4 Formulation of a custom rendering pipeline

Our first objective is to develop formulas allowing us to compute the parameters of a standard 3D perspective projection matrix (*l*, *r*, *b* and *t*) based on the relative position and orientation between the viewer's eyes and the screen. In order to constantly feed these parameters to the projection matrix a constant link between the tracking system and the projection matrix is needed. This is depicted in Fig. 8. Also it is mandatory to know the exact transformation between the tacker-space coordinate system and the world space-coordinate system. However, this can be easily determined once the location and the orientation of the tracking device(s) are fixed and known.

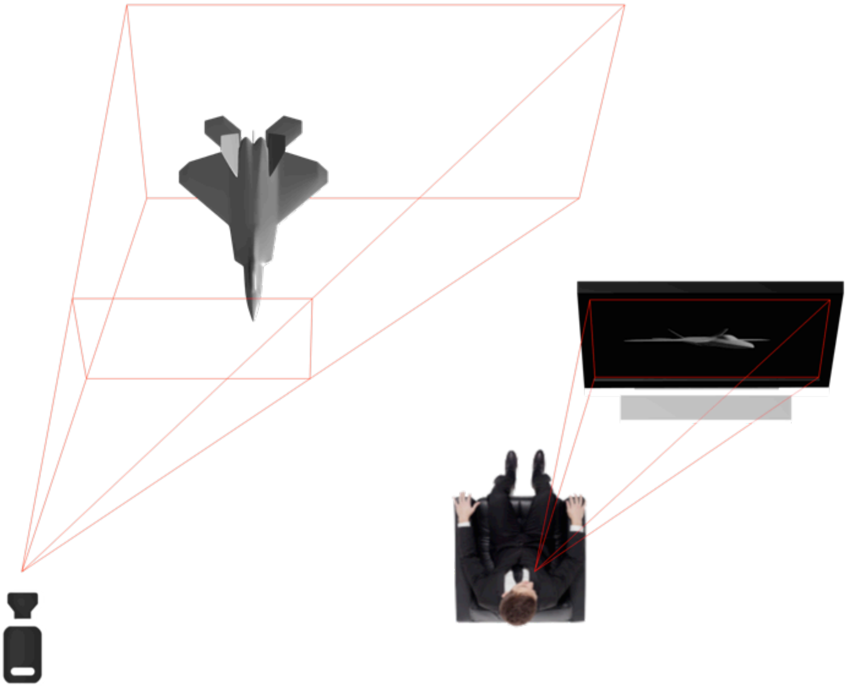

Fig. 3. The skewed frustum projection model, a.k.a. off-axis projection model.

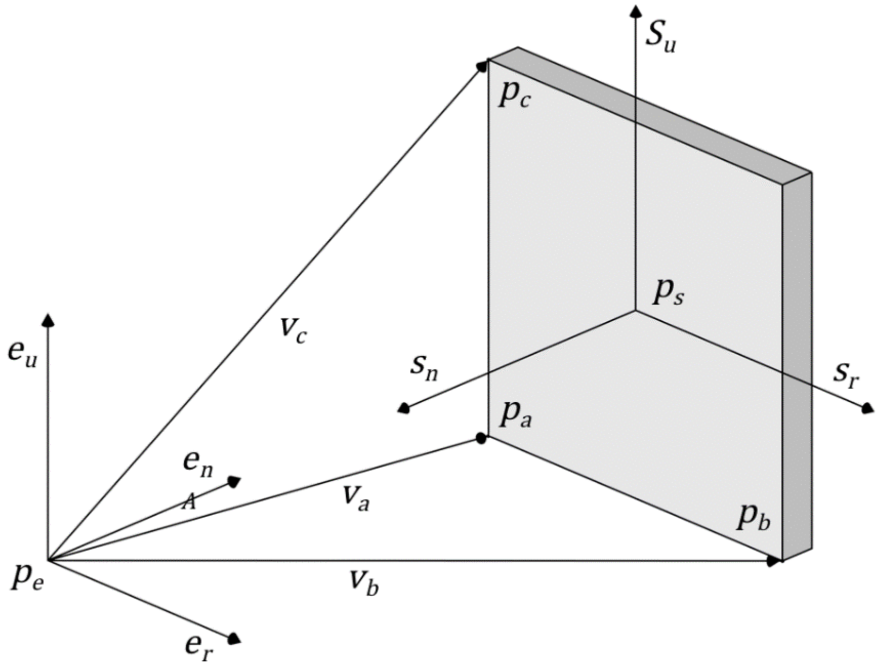

Fig. 4. The eye-space coordinate system (origin p_e), the screen-space coordinate system (origin p_s) and the screen corners vectors v_a, v_b and v_c.

