve productivity), but the prime difference is that they make a direct effort to appeal to the 'user experience', measured beyond performance metrics such as speed or efficiency. Examples of such applications are the CoolIris[3] (formerly PicLens) web-browser image viewer plugin, and minor features of certain internet browsers, such as the 'Top Sites' feature of Apple Safari[22].

The visual data category of 3D interface seems to have little impact in terms of effects on direct user productivity – however its relatively popular uptake has had considerable impact in commercial terms. CoolIris, pictured in Figure 3, recently received $15 million of second round venture capital funding, and claims that its plugin has been downloaded tens of millions of times[23]

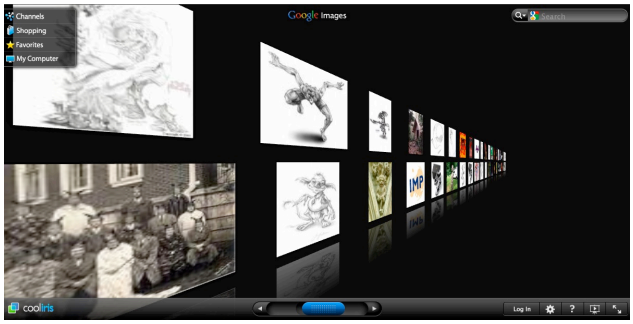


Figure 3: Screen capture of CoolIris image and video browser

The final category, *Applied data* 3D applications, is distinguished from the other two categories described above by the fact that the interfaces use the third dimension to present information about a set of data in a manner that would not be possible otherwise. In other words, the use of the third dimension is not merely used for making the interface more visually attractive – it actually provides the user with extra *information*.

A prime example of such an interface is Microsoft Photosynth[4]. While the key technological novelty of Photosynth lies in pattern recognition and point cloud generation, its interface locates the photos in relation to one another within a 3D space (see Figure 4); providing information to the user in a manner that would not otherwise be possible.

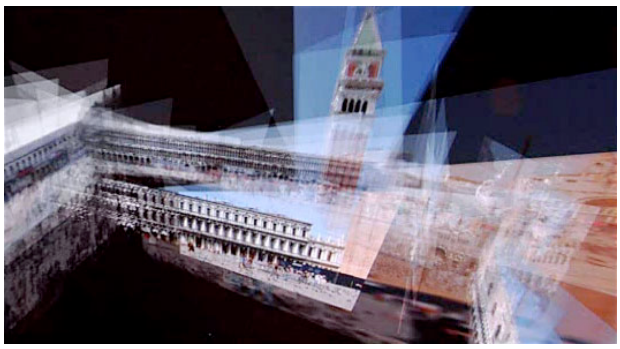


Figure 4: Microsoft Photosynth interface

Placing photographs within a 3D environment is also the goal of recent research from Nokia[24]. Their system uses GPS information to geo-locate photographs taken with camera phones. These images are then uploaded into a 3D-mapping application that overlays the images, in a 3D environment, onto a 2.5D map. Again, the 3D aspect of the interface is being used to provide information that would be more difficult to present otherwise. A further example of maps within a 3D interface is Google Earth[2], which provides a 3D interface with restricted camera controls, but lets the user view and move the entire globe, even tilting the camera to view the relief of the terrain.

It is worth pointing out that the three categories described above are not necessarily mutually exclusive, and as 3D interfaces become more popular in the commercial world, it is possible for interfaces to overlap categories. One example is the interface for Time Machine, the data back-up feature present on the Apple Mac OSX [25] operating system (See Figure 5). Time machine is, essentially, a file and desktop management application that deals with back-up data. However, the various backups made over time are presented along the *z*-axis of the 3D interface. The user intuitively understands that the images and icons further back along the *z*-axis represent older backups of data.

