Gaze-coupled Perspective for Enhanced Human-Machine Interfaces in Aeronautics

N. Masotti[1] and F. Persiani[2]
University of Bologna, Bologna, Italy

In aeronautics, many Virtual/Augmented Reality (V/AR) facilities, such as flight simulators, control-tower simulators, remote towers and flight reconstruction software, rely on the assumption that the viewer will most likely stay still in a pre-defined position. For this reason, they can be dubbed Desktop Virtual/Augmented Reality (D-V/AR) interfaces, in contrast with 'gaze-coupled' V/AR interfaces, which take into account the viewpoint position within the rendering pipeline.

Surprisingly, in spite of the rough perspective model being used, D-V/AR is often well accepted by both designers and final users. Indeed, in some cases, it yields to credible results. However, when the viewer's eyes move far away from their 'default' position, the rendering outcome will be affected by significant error, resulting in a poorly immersive and/or unrealistic experience.

This paper discusses gaze-dependent visual interfaces as a means to enhance Human-Machine Interaction (HMI) and visual perception in V/AR based aeronautical facilities. Within the dissertation, a classification of leading V/AR display techniques is given, including D-V/AR, Off-axis V/AR (O-V/AR), Generalized V/AR (G-V/AR), Stereoscopic V/AR (S-VAR), Head-coupled V/AR (H-V/AR) and Fish-Tank V/AR (F-V/AR). For each technique, benefits, downsides and constraints have been exposed. Also, a set of suitable applications for gaze-dependent HMI has been identified, including, but not limited to, flight simulation, flight reconstruction, air navigation services provision and unmanned aerial system governance.

I. Introduction

Virtual Environments (VE) may be defined as computer-based facilities capable of recreating sensory experiences, including taste, sight, smell, sound and touch. Nevertheless, most applications primarily focus on the visual component. Often, synthetic information is shown directly – i.e. the VE uses Virtual Reality (VR) as a medium –, but, in some cases, information might be superimposed to the physical world as well. In the latter case we would better talk about Augmented Environments (AE) rather than VE.

Nowadays, both VE and AE are profitably used in many fields, including entertainment, product design, automotive, navy and aeronautics.

Historically, several display techniques have been developed for Virtual/Augmented Reality (V/AR) facilities. Many of these have been used in aeronautical tools and equipment, such as flight simulators, control tower simulators, synthetic vision systems, head-up displays, flight-reconstruction software and manned/unmanned aircraft avionics. As a compromise between correctness and viability, most of these techniques merely approximate the viewer's perspective model, when, in fact, a much more complex algorithm should be used[2,3]. Furthermore, VEs often rely on specific equipment to be used. Therefore, V/AR developers are not only concerned with computer graphics content creation – namely modelling, texturing and animation – but also with software programming, artificial intelligence design, and I/O peripherals management. Moreover, for these systems to perform effectively, applied science, such as interface design, needs to draw upon basic sciences, such as computer vision, anthropometry, physiology and cognitive ergonomics[1]. The intertwining of these disciplines is sometimes referred as Human Factors and Ergonomics (HF&E).

[1] Ph.D. Student, Department of Industrial Engineering, nicola.masotti@unibo.it
[2] Full Professor, Department of Industrial Engineering, franco.persiani@unibo.it

II. Virtual/Augmented Reality display techniques classification

In V/AR facilities, display techniques are responsible for visually presenting computer-generated information to the user. Nearly all of conventional techniques operate on some variant of the pinhole camera metaphor, i.e. a camera object exists in the virtual environment and regularly takes bi-dimensional snapshots of the computer-generated scene, to be displayed on a physical device. Advanced techniques originate either as enhancements to conventional ones or as specific V/AR implementations. The number of such techniques is overwhelming; therefore, an in-depth report is well beyond the scope of this paper. However, a brief classification will be given. This will be based on the excellent job made in Ref. 4.

A. Conventional VR Display Techniques

1. Desktop Virtual/Augmented Reality (D-V/AR)

Desktop Virtual/Augmented Reality (D-V/AR) is a basic implementation of the pinhole camera model. D-V/AR is ubiquitously supported as the default output mode of nearly every graphics engine or application available today. It is based on a static projection model, which uses symmetrical *frustum*[3]. As a result, it produces a single, camera-centred, perspective image and does not require any special equipment to be used, meaning that any framed or unframed planar display is sufficient. As the simplest form of V/AR, D-V/AR avoids many issues, such as eyestrain, increased computational cost, latency, etc. For this technology to work properly, the user should be positioned relatively to the screen as the camera object is positioned with respect to the near clip plane[3], e.g. head-centred on the screen normal[4]. Given that proportions – i.e. horizontal and vertical Field Of View (FOV) – should be kept identical, a *frustum* scale factor is acceptable as long as the viewer is aware of watching an exaggerated or diminished virtual world. On the contrary, relative movements between the observer and the screen, including back and forward movements, should not be allowed, as they modify the physical *frustum,* whereas the projection model behind the software remains unvaried. In other words, D-V/AR should be considered a 'static' display technique.

2. Off-axis Virtual/Augmented Reality (O-V/AR)

Off-axis Virtual/Augmented Reality (O-V/AR) comes in handy when the viewing position is not screen-centred, meaning that the straight line from viewer's eyes to the screen, drawn along the screen normal direction, no longer strikes the display in the middle. In this case, an asymmetric *frustum* is used to render the scene. However, the near clip plane stays perpendicular to the camera depth axis, therefore, the same orientation has to be used for the physical display. Relative movements between the observer and the screen are still forbidden.