Let's start reviewing the main characteristics of the AR system. These are the coordinates of the display corners, the origin of screen-space coordinate system, and the distance from the eye-space coordinate system origin to the screen (Fig. 5 and 6).

The coordinates of the head-up display corners, namely p_a (lower left corner), p_b (lower right corner), and p_c (upper left corner) are expressed with respect to world-space coordinate system. Assuming that flat screen is used, the position of the fourth point is implicit. Together these points encode the size of the screen, its aspect ratio, its position and orientation. Also, they can be used to compute an orthonormal basis for the screen-space coordinate system[4]. We refer to this basis as the triad of vectors composed by s_r (the vector toward the right), s_u (the vector pointing up), and s_n (the vector normal to the screen, pointing in front of it).

---

[4] In linear algebra an orthonormal basis for an inner product space is a basis whose vectors are all unit vectors orthogonal to each other.

$$s_r = \frac{p_b - p_a}{|p_b - p_a|} \qquad (3)$$

$$s_u = \frac{p_c - p_a}{|p_c - p_a|} \qquad (4)$$

$$s_n = \frac{s_r \times s_u}{|s_r \times s_u|} \qquad (5)$$

The origin of the screen space coordinate system is the intersection between the perpendicular line drawn from p_e to the screen, and the plane of the screen itself. Since neither $\underline{p_e}$ nor p_s are fixed in space, when the viewer moves with respect to the screen, the screen-space origin changes accordingly. If s/he moves far to the side of the screen, then the screen space origin may not fall within the screen at all.

The distance from the eye-space origin p_e and the screen-space origin p_s may be computed by taking the dot product of the screen normal v_n with any of the screen vectors. However, because these vectors point in quite opposite directions, their product must be negated.

$$d = -(s_n \cdot v_a) \qquad (6)$$

In order to compute the frustum extents we need the vectors from the camera space origin (p_e) to the screen corners. Once again, these can be easily calculated using the screen corners.

$$v_a = p_a - p_e \qquad (7)$$

$$v_b = p_b - p_e \qquad (8)$$

$$v_c = p_c - p_e \qquad (9)$$

Frustum extents may be interpreted (and computed) as distances from the screen-space origin to the edges of the screen (as shown in Fig. 6). However, because these are not specified at the near clipping plane, we must scale them back from their value at the plane of the screen, d units away from the eye-space origin, to their value at the near clipping plane, n units away from the eye-space origin.

$$l = \frac{(s_r \cdot v_a)\, n}{d} \qquad (10)$$

$$r = \frac{(s_r \cdot v_b)\, n}{d} \qquad (11)$$

$$b = \frac{(s_u \cdot v_a)\, n}{d} \qquad (12)$$

$$t = \frac{(s_u \cdot v_c)\, n}{d} \qquad (13)$$

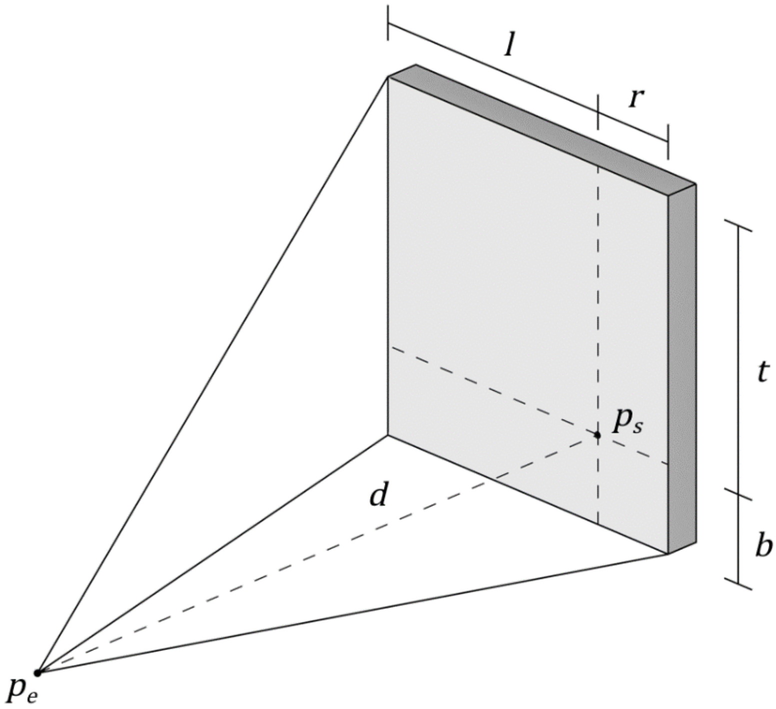

Fig. 5. The length of the frustum extents (l, r, b and t) at the plane of the screen.