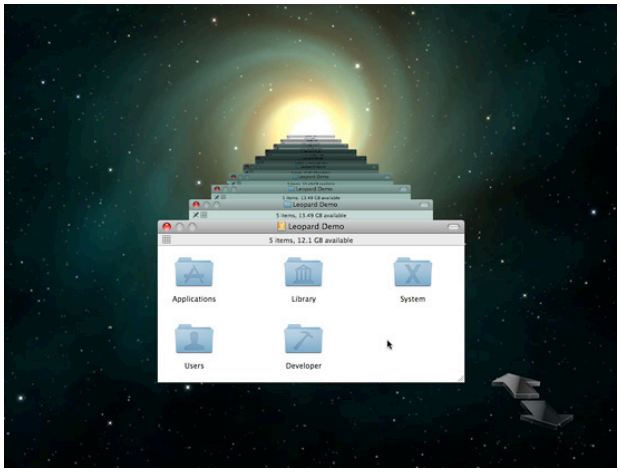


Figure 5: Apple OSX Time Machine interface

In summary, it is becoming apparent that 3D interfaces have undergone a resurgence in popularity, both in the academic and commercial sectors. This surge, along with the requirements of projects such as iMP, provides the principal motivation for this paper.

3. Design Constraints for 3D Interfaces

The conclusion from much of the academic work carried out on 3D interfaces is that *poorly created* interfaces both damage productivity and waste computing resources; and thus serve little or no purpose. However, recent research, combined with an increase in commercial popularity, suggests that 3D interfaces can also have the opposite effect, if they are well designed and implemented.

Thus, in this section we highlight the four most relevant HCI design constraints which, if followed, should ensure that a 3D interface should not fall into the trap of providing 'style over substance'. Although the constraints are by no means novel ideas within interface design, we propose them as the four *most important* constraints that, when fulfil, should assist in the creation of successful interfaces.

Constraint 1: Maintain familiarity. The data and literature[5][15] suggest that learning bias (the fact that users are more familiar with an interface that they have used before) is a significant hurdle for any new interface – let alone one that uses an even less familiar third dimension. Thus, to minimise confusion, any 3D interface should attempt to present data in a manner that is as familiar as possible to the user. Put another way, the challenge of the application is to present *new* information to the user, while letting them still navigate the data in a familiar way.

Constraint 2: 3D used to provide information. The main problem with many failed attempts at creating 3D user interfaces is that they have only used 3D to make the interface more visually appealing. While this is important, the best uses of 3D within interfaces are those where the third dimension should be used *to present new information*. Frequently, metadata provide additional information about the data being viewed, but they are often presented as text or figures. 3D interfaces give the opportunity to present that metadata to the user in an instinctive way. Perhaps the simplest and most successful example of this is Apples's Time Machine interface, where 'time' metadata are represented within the third dimension.

Constraint 3: Interface is fast and uses few resources. If the interface is designed to improve the users experience and/or productivity in doing an existing task, it should be fast and use as few resources as possible. For example, if a file manager takes several seconds to load the display, and uses up half the available processing power, it is far less useful that a standard 2D file manager, despite any features provided by the 3D component. Not following this constraint led to the demise of several projects, most recently Pogo[19].

Constraint 4: Visually attractive interface. Despite constraint 2 above, all novel interfaces will need to impress the user in order gain popularity. CoolIris is an excellent example of an application that uses 3D to create a highly attractive visual effect which appeals to users, thus driving uptake of its product and increasing revenue. Furthermore, this constraint is particularly important if the interface is to be used within the creative industry, which is used to applications having a high level of visual polish.

Table 1 shows four successful implementations of 3D interfaces and comments on how they perform when considering each constraint. As the table shows, only one of the interfaces, Photosynth, could be said to not fulfil all of the constraints, as the software requires considerable processing power. However, it is reasonably unique in that a person using Photosynth is doing to do so for the specific features of the application, and therefore is likely to be more accepting of lower performance.

	1.	2.	3.	4.
Photosynth	Photos placed in situation that fits their content	Location metadata places photos within 3D environments	Interface is slow, particularly over internet.	Attractive interface presenting overlapping photos on black background
Google Earth	Globe of earth similar in style to real scale models	Pan and Tilt features allow metadata relief mapping and provide sense of distance	Fast movement in 3D; bottleneck lies in the data streaming speed	Real satellite images used as textures within 3D interface
CoolIris	Grid of images similar to Google Images or popular photo software	Metadata provides related images and videos, presented within 3D interface	Plugin for web-browsers. Fast, simple 3D effects require little real processing power	Attractive animations give sense of perspective of data.
Time Machine	Uses same folder icons and design	Metadata intuitively used to represent data	Complete and simple integration with OS	Use of alpha and attractive background adds to gloss

Table 1: Comparison of four commercial 3D interface with the proposed constraints

4. Using metadata for 3D browsing of post-production data

In this section, we present a prototype application that was created according to the design constraints presented in Section 3. The motivation behind the design of the application comes from the iMP FP7 European collaborative project[26]. iMP's goal is to produce a step change in the efficiency of audiovisual postproduction and distribution, along with improved methods of producing finished versions of media for distribution to the correct destination. Within iMP, our role is to investigate novel ways of using metadata to present information, and develop creative applications based on this research.