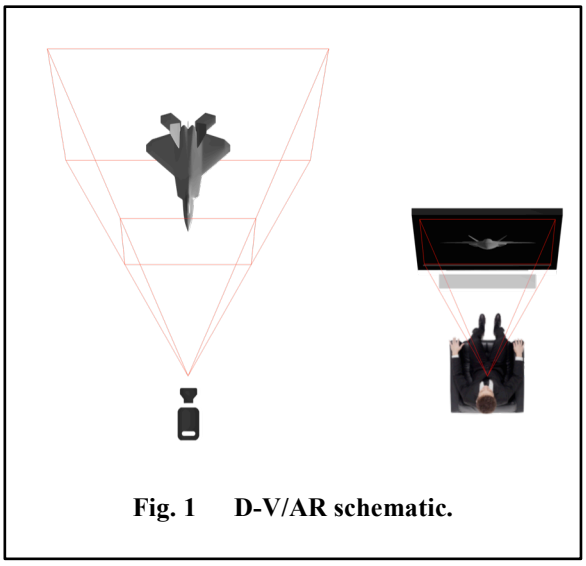

Fig. 1 D-V/AR schematic.

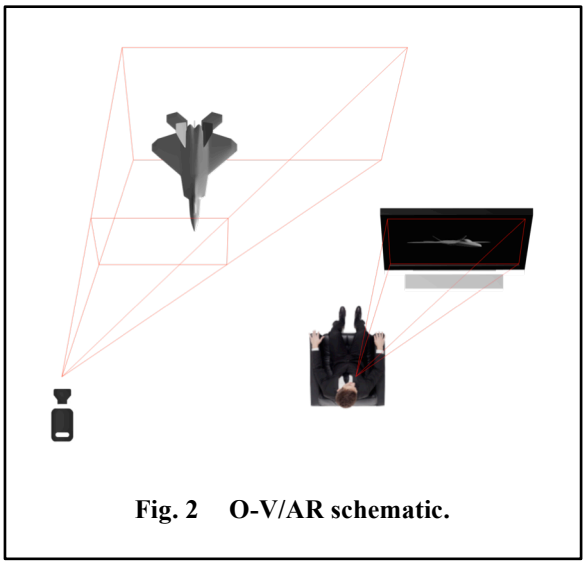

Fig. 2 O-V/AR schematic.

[3] A *frustum* is a six-sided truncated pyramid, which originates sectioning the shape the virtual camera field of view by means of two user-defined clipping planes. These are known as the 'far clip plane' and the 'near clip plane'. The latter, is the one on which the virtual world is projected as a necessary step of the rendering pipeline.

[4] For the sake of readability, here and from now on, we will refer to the straight line being orthogonal to the screen and passing by the center of it simply as the 'screen normal'.

3. Generalized Virtual/Augmented Reality (G-V/AR)

Generalized Virtual/Augmented Reality (G-V/AR) is an off-axis projection development that allows the projection plane, therefore the viewing device surface, to be arbitrary oriented. This is achieved by multiplying the standard off-axis projection matrix by a further rotation matrix (more about projection matrixes in section III). Once the viewer standpoint is known (and stays still), the display surface might be arbitrary oriented, i.e. rotated, installed upside down, laid flat on the floor or hung from the ceiling. For all intents and purposes, this makes G-V/AR applicable to a wide range of VE architectures, such as, fixed-viewpoint, non-planar, multi-screen VEs. A comprehensive G-V/AR code review can be found in Ref. 2.

4. Stereoscopic Virtual/Augmented Reality (S-V/AR)

Stereoscopic Virtual/Augmented Reality (S-V/AR) is a dual camera paradigm suited for binocular vision. Stereovision is achieved by rendering the virtual scene twice, once for each eye. Image pairs (a.k.a. stereo pairs) are encoded and filtered so that each single image is only seen by the matching eye. Encoding techniques include colour spectrum decomposition, light polarisation, temporal encoding and spatial encoding. Filtering is most easily attained through special equipment, e.g. polarized eyeglasses, coloured eyeglasses and shutter glasses, but might be also achieved by looking at the screen from a specific position. Encoding and filtering techniques are paired together and named Passive, Active or Auto-stereoscopic techniques, based on their working principles. The need for a triggering system in the filtering equipment – if any – determines whether a technique is Active – or Passive. Following this criterion, passive systems are colour filtering and polarisation, whereas temporal encoding, in combination with shutter glasses, is to be considered Active. Finally, auto-stereoscopic techniques, such as *Parallax Barrier* and *Lenticular Lens*, do not require additional filtering equipment because they encode spatially. In this case, it is the physical distance between the viewer's eyes that filters the images.

5. Head-Coupled Virtual/Augmented Reality (H-V/AR)

Head-Coupled Virtual/Augmented Reality (H-V/AR) operates on a slightly different principle than D-V/AR. It is the virtual window metaphor, rather than the pinhole camera model, that better fits this technique. A projection surface, representing the physical display, is defined in the virtual environment. Also, the viewer's head position is tracked in space and time. Virtual objects are projected through the so-defined surface, toward the user's head. Thus, the projection outcome depends on the relative position between the viewer's head and the projection surface. Clearly, a perspective projection is still used. Nevertheless, this time the projection model is not defined *a priori,* but rather computed 'just-in-time'. As a matter of fact, while the observer moves freely in the physical environment, the display becomes a framed window on the virtual world.

A strong H-V/AR limitation is that any other viewer, looking at the very same display, will perceive a distorted (incoherent) image. This is always true unless multiple perspectives are being used – i.e. calculated and displayed. To make this possible some special equipment has to be used, namely the same type of equipment that is used for S-V/AR.