Inserting these values into the standard perspective projection matrix allows us to build a head-tracked (or eye-tracked), off–axis projection (Fig 7). In other words, we now have the ability to create a skewed frustum for an arbitrary screen viewed by an arbitrary ‘eye’.

There is one final limitation that we must work past, and that is that the near clipping plane is orthogonal to the eye-space depth axis (e_n), whereas the plane of the screen may not be. In practice, we need to free the projection plane from the orientation of the e_u-e_r plane. Unfortunately the way the standard perspective projection matrix was built simply disallows this. What we can do instead is to rotate the virtual world in order to line up the desired projection plane with the e_u-e_r orientation. As far as the projection outcome is concerned this is equivalent to rotating the viewing frustum aligning the near clipping plane to the plane of the screen. Note that this operation does not affect the frustum extents calculation.

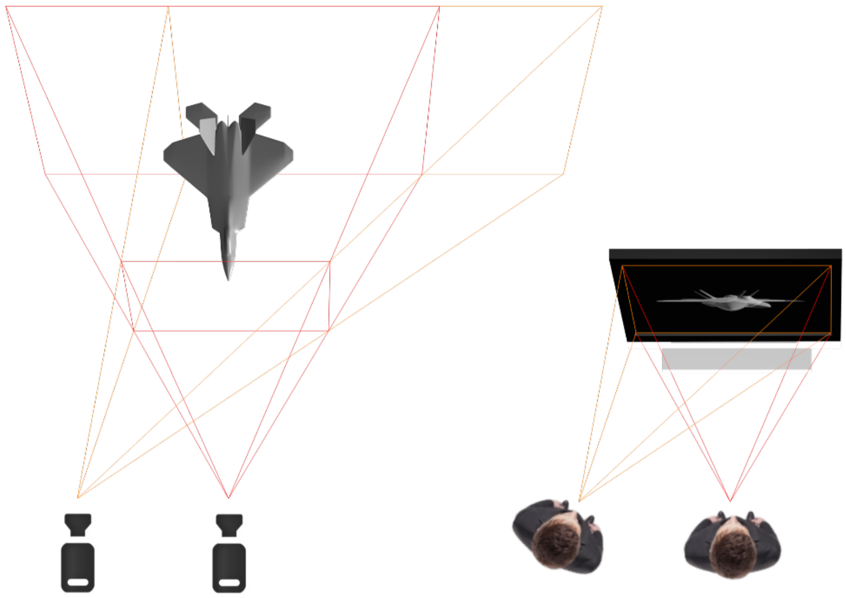

Fig. 6. Example of a head-tracked off-axis perspective.

We can build a transformation matrix that rotates the eye-space coordinate system so that its standard axis e_r, e_u and e_n match the orientation of the screen-space coordinate system like so:

$$R = \begin{bmatrix} s_{rx} & s_{ux} & s_{nx} & 0 \\ s_{ry} & s_{uy} & s_{ny} & 0 \\ s_{rz} & s_{uz} & s_{nz} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (14)$$

Which can be easily deduced from:

$$R \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = s_r \qquad (15)$$

$$R \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = s_u \qquad (16)$$

$$R \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = s_n \qquad (17)$$

If something lies on the e_u-e_r plane, this transformation will align it to the plane of the screen. However, what we really need is the inverse mapping:

$$R^{-1}s_r = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{18}$$

$$R^{-1}s_u = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \tag{19}$$

$$R^{-1}s_n = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \tag{20}$$

Which is produced by the inverse of R:

$$R^{-1} = \begin{bmatrix} s_{rx} & s_{ry} & s_{rz} & 0 \\ s_{ux} & s_{uy} & s_{uz} & 0 \\ s_{nx} & s_{ny} & s_{nz} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{21}$$

However, since R is orthogonal, R^{-1} is simply its transpose:

$$R^{-1} = R^T \tag{22}$$

Applying this transformation to all the objects in the virtual world will rotate the scene until the plane of the screen lines up with the e_u-e_r plane, which is exactly we needed. If we compose the standard projection matrix P with the rotation matrix R^T, the resulting matrix M covers everything we need and will work under any circumstances.

$$M = R^T P \tag{23}$$

With this matrix, we are finally able to render the scene in order to generate the AR layered.