Importantly, we are not proposing this interface as a general replacement for file operations. Rather, we are presenting it as example of how the four proposed design constraints should be followed when designing an application interface for a specific use-case (described below).

4.1 Use case

As the iMP project deals with post-production data and metadata, one of the motivations behind this work was the desire to create novel creative applications that make better use of metadata to improve post-production workflow. One characteristic of post-production work is that it invariably involves enormous amounts of image-based data. Production work is stored as image sequences; the same sequence may be stored at several different image resolutions, and any post-production modification (e.g. grading, conforming) to each sequence must be saved as different files. This naturally leads to many terabytes of data for each production, and this in terms requires a highly structured data storage system, where every member of the post-production team knows exactly in which subdirectory a particular type of data belongs.

The problem with such a highly structured directory tree is that it leads to large amount of directory tree redundancy, in the sense that even if the data doesn't exist, the directory that should contain it does exist. This in turns leads to loss of productivity in the search for data, as during the development of a project it is likely that many folders remain empty – without any indication of them being so.

Figure 6 shows a screenshot of a typical Microsoft Windows Explorer window showing an example of a sample post-production data data and directory structure, obtained via industrial post-production partners within iMP. Each project home directory is given a name (in this case blanked out for privacy reasons) and that directory contains a suibdirectory containing the results for each post-production pass that has been applied to the data (e.g. 'FLAME_EXP' and 'SHAKE_EXP' would contain the image data after processing by Autodesk Flame and (the now discontinued) Apple Shake applications.

Each of these export directories is further split into five subdirectories, representing the different resolutions at which the images are stored (1K, 2K, 4K, HD and SD). Each of these further subdirectories are then potentially split again into LEFT and RIGHT directories, if the clip has been filmed in stereoscopic 3D. Clips are stored in the directories as a series of sequential image files representing each frame of the clip. The images are either raw uncompressed files, or compressed using a lossless format, at the resolution specified by the name of the parent directory. Any metadata associated with the files is present within in the file itself.

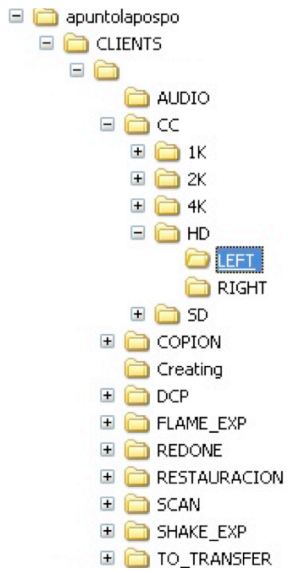


Figure 6: Example of Lapospo data structure in a typical Windows Explorer window

Our primary objective thus was to create a 3D interface to navigate this structure effectively, guided by the constraints of Section 3.

The other objective for the interface was to act as future plugin for the large scale distributed metadatabases and fileservers that are being developed elsewhere within the iMP project. These databases and servers will provide rapid access to large volumes of data and metadata, and thus the interface implementation should be heavily XML based and easily extensible for further integration with future technology.

4.2 Interface design and implementation

According to the requirements of the use case, and the design constraints set out earlier in the paper, the interface was designed to present the directory-based image data in a way that uses basic metadata to provide useful information at a glance. With this in mind, the design and implementation of the application can be split into two parts, requiring different approaches; a *3D directory viewer*, and a *streaming video onto 3D surfaces* component.

4.2.1 3D Directory viewer

The 3D directory viewer provides three different 'views' of the data. The user is able to rapidly switch between views by clicking icons in the top-left corner of the screen.