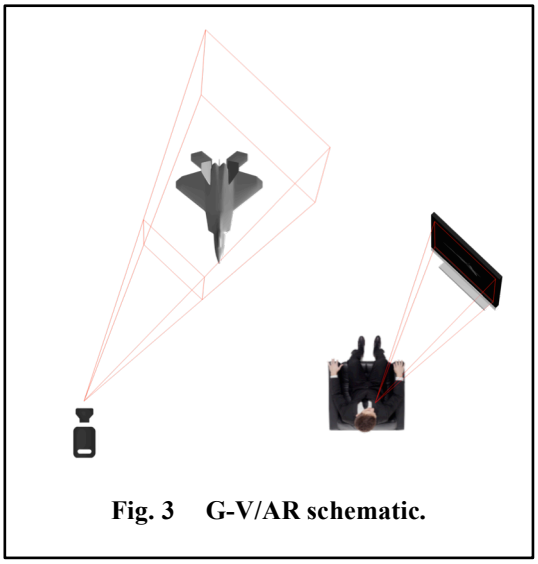

Fig. 3 G-V/AR schematic.

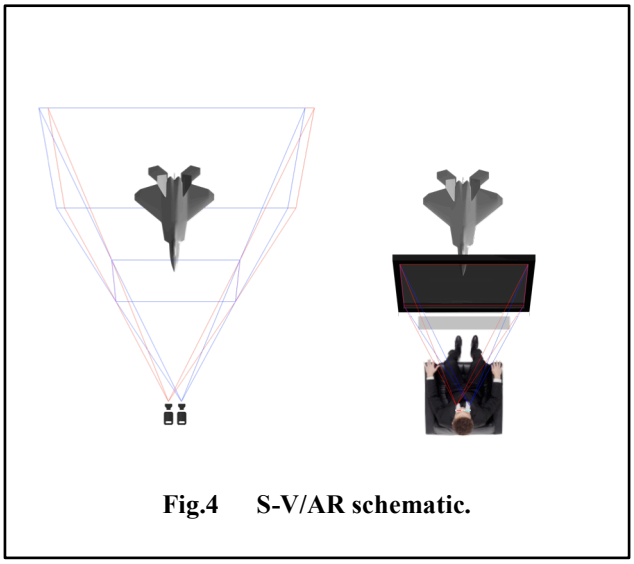

Fig.4 S-V/AR schematic.

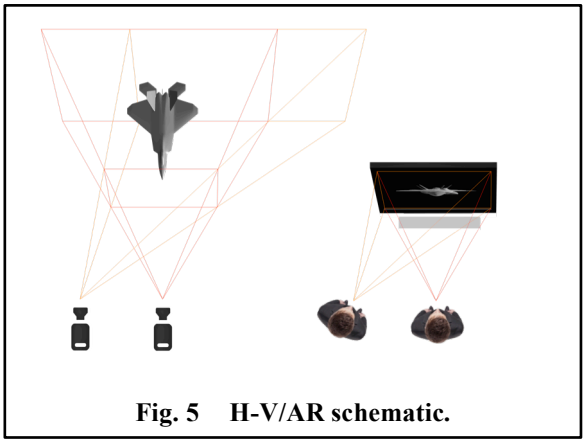

Fig. 5 H-V/AR schematic.

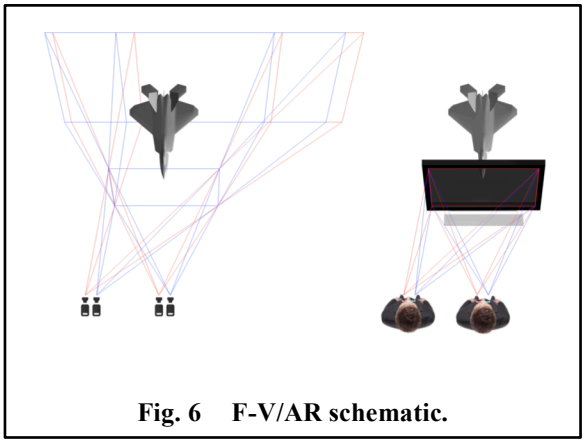

Fig. 6 F-V/AR schematic.

6. Fish-Tank Virtual/Augmented Reality (F-V/AR)

Fish-Tank Virtual/Augmented Reality (F-V/AR) – a.k.a. Eye-Coupled V/AR (E-V/AR)[4] or True Dynamic 3D (TD3D)[3] – is an improvement over H-V/AR, which separately considers the viewer's left and right eye position. As will be further discussed in section IV, F-V/AR is truly a combination of Eye-Coupled Perspective (ECP) and S-VAR. Whenever F-V/AR is involved, special encoding/decoding equipment is already being used for stereovision achievement and can hardly be exploited for more than that. Consequently, so far, F-V/AR has been a one person at a time experience. Further improvements may amend this.

7. Volumetric Virtual/Augmented Reality (V-V/AR)

Volumetric Virtual/Augmented Reality (V-V/AR) can be defined as the set of techniques that form a visual representation of a 3D model in the three physical dimensions. To some extent, holography might be considered as a form of V-V/AR as well.

B. Advanced VR display techniques

1. Head-Mounted Displays

Head-Mounted Displays (HMDs) and Sea-Through Head-Mounted Displays (ST-HMDs) are single-user V/AR equipment combining stereoscopy with a large FOV and head motion based interaction. While HMDs completely override the observer's visual perception of the physical world with a comprehensive view of the virtual environment, ST-HMDs superimpose additional information to it. This is accomplished by means of two tiny displays, paired together and positioned in front of the viewer's eyes (one per eye). Given the proximity of the screens, a lens system is used, allowing for a more natural focus. Multiple orientation trackers, such as three-axis gyros, accelerometers and magnetometers might be embedded in the headgear. Thanks to this, the user's head orientation (rotation) is tracked. This ensemble of technologies often results in rather cumbersome equipment, depending on your definition of 'rather' and 'cumbersome'. Notice that the HMD tracking system is different from the one used in H-V/AR or F-V/AR, where the head/eyes position (not the orientation) is tracked. For starters, because the HMD itself rotates with the head, the user's eyes position with respect to the screen does not change at all. This is why HDMs do not need an eye tracking system, unless some gaze-dependent interaction paradigm has to be used. The software requirements needed to support HMDs VR or ST-HMDs AR resemble the ones necessary for S-V/AR, with a few additional requirements concerning:

i. The distortion caused by the lens system, which has to be corrected for.
ii. The spatial orientation of the HMD, which must be considered as an input parameter by the rendering pipeline.