5 Conclusion

Various analysts have estimated the benefits of using AR tools in control tower operations. However it is unclear which one between the head-worn AR technology and the spatial AR technology will prevail. Also, it has been rarely specified how such tools should be designed and operated. Our review confirms that many problems must be addressed before these tools become operational. However, there is ample reason to believe that, eventually, this will happen.

With regard to the use of very large format AR displays, we have developed an advanced rendering pipeline that is capable of generating registered overplays for multiple, arbitrary oriented head-up displays, defined together in a common coordinate system (Fig 8). Our concept is based on the excellent work by Robert Kooima (Electronic Visualization Laboratory, University of Illinois) [39]. However, because of a different target and development framework, the here-proposed implementation turned out to be a quite different thing. For instance, we do not consider the eye-space coordinate system to be bound to the world-space coordinate system (which probably seems quite reasonable to game engine developers).

In this document, the algorithm description has been deliberately mathematical, (i.e. not linked to any specific development framework or programming language). However, at our facilities, we have developed (and tested) a Python/C# code using an open source game engine and a Kinect^{TM} for Windows^{TM} tracking sensor. We plan to use this software as a basis for further research in the field of spatial see-through HUDs for the control tower.

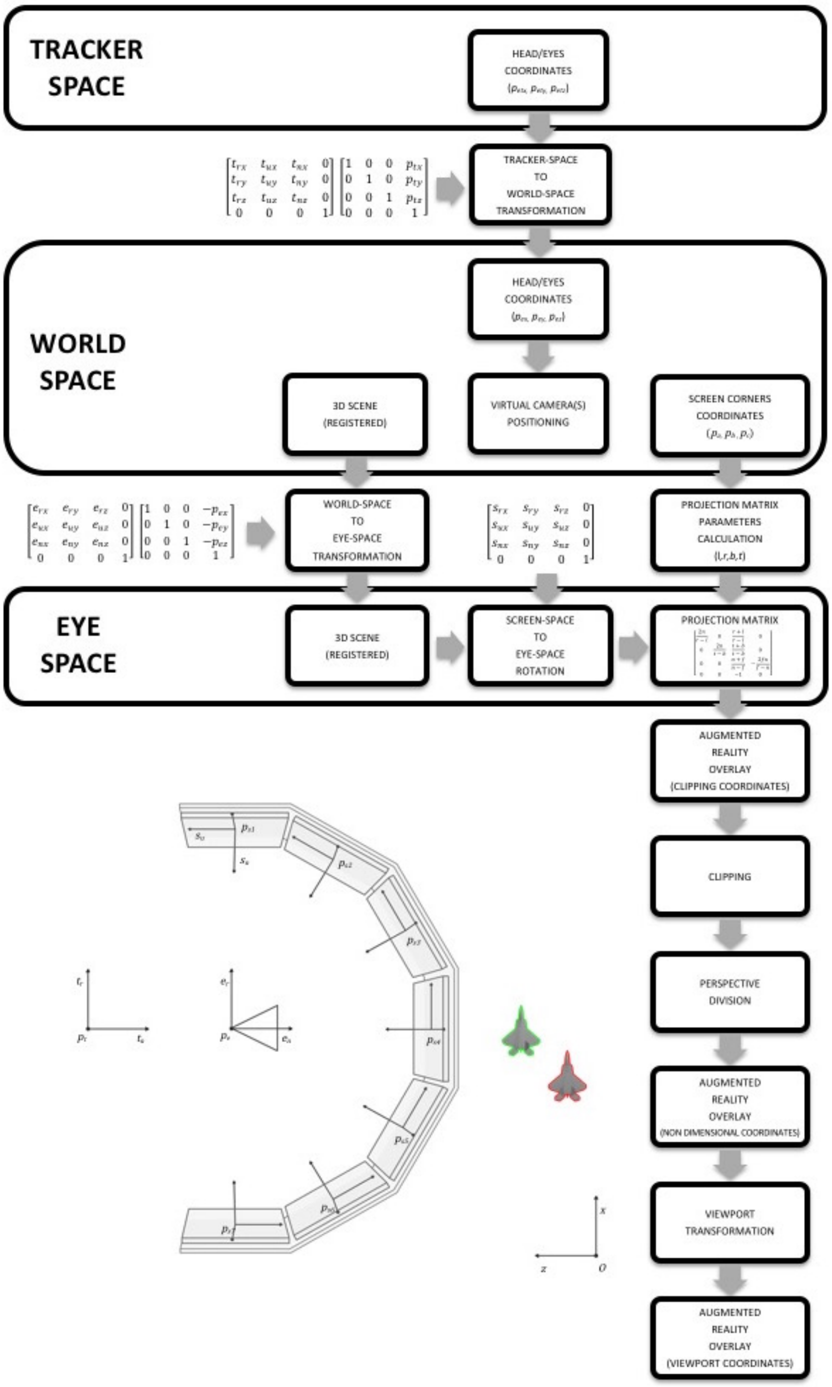

Fig. 7. An overall schematics of the modified rendering pipeline

References

1. Office for Official Publications of the European Communities: Single European Sky - Results from the transport research programme, http://www.transport-research.info/sites/default/files/brochure/20040617_111440_74437_singlesky.pdf.
2. SESAR JU: Enhanced Ground Controller Situational Awareness in all Weather Conditions, https://www.atmmasterplan.eu/oisteps/AO-0201.
3. Clean Sky JU: Clean Sky 2 - Joint Technical Programme, http://ec.europa.eu/research/participants/data/ref/h2020/other/guide-appl/jti/h2020-guide-techprog-cleansky-ju_en.pdf.
4. SESAR JU: Honeywell in SESAR: Advancing ATM in the heart of Europe, http://www.sesarju.eu/sites/default/files/documents/reports/SESAR_magazine_issue_7.pdf, (2012).