The **Standard Directory View** is exactly that (see Figure 7). Upon initialising the application, the standard directory view presents the directory structure to the user, and allows all the expected interactions: double click to expand/close a tree branch, keyboard shortcuts for fast navigation, right-click context menus to open files in different programmes, etc. Although this particular view makes no effort to use 3D effects, it is important as it fulfils **Constraint 1** from section 3; it is an immediately obvious and familiar experience for the user. The iconography is familiar, and folder and filenames are displayed clearly to the right of the icons. Scrolling is accomplished via the mouse wheel. The 'current directory' is highlighted in a slightly different colour shade, both for the icon and the corresponding text. The only two differences to a standard directory view is that scrolling is also accomplished via click-dragging the background (holding mouse button and moving the mouse up or down); and video images are shown on the cover of directories which contain image sequence files (see section 4.2.2 below).

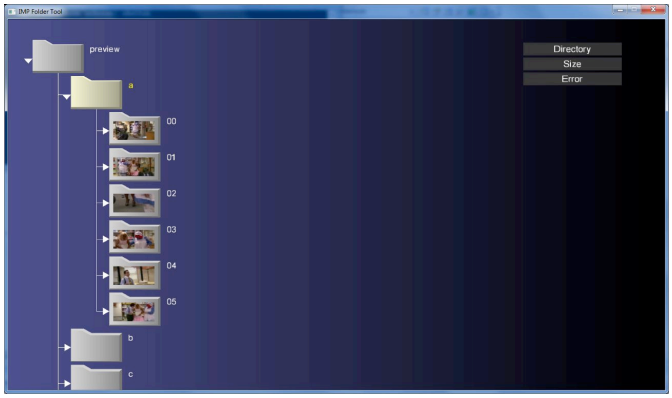


Figure 7: The Standard Directory view of the example application

The second view implemented by the application is termed the **Directory Size View**. As discussed in section 4.1 above, the rigid directory structure maintained for post-production project allows much room for redundancy. By following **Constraint 2**, this view of the interface uses 3D to visualise directory metadata that is usually either hidden, or displayed as numbers.

Two straightforward examples of such directory metadata is the number of files in the directory, and the size (in bytes) of the directory. The Directory Size view uses this metadata to turn the directory icons into polygons, where:

- the size (in bytes) of the directory is represented by the frontal area (*x* and *y* axes) of the polygon
- the number of files in the directory is represented by the depth (*z* axis) of the polygon

Figure 8 show examples of the effect. The results are that directories that have small numbers of large files (for example, video sequences stored in one file) have large frontal areas but little depth, whereas directories that have large numbers of smaller files (for example, video sequences stored as individual images) have small frontal areas but much depth.

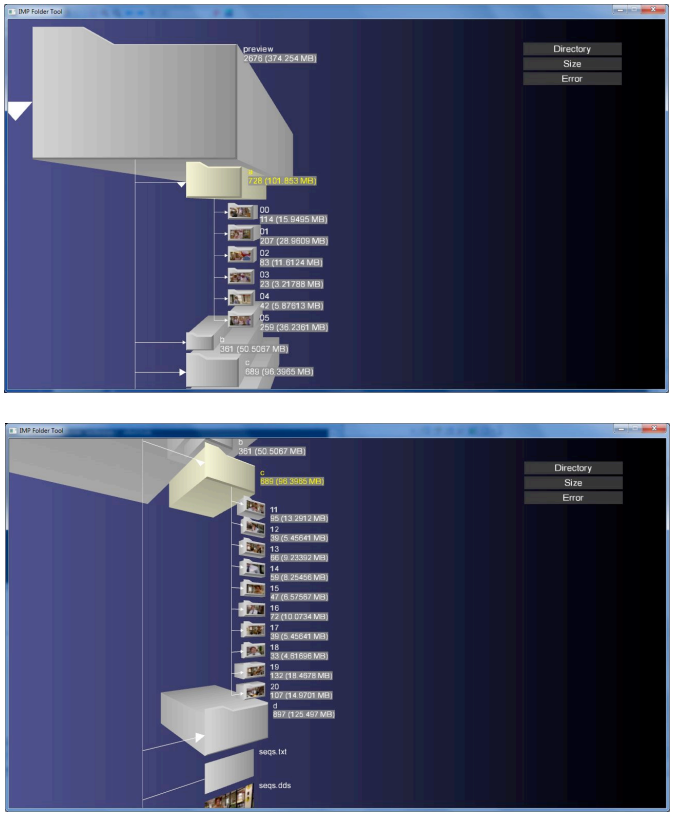


Figure 8: Screenshots of Directory Size view, from two different angles

As soon as the view is activated using the left-hand menu, the camera perspective changes in order to better see the effect. The user user is able to rotate the directory tree by click-dragging with the right mouse button. All the other features from the standard view (scrolling, key control etc) are maintained.