2. Power-walls and curved Mosaics

Power-walls and curved Mosaics are the most straightforward way to extend the FOV without decreasing spatial resolution – i.e. PPI (Pixel Per Inch) resolution. These are made out of many conventional displays tied together with the only propose to form a larger, wide-view, giant display. The result, either planar or curved, looks as if one was looking through a framed window. Therefore, appropriate bezel correction is used. Sometimes screen edges can be made to overlap with physical frames, such as in car and flight simulators, which is very convenient. Finally, it is worth noting that the perspective model given by a single, wide view *frustum* is only applicable to planar multi-displays architectures – i.e. Power-walls. If a curved arrangement is used, the perspective model must use multiple, however oriented, asymmetric *frusta* – i.e. G-V/AR.

3. Cave Automatic Virtual Environments

Cave Automatic Virtual Environments (CAVEs) are well known VEs that fill in the peripheral vision by means of multiple, rear-projected, flat screens. Three to six screens are typically arranged in a 'cubical' configuration, although unconventional architectures, such as the ones in Ref. 5 or Ref. 6, may be used as well. F-V/AR is used for rendering purposes, while edge blending might be needed for a seamless result.

4. Gaze-dependent depth of field V/AR

In a gaze-dependent V/AR application a scene is rendered and visualised while the viewer's eyes movements and relative position regulate a depth of field simulation. Saccades and fixations are captured by means of an eye tracker and the line of sight direction computed with respect to the virtual environment.

5. Fish-eye VR

Fish-eye VR (a.k.a. hemispherical *VR)* has been portrayed as the ultimate technology for VR: a synthetic (computer-based) environment where no frame impinges on the user comprehensive view of the virtual world. A seamless, widescreen, hemispherical display (a.k.a. 'dome') is used, so that the entire user's FOV is engaged. In order to avoid heavy distortion, a custom rendering pipeline is needed. First, a CAVE-like generalized projection is used. Second, a series of unconventional coordinate mappings adapt the result to be projected[7].

III. Conventional VR Display Techniques in depth

D-V/AR has been the dominant display technology since the advent of three-dimensional computer graphics. Most OpenGL applications simply select an horizontal FOV, specify a preferred aspect ratio, along with near clip plane and far clipping plane distances, and call gluPerspective(). This is a well-known OpenGL function that sets up a projection matrix and multiplies it by the 'current matrix'. In other words, a symmetrical *frustum* is used, in combination with certain specified parameters. The same goes for many Direct3D applications. In case you are not familiar with OpenGL or Direct3D Application Programming Interfaces (APIs), all you need to know is this: at some point of the graphics pipeline, a perspective projection is needed. Eventually, this gets done by means of a 4x4 projection matrix, which is multiplied by all vertexes in your scene[5]. When D-V/AR is involved, the projection matrix looks like this:

$$\begin{bmatrix} \frac{n}{r} & 0 & 0 & 0 \\ 0 & \frac{n}{t} & 0 & 0 \\ 0 & 0 & \frac{n+f}{n-f} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \qquad [1]$$

Where *r* and *t* represent half of the horizontal and vertical near clip plane extents respectively, while *n* (*nearVal*) and *f* (*farVal*) refer to the distances between the viewpoint (i.e. the camera origin) and the near and far clipping planes respectively. You may find extensive information about OpenGL projection matrix either on the Internet[8] or in the *OpenGL Programming Guide*, alias *The Red Book*[9]. For the purpose of this paper, just be aware of this: when a matrix such as [1] is used, a few underlying assumptions have been made, i.e. the viewer is positioned in front of the screen, facing perpendicular to it and looking at the centre of it. Moreover, relative movements between the eyes and the display surface, e.g. back and forward movements, are not accounted for, as they modify the physical FOV whereas the virtual camera projection model – i.e. the projection matrix – does not change at all. As already stated before, a scale factor between the physical and the virtual world, therefore between the physical and the virtual viewing volumes, is acceptable, provided that the user is aware of that. For a schematic of D-V/AR working principals please refer to Fig. 1.

As one can easily grasp, actual practice seldom satisfies these criteria. For starters, think about binocular vision. If the left eye is located in space in such a way that D-V/AR terms are satisfied, the same cannot be possibly said for the other eye and vice versa. Besides, what if the viewer moves away from his or her pre-defined position? Eventually, the field of V/AR introduces circumstances under which D-V/AR assumptions fail, resulting in intolerable inaccuracy.

Don't get us wrong; most of the time, perspective projection remains believable in spite of this. As cleverly recalled in Ref. 2:

> "Leonardo's 'The Last Supper' uses perspective, but still appears to be a painting of a room full of people regardless of the position from which you view it. Likewise, one can still enjoy a movie even when sitting off to the side of the theatre".

[5] In homogeneous coordinates, the vector representing a vertex position in space has four components: x, y, z and w.

Nevertheless, for a true bias free experience, the rendering pipeline must be modified to properly reflect the viewer's perceptual model. This is where glFrustum() comes in. glFrustum() is an OpenGL function that generalizes the standard 3D projection matrix as follows:

$$\begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{n+f}{n-f} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \quad [2]$$

This time, *l*, *r*, *b* and *t* represent the distances between the near clip plane edges and the straight line that goes from the camera origin to near clip plane itself (in a perpendicular manner). Again, you may find extensive information about glFrustum() input parameters – namely *l* (*left*), *r* (*right*), *b* (*bottom*), *t* (*top*), *n* (*nearVal*) and *f* (*farVal*) – either on the Internet[10] or in the in the *OpenGL Programming Guide*[9]. What really matters right now is that a matrix such as [2] allows for asymmetric *frusta* to be used. Therefore it frees the viewpoint position from the screen normal[6]. This is portrayed in Fig. 2. In case of need, the perspective projection can be determined separately for each eye-screen pair, resulting in a better V/AR implementation whenever stereovision is used (Fig. 6). As a matter of fact, the off-axis perspective model delivered by [2] is much more flexible than the one provided by [1]. Nevertheless, glFrustum() still assumes the near clip plane to be orthogonal to the virtual camera depth axis. Hence, the use of this function alone does not fit multi-screen, non-planar architectures, such as CAVEs or curved Mosaics. Moreover, it does not account for changes in the viewpoint position. In order to break free from these constraints, a far more generic perspective model is needed, i.e. Head Coupled Perspective (HCP).

HCP is a user-in-the-loop V/AR perspective model, whose only assumption is that the physical VE setup will not change while the V/AR application is being used. Therefore, the shape of each camera *frustum* depends on the observer's viewpoint position with respect to the screen. Once (and for all) the location of the screen corners is known and the user's head is being tracked in space and time, the *frustum* geometry is dynamically determined. This also defines the projection matrix. However, if we want the *frustum* to be pointed at any arbitrary direction, rather than to be 'aligned' with the camera depth axis direction, a further adjustment is needed before the scene is rendered. This consists of a matrix multiplication, which rotates the virtual world using the viewpoint position as a pivot, so that the rendering result will look like as if the *frustum* itself had been rotated around its apex (i.e. the camera origin). To all intents and purposes, this is an off-axis projection development that allows the near clip plane, therefore the viewing device, to adopt any arbitrary orientation. Whenever HCP is used, the viewer perceives the display as a physical frame through which the virtual world can be inspected. If s/he aims for a certain perspective, s/he might not only navigate the scene (as usual), but also try to change her viewing position with respect to the screen. This human-machine interaction archetype better suits those systems wherein the VE itself simulates the physical or metaphorical presence of frames, such as control-tower simulators, flight simulators or TT displays.

Finally, if you manage to track down the viewer's eyes position, render the scene twice (once for each eye) and set up one of the S-V/AR techniques described in section II, you will finally reach a proper F-V/AR implementation. For a comparison between G-V/AR and F-VAR pipelines, please refer to Fig. 7.

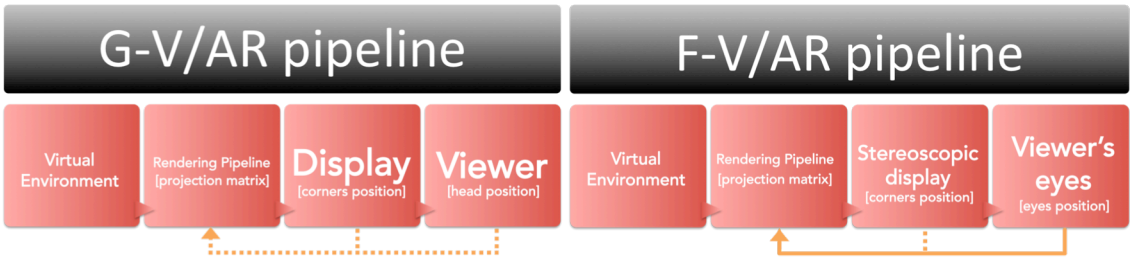

Fig. 7 Comparison between G-V/AR and F-V/AR pipelines. Dotted lines are used whenever spatial information is accessed by the control loop once and for all. Such information is typically stored and retrieved in editable configuration files or made available through graphical user interfaces. On the contrary continuous lines represent real time, constantly tracked information.

[6] The meaning of 'screen normal' has been clarified in section footnote 4.

IV. Stereovision

At this point of our argument, the reader may have already realized that Eye-Coupled Perspective (ECP) and stereovision are deeply bounded together. Nevertheless, s/he probably recalls having experienced stereoscopic 3D without the need for her or his eyes to be tracked. How was this possible? Truth is, even though stereo vision techniques have been around for at least six decades, there are still many widespread misconceptions. As Lang commented[11] on a brief interview with Oliver Keylos:

> "It turns out that rendering stereoscopic 3D images is not as simple as slapping two slightly different views side-by-side for each eye. There's lots of nuance that goes into rendering an appropriate 3D view that properly mimics real world vision – and there's lot's that can go wrong if you aren't careful".

This is especially true given that stereoscopic 3D has been pushing hard into the mainstream market segment over the last few years[12]. Furthermore, as noted by Keylos[12]:

> "The subtleties of improper 3D rendering could be a major hurdle to widespread consumer adoption of virtual reality, in a way that the everyday first-time VR user won't think: – this is obviously wrong, let me see how to fix it –. They'll say instead: – I guess 3D isn't so great after all; I'll pass. –".

In a nutshell, there are primarily three ways of generating stereo pairs: Parallel stereo, Toe-in stereo and Skewed-*frusta* stereo. As painful as this may sound, the latter is correct, whilst the others are not.