The view is designed in order to follow **Constraint 2** closely: the 3D aspect is only used to visualise metadata that would otherwise would require more interaction to view, and would likely only be viewed as text or figures.

The final view of the application is more use-case specific, but nonetheless also follows **Constraint 2** closely. One typical problem with post-production data is that the process of ingesting video is subject to errors, and in a directory of thousands of image frames, typically there will be several frames ingested incorrectly. These corrupted image frames are usually identified by having a very small file-size – indeed, when using raw image data the file size should be identical for every frame, so even a minor change in size indicates a corrupt file.

Typically, these corrupt files are identified by the user when viewing the video sequence, or when browsing the file size metadata column in a standard file explorer. To combat this, our interface defines an **Error view**, which is designed specifically to assist in identifying corrupt files. The Error view maintains the standard folder size from the Standard Directory view (i.e. folder frontal areas are standard). The most important difference is the use of colour to indicate corrupt files within a folder:

- If a folder does not contain any corrupt files, and neither do any of its sub-folders, then the folder icon is shaded green.
- If the folder contains corrupt image files, it is shaded red, and a red polygon *z*-axis depth represents the number of files
- The location of the corrupt images within the sequence is shown by a lighter shade of red, and clicking on the lighter area leads directly to a view of the corrupt files.
- If a folder does not contain corrupted files, but one of its sub-folders *does* contain corrupted files, then the parent folder is shaded red, but is not elongated as a polygon.

Figure 9 shows a screenshot of the view in action.

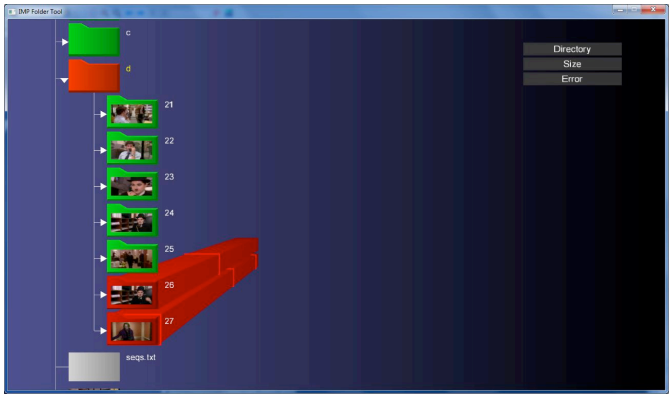


Figure 9: Screenshot of the error view

We observed carefully both **Constraint 1** and **Contraint 2** in the design of this view. The 3D effect is only be used when it is strictly necessary to view errors. There is no purpose in showing 3D polygons for any folder in this view unless that folder directly contains corrupt image files.

4.2.2 Streaming video onto 3D surfaces

An interesting feature request from the use-case was the ability to view thumbnail video clips on the surface of the directory, and furthermore have rapid access to actual video sequence. As the sequences are invariably stored as separate image files, usual methods of viewing them involve loading the image sequences into a separate viewing application, all of which takes time.

While streaming video onto a 2D interface is straightforward, doing the same onto a 3D interface is technically more challenging. First, the video sequences are pre-processed

using the methods described in section 4.2.3 below. The processed images are then applied as a texture overlay for a flat rectangular surface, drawn within the 3D interface. The shape of the video textured surface (VTS) is set according to the aspect ratio of the corresponding video clip, and the VTS texture is updated according to the clip's framerate. Multiple VTSs can be displayed on the screen simultaneously and, due to the general nature of 3D applications, can be translated, rotated and scaled in a variety of ways, while still maintaining real-time video texture update.

Maintaining multiple video streams in memory is a difficult task that requires careful management of the flow of data. To facilitate this, culling algorithms are used to avoid updating the video on VTSs that are not visible in screen, thus reducing the video memory bandwidth. With the current configuration, we are able to display between 8 to 12 video sequences at the same time without visible performance problems. The result is moving video thumbnails showing the sequence content of multiple folders, in real time (see Figures 7 and 8).