1. *Parallel stereo*

Parallel stereo is the easiest and arguably the most common stereovision content creation technique. You simply take two physical or virtual cameras and put them next to each other, with their viewing directions precisely parallel. Somehow, this symmetric-*frustum* setup will 'work', as the view of the resulting footage will produce a three-dimensional effect. After a short first-time experience enthusiasm, you will come to realize that the output does not quite produce the desired effect. Looking carefully, you will notice that everything in the scene, up to infinity, appears to float in front of your screen, and this feels wrong. What you are looking for, instead, is for near objects to be floating in front of the screen, and for far objects to be floating behind the screen. Unfortunately, with a parallel set up, there is no way to achieve this. Since the two cameras are 'stereo-focused' at infinity, they can only produce negative horizontal disparity – i.e. you will always perceive objects as if they were in front of the screen. If you want both positive and negative disparity values to result from the rendering process, you must move the stereo-focus plane closer to the cameras set up.

2. *Toe-in stereo*

Playing around with cameras set ups, you will find that a symmetrical *frustum* projection model is anything but flexible. In fact, the best you can do, in order to move the stereo-focus plane away from infinity, is to slightly rotate both cameras inwards. This way, the *frustum* bisectors intersection defines a closer stereo-focus point (not plane). This approach is often called Toe-in stereo, and, again, it 'sort of' works. Toe-in stereo makes sense intuitively. After all, our eyes rotate inwards when we focus on nearby objects[12]. But let not the appearances fool you: the perspective model lying at the basis of three-dimensional computer graphics should be different from the one used by our own eyes. In real sight, the physical world is directly projected onto our retinas, whilst in computer graphics, an intermediate projection screen is used – i.e. the V/AR display. Therefore, the screen orientation, not the eye orientation, should define the perspective projection. Later on, the retinal projection will take care of itself. If you are still in doubt, please have a look at what happens when a

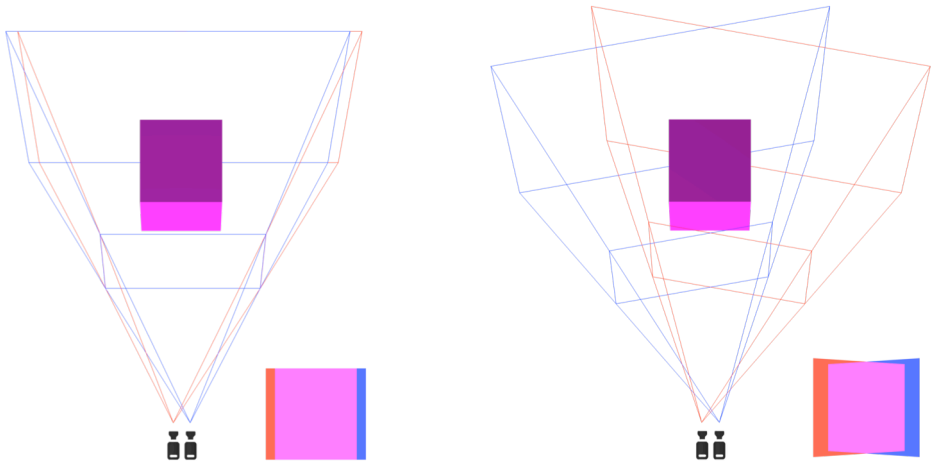

Fig. 8 Comparison between Off-axis stereo (left image) and Toe-in stereo (right image). Respective rendering outcomes are placed side by side.

Toe-in stereo setup is used to render a simple cube. What you see in Fig. 8 (right part) is the render result of the Toe-in rig. As none of the cube's faces is orthogonal to the cameras' viewing directions, a trapezoidal shape is rendered, which looks like a keystoning[7] effect. The two images are then stick together on a single frame, which will be oriented in space perpendicularly to the median line of sight. Since neither the left nor the right view looks as if a real cube was seen through the screen to the naked eye, the Toe-in perspective model fails. This failure leads to serious problems in stereovision, e.g. incorrect depth assessment, inaccurate shape evaluation or three-dimensional illusion breakdown[3]. If you look closely at Fig. 8 (right part) you will notice that the keystoning effect is more severe towards the left and right edges of the image. This is the reason why Toe-in stereo is considered to work 'well enough' around the centre of the screen, whereas the stereoscopic 3D effect breaks down at the edges of the image. No wonder that in Toe-in stereo cinematography two basic rules of thumb are used[12]: first, one should reduce the amount of eye separation and, second, one should keep the action – therefore the audience's eyes – at the centre of the screen. However, these rules are merely workarounds for a problem that would not exist in first place, if stereo were done properly. All in all, Toe-in stereo is only a rough approximation to correct stereo; therefore it should not be used. Even when the keystoning effect is less severe, our eyes will dart around, trying to make sense of the mismatching images, possibly leading to eye strain and headaches[3,12,13]. The fact that Toe-in stereo is seemingly widely used in the three-dimensional industry could (partly) explain the discomfort that many people report while experiencing stereoscopic movies and stereoscopic computer graphics[12]. Remember, a good practice against headaches is the one that treats the cause, not the symptoms.

3. Off-axis stereo

So far, we have showed that conventional Toe-in stereo often leads to depth, shape or layout misperception, resulting in three-dimensional illusion breakdown and serious discomfort for the user. Off-axis stereo uses asymmetric-*frusta* (a.k.a. skewed-*frusta*) to solve these issues. Each *frustum* extends from the corresponding eye to the screen corners positions. With this set up a shared plane exists, which is the plane of the screen, a.k.a. the 'stereo-focus' plane or 'zero-parallax' plane. This is oriented in space like the near (or far) clip planes. Nevertheless, two distinct viewpoints exist. Therefore, while keystoning is avoided, separation is ensured.

All in all, a physical camera rig should always use lens shift, whereas a virtual cameras setup should always use skewed frusta – i.e. off-axis projection.

It stands to reason that, as the viewpoint changes, the perspective model should be modified as well. This is precisely why eye tracking is the only way toward a consistent projection matrix initialization and maintenance. Moreover, this is how one generates binocular disparity signals that are consistent with depth cues coming from motion parallax, instead of being a further cause of visual fatigue for the observer[3]. At this point, the only problem left, associated with stereovision, is the vergence - accommodation issue[12,14], which we will address in the next paragraph.

4. The vergence - accommodation issue

When looking at the real world to the naked eye, the brain will perceive individual objects to be at a certain distance and will tell the eyes to verge and focus (accommodate) their lenses accordingly. Within a certain depth of field, this process helps attaining stereo vision[15]. As a matter of fact, it takes place in real sight as well as in synthetic vision. However, when the latter is involved, lights and shadows reaching the viewer's eyes from either close-by or further away virtual objects will truly come from the V/AR display, which makes the image blurry. In practice, artificial stereovision requires us to focus at one distance and converge at another, which is a problem that four hundred million years of evolution have never presented before[14]. In fact, all the living beings with eyes have always focused and verged at the same point. Unfortunately, there is nothing we can do about this right now, but, at least, it is a rather subtle effect[12].

V. Fields of application

In this section we address suitable aeronautical applications for H-V/AR and F-V/AR. These range from V/AR based simulation facilities, to innovative Human-Computer Interaction (HCI) concepts for manned/unmanned aerial system governance, flight reconstruction and air navigation services provision.

[7]Keystoning is a typical video projection unwanted phenomenon due to the use of a projection surface non-orthogonal to the projection beam.

1. Flight Simulators and Control Tower Simulators

Mainstream flight simulators and control-tower simulators implement neither stereovision nor HCP. Indeed, the 'one-person at a time' nature of H-V/AR and F-V/AR contrasts with a prevalent simulation practice, which is to involve a minimum of two participants in the simulation. Truth is, at the moment, there is no drawback-free solution to this issue. In the future, we might be able to filter multiple perspectives through spatial or temporal encoding, similarly to what we do for stereovision achievement today. For now, given that the problem exists no matter what you do, you might wonder whether you should prefer 'half' of the perspective bias to strike every participant – which is how simulation gets done today – or rather favour one over the other. Although the latter may not seem the best solution, the simulation practice presents circumstances under which 'preferential' H-V/AR or F-V/AR might be the best fit. This is, for example, when a single participant is actually training (or being tested), whereas the other is just delivering credibility to the simulation.

2. Conformal Head Up Displays

Conformal Head Up Displays (C-HUDs) for flight governance are typically hung from the cockpit ceiling or mounted atop the glare shield. Due to perspective biases, C-HUDs can only be used if the pilot's eyes lie inside a certain designated volume (a.k.a. the eye-space), which is typically a small box of few inches in each dimension. If the viewpoint position exceeds the box limits, the erroneous collimation, arising from the perspective bias, becomes unacceptable. For the pilot, to stay still in a pre-defined position can be challenging, especially when he needs to turn his head to look around, looking at the cockpit instrumentation (e.g. the head-down displays). Whenever the downsides of a fixed Head-Up Display (HUD) have been found unacceptable, such as in several military contexts, the problem has been mitigated through the use of Head Worn Head-Up Displays (a.k.a. Head Mounted See-Through Displays), which clearly introduce different types of concerns. For C-HUDs, HCP could be a rather straightforward solution, given that the displayed image could be adapted not only to the aircraft aptitude, but also to the pilots' viewpoint position. The same goes for experimental C-HUDs used in AR control towers. Indeed, Air Traffic Control Operators (ATCOs) are more likely to change their viewing position in control towers than pilots in the cockpit.

3. Synthetic Vision Systems

Like HUDs, Synthetic Vision Systems (SVSs) can be integrated into on-board avionics or Remotely Piloted Aerial Systems (RPAS) Ground Control Stations (GCSs). Synthetic Vision (SV) consists in a real-time, computer-generated image of the topography surrounding the aircraft. This disregards atmospheric occlusion, synthesizing multi-source information into a single, clear and accurate three-dimensional representation of the external environment. Thanks to SV, the pilot maintains excellent ground and airborne situational awareness, even when flying remotely or in adverse conditions – e.g. reduced visibility or darkness. In practice, SV allows the pilot to see through haze, clouds, fog, rain, snow, dust and smoke, while displaying the vehicle's position with respect to the terrain. Advanced SVSs also integrate urban features, obstacles and other significant information, such as flight hazards, flight paths, waypoints, aerodromes, landing points, surrounding facilities (friendly, neutral or hostile), and nearby airspace users. These systems can be dubbed Tactical Synthetic Vision Systems (T-SVSs) instead of simply SVSs. Because SV is completely artificial, aircraft operations can be monitored from either a 'pilot' perspective (egocentric) or an 'out-of-the-cockpit' perspective (exocentric). The latter, is most commonly used in experimental RPAS GCSs. In this case, a gaze-coupled perspective could be used, yielding to a better perspective model and a slightly more flexible FOV.

4. Table Tops and Digital Workbenches for Air Traffic Control and Flight Reconstruction (Virtual Debriefing)

Table Tops (TTs) and Digital Workbenches typically consist in high resolution, head-down, visual interfaces. Often, these are large, multi-touch, interactive displays, which look like glossy tables or rather bulky all-in-one personal computers. Although TTs are becoming increasingly common, few applications exhibit their full potential.