Once the user selects a directory for more detailed preview, the interface zooms to that particular sequence, and presents a standard video player with Play/Pause controls, and a progress bar to allow movement within the sequence (see figure 10).

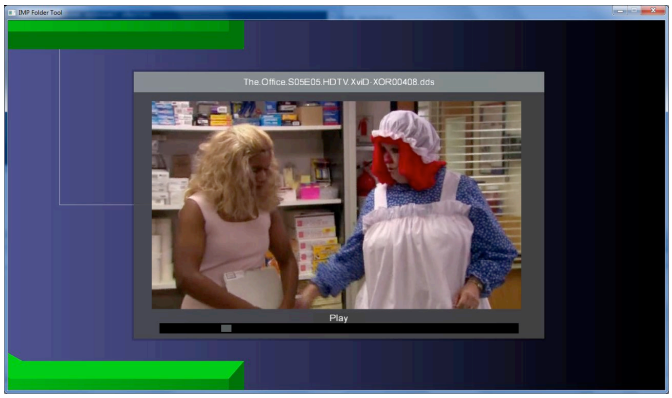


Figure 10: Preview of sequences with standard playback controls

4.2.3 Performance and visual design

Constraint 3 states that the interface should be fast, and use few resources. The more general goal of the iMP project is to provide fast access to metadata in order to speed up the sort of database and file-server requests that are required by the application described in this paper. Given that the metadatabase work is not yet finished, we were forced to undertake a pre-processing step in order to fulfil Constraint 3.

The data is pre-processed using a file crawler that generates both an XML file with the error metadata required for the Error view, and the DDS textures of the image files used for the real-time video display. The disadvantage of this crawler system is that any changes in the structure will not be updated in the XML automatically. However, the metadatabase work in iMP is designed to directly address this problem, and so future integration should lead to excellent results.

Constraint 4 states that the interface should be visually attractive. To this extent, once the basic interface design was coded and proved to be working, a professional graphic designer created the final visual design scheme, iconography, and colour palette. The designer followed **Constraint 1** closely, ensuring in particular that the iconography and colours were similar in style to those used in other file management systems.

5. Conclusions and Future Work

In this paper we review the state of the art, both in the academic and commercial domain, in the field of 3D interfaces. In carrying out this review, our objective was to see if there were any common features shared by successful 3D interfaces and, if so, if those features could be used to guide future interface design.

Thus, the contribution of this paper is the review of previous work in order highlight the four most relevant HCI design constraint*s* for the creation of successful 3D interfaces. After showing how popular 3D interfaces already successfully implement these constraints, we report the design process and implementation of an interface that fulfils the constraints, and also the design requirements of a specific use case, in this case the use of metadata to improve previsualisation of post-production image data.

The main conclusion of this work is that, since the first 3D interfaces appeared, here has been a lack of concrete design principals applied to their creation. The study of the state of the art shows a haphazard approach, both in the academic and commercial sectors. After the initial boom in publications and applications (between 1990 and 2000), interest in 3D interfaces fell, and only really revived academically following the publishing of the first book on the subject in 2004[5].

Since that time, 3D interfaces have developed and improved, which has enabled us to extract the four constraints proposed in section 3. Although each constraint is important, the order in which they are presented perhaps reflects their importance to one another. It is apparent that the hybrid 2D-3D approach[16], reflected in Constraint 1, is important in order to not confuse the user, while studies of commercial applications show that Constraint 2 is important in order to make popular interfaces.

Constraints 3 and 4 are less conceptual and more straightforward – it is clear that fast, attractive applications and interfaces will always perform and sell better than those that force the user to wait, and have poor or incoherent graphic design.

The example application presented in section 4 was created while applying the design process from section 3, and aimed at resolving the challenges posed via a defined use case. Our immediate future work is to critically evaluate the interface, using performance metrics that should carefully reflect the requirements of the use-case and attempt to remove, or at least identify, familiarity bias.

Further work will focus on the tighter integration of metadata, and the iMP metadatabase, in order the remove the data pre-processing step, and perhaps introduce new views that enable the user to view high-level semantic connections between sequences of data (e.g. shots clustered by actor).