In Ref. 16, a (virtual) TT interface for ATC was conceived and portrayed as useful for balancing traffic amongst airspace sectors, so that saturation could be avoided. In Area Control Centres (ACCs) this task is performed by supervisors and planners, who are responsible for long-term and medium-term conflict avoidance, respectively. Supervisors handle traffic coordination amongst airspace sectors, making sure that no individual sector is overloaded. Doing so, they take weather, current flight regulations and controllers workload into account. On the other hand, planners handle traffic allocation within airspace sectors. They decrease the likelihood of conflicts between aircrafts approaching, leaving, and flying through the sector. For these tasks to be performed, both need a larger picture and a longer-term view than tactical controllers (a.k.a. executor controllers). As a matter of fact, the traffic allocation activity requires controllers to explore and organize traffic from a 'global perspective', rather than to get into exact metric judgments. Because it delivers an intuitive representation of the airspace, the TT prototype was claimed to be capable of improving controllers' performance. Furthermore, such a display was supposed to assist coming on duty ATCOs in understanding complex shapes, such as military restricted airspaces and critical weather fronts. It could be useful to deal with

dynamic re-sectorization as well. Finally, collaborative decision-making can be encouraged providing each controller with a direct interaction tool and fast access to digital information. This could possibly change the entire nature of the ATM task, with multiple ATCOs sharing responsibility for a larger, joined, airspace sector. All in all, such a prototype was a discrete success.

In Ref. 17 a distributed ATC training environment based on 4D flight reconstruction and a TT device was conceived. In Ref. 18 the very same system was claimed to be suitable for pilots training (virtual debriefing) and incident/accident reconstruction. The main goal of such a framework was to make the cognitive process easier and faster during flight data analysis for incident/accident investigation.

These systems, along with others, could better perform if supported by F-V/AR, which would enhance the user engagement by means of a strong depth perception. Moreover, as the viewer moves around, experiencing a fairly realistic 3D effect, such an interface would keep the perspective bias to a minimum.

VI. Conclusion

This paper has provided a sum up of theoretical principles and suitable applications for gaze-dependent HCI. Firstly, essential background on V/AR display techniques has been given, including D-V/AR, O-V/AR, G-V/AR, S-VAR, H-V/AR and F-V/AR. Secondly, pros, cons and constraints of each technique have been exposed. Thirdly, the relationship between ECP and stereovision has been inspected in depth. Finally, a set of suitable aeronautical uses has been identified, including several V/AR-based simulation facilities and groundbreaking HCI concepts for unmanned aerial system governance, flight reconstruction and air navigation services provision.

The dissertation has been kept deliberately conceptual, i.e. unrelated to any specific development framework.

All in all, we hope this paper will contribute to narrow down the gap between the way many V/AR applications work versus how they should really work, i.e. tracking down the viewpoint position to be used within the rendering pipeline.

REFERENCES

1. Smallman, H. S. & St. John, M. Naive Realism: Misplaced Faith in Realistic Displays. *Ergon. Des. Q. Hum. Factors Appl.* **13,** 6–13 (2005).
2. Kooima, R. *Generalized Perspective Projection*. (Lousiana State University, Generalized Perspective Projection, 2009).
3. Solari, F., Chessa, M., Garibotti, M. & Sabatini, S. P. Natural perception in dynamic stereoscopic augmented reality environments. *Displays* **34,** 142 – 152 (2013).
4. Peek, E., Wünsche, B. & Lutteroth, C. Virtual Reality Capabilities of Graphics Engines. in (2013).
5. DeFanti, T. A. *et al.* The StarCAVE, a third-generation {CAVE} and virtual reality OptIPortal. *Future Gener. Comput. Syst.* **25,** 169 – 178 (2009).
6. Liverani, A., Persiani, F. & De Crescenzio, F. An Immersive Reconfigurable Room (I.R.R.) for Virtual Reality Simulation. in *Electronic Proceedings of 12th International Conference on Design Tools and Methods in Industrial Engineering* E1–9 – E1–16 (2001).
7. Bourke, P. D., Hwy, S. & Felinto, D. Q. *Blender and Immersive Gaming in a Hemispherical Dome*.
8. Song Ho Ahn. OpenGL Projection Matrix. *http://www.songho.ca* at <http://www.songho.ca/opengl/gl_projectionmatrix.html>
9. Woo, M., Neider, J., Davis, T. & Shreiner, D. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 1.2*. (Addison-Wesley Longman Publishing Co., Inc., 1999).
10. glFrustum - OpenGL. *www.opengl.org* at <https://www.opengl.org/sdk/docs/man2/xhtml/glFrustum.xml>
11. Ben Lang. VR Expert to Oculus Rift Devs: Make Sure You're Doing 3D Right. *http://www.roadtovr.com* (2013). at <http://www.roadtovr.com/vr-expert-to-oculus-rift-devs-make-sure-youre-doing-3d-right/>
12. Good stereo vs. bad stereo. *Doc-Ok.org* (2012). at <http://doc-ok.org/?p=77>
13. Bando, T., Iijima, A. & Yano, S. Visual fatigue caused by stereoscopic images and the search for the requirement to prevent them: A review. *Displays* **33,** 76 – 83 (2012).
14. Ebert, R. Why 3D doesn't work and never will. Case closed. | Roger Ebert's Journal | Roger Ebert. *rogerebert.com* (2011). at <http://www.rogerebert.com/rogers-journal/why-3d-doesnt-work-and-never-will-case-closed>
15. Cassin, B. & Rubin, M. L. *Dictionary of Eye Terminology*. (Triad Publishing Company, 2001). at <http://books.google.it/books?id=E35sPwAACAAJ>
16. Wong, W., Gaukrodger, S. & Han, F. *Year 1 Prototypes Evaluation (Lot No. 1, WP 2)*. (EUROCONTROL, 2008).
17. Bagassi, S., De Crescenzio, F. & Persiani, F. Design and Development of an ATC Distributed Training System. in *Proceedings of the 26th International Congress of the Aeronautical Sciences including the 8th AIAA 2008 ATIO Conference* (Optimage, 2008).
18. Boccalatte, A., Persiani, F., De Crescenzio, F. & Flamigni, F. A Highly Integrated Graphics Environment for Flight Data Analysis. in *Proceedings of the Joint International Congress 17th INGEGRAF-15th ADM* (2005).