It is hoped that the proposals in this paper, and the examples of their implementation, will lead to a new, formal approach to the design of 3D interfaces. This should allowing future work to be critically evaluated, both in terms of performance and design, which in turn will lead to constructive and practical use of 3D graphics within user interfaces.

Acknowledgements

This work is funded by the iMP FP7 project[26]. Many thanks to Apunto Lapospo for providing data and valuable insight in the post-production process, and for Dark Ride Studios for assistance with aspects of the development.

References

[1] D. A. Norman. Emotional design. Ubiquity, 4(45):1–1, 2004

[2] Google Earth. http://earth.google.com/

[3] CoolIris. http://www.cooliris.com/

[4] Microsoft Photosynth. http://www.photosynth.net/

[5] D. A. Bowman, E. Kruijff, J. J. LaViola Jr., and I. Poupyrev. 3D User Interfaces: Theory and Practice. Addison-Wesley, ACM. Boston, 2004.

[6] A. Cockburn and B. McKenzie. Evaluating the effectiveness of spatial memory in 2d and 3d physical and virtual environments. In *CHI '02: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 203–210, 2002.

[7] G. Robertson, M. Czerwinski, K. Larson, D. C. Robbins, D. Thiel, and M. van Dantzich. Data mountain: using spatial memory for document management. In *UIST '98: Proceedings of the 11th annual ACM symposium on User interface software and technology*, pages 153–162, 1998.

[8] S. Snibbe, K. Herndon, D. Robbins, D. Brookshire Conner and van Dam. Using Deformations to explore 3D widget design. *ACM SIGGRAPH Computer Graphics*, 26(2):351-352, 1992.

[9] P Cubaud, C Thiria, A. Topol. Experimenting a 3D interface for the access to a digital library. In *Proceedings of the third ACM conference on Digital Libraries*, pages 281-382, 1998.

[10] G. Robertson, J Mackinlay, and S.K. Card. Cone Trees: Animated 3D visualizations of hierarchical information. In *CHI '91: Proceedings of SIGCHI conference on Human factors in computing system*, pages 189-194, 1991.

[11] A. Cockburn and B. McKenzie. 3D or not 3D? Evaluating the Effect of the Third Dimension in a Document Management System. In *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 434-441, 2001.

[12] B. Zhu and H. Chen. Using 3d interfaces to facilitate the spatial knowledge retrieval: a geo-referenced knowledge repository system. *Decis.Support Syst.*, 40(2):167–182, 2005.

[13] S. Xu, T. Jin and F. Lau. A new visual search interface for web browsing. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 152-161. 2009.

[14] J. Pu, Y. Kalyanaraman, S. Jayanti, K. Ramani. Z. Pizlo. Navigation and Discovery in 3D CAD Repositories. *IEEE Computer Graphics*, 27(4):38-47, 2007.

[15] A Leal. C. Wingrave and J. LaViola Jnr. Initial explorations into the user experience of 3D file browsing. In *Proceedings of the 2009 British Computer Society Conference on Human-Computer Interaction*, pages 339-344, 2009.

[16] S. Baumgartner, A. Ebert, M. Deller, and S. Agne. 2D meets 3D: a human-centered interface for visual data exploration. In *CHI' 07: CHI' 07 extended abstracts on Human factors in computing systems*, pages 2273–2278, 2007.

[17] Tactile 3D Interface. http://www.tactile3d.com/

[18] 3DNA. http://www.3dna.net/

[19] J. Cheng. First look: AT&T's Pogo browser beta tries too hard, fails. http://arstechnica.com/software/news/2008/04/pogo-browser-beta-first-look.ars/ 2008

[20] Windows Aero. http://www.microsoft.com/windows/windows-vista/features/aero.aspx

[21] A.Weiss. Desktops in 3D. *networker*, 11(1):26-33 2007.

[22] Apple Safari. http://www.apple.com/uk/safari/

[23] A. Ha. CoolIris raises $15M for (improved) 3D wall. http://venturebeat.com/2009/04/12/cooliris-raises-15m-for-improved-3d-wall/

[24] A. Lucero, M. Boberg and S. Uusitalo. Image Space: capturing, sharing and contextualizing personal pictures in a simple and playful way. In *Proceedings of ACM ACE '09,* pages 215-222. 2009.

[25] Apple Mac OSX. http://www.apple.com/uk/macosx/

[26] iMP. http://www.imp-project.eu/