5. Reisman, R., Brown, D.: Design of Augmented Reality Tools for Air Traffic Control Towers. In: Proceedings of 6th AIAA Aviation Technology, Integration and Operations Conference (ATIO). American Institute of Aeronautics and Astronautics, Wichita, KS (2006).
6. Monica Tavanti: Control Tower Operations: A Literature Review of Task Analysis Studies. EEC Technical/Scientific Report No. 2006-013 (2006).
7. Ella Pinska: An Investigation of the Head-up Time at Tower and Ground Control Positions. In: Proceeding of the 5th Eurocontrol Innovative Research Workshop. pp. 81–86 (2006).
8. Schmidt, M., Rudolph, M., Werther, B., Möhlenbrink, C., Fürstenau, N.: Development of an Augmented Vision Video Panorama Human-Machine Interface for Remote Airport Tower Operation. In: Smith, M.J. and Salvendy, G. (eds.) HCI (9). pp. 1119–1128. Springer (2007).
9. Ellis, S.R.: Towards Determination of Visual Requirements for Augmented Reality Displays and Virtual Environments for the Airport Tower. In: Virtual Media for Military Applications RTO-MP-HFM-136. pp. 31–1 – 31–10. RTO, Neuilly-sur-Seine, France (2006).
10. Shackelford, R, Karpe, P.: Low/Zero Visibility Tower Tools (L/ZVTT) Benefits Assessment. , NASA Ames Research Center Technical Reports, Moffett Field, CA 94035 (1998).
11. Neveu, C., Blackmon, T., Stark, L.: Evaluation of the Effects of a Head-mounted Display on Ocular Accommodation. Presence. 7, 278–289 (1998).
12. Stephen R. Ellis, Bernard D. Adelstein, Ronald J. Reisman, Joelle R. Schmidt-Ott, Jonathan Gips, Jimm Krozel: Augmented Reality in a Simulated Tower Environment: Effect of Field of View on Aircraft Detection. NASA Ames Research Center Technical Reports (2002).
13. Weintraub, D.J., Ensing, M.J.: Human factors issues in head-up display and design: the book of HUD. Crew System Ergonomics Information Analysis Center, Wright-Patterson AFB, Ohio (1992).
14. Herman Rediess: An augmented reality pilot display for airport operations under low and zero visibility conditions. In: Guidance, Navigation, and Control Conference. American Institute of Aeronautics and Astronautics (1997).
15. Ruffner, J.W., Deaver, D.M., Henry, D.J.: Requirements analysis for an air traffic control tower surface surveillance enhanced vision system. Presented at the SPIE-The International Society for Optical Engineering (2003).
16. N. Fürstenau: Virtual Tower. 5th ATM R&D Symposium, Braunschweig. (2005).
17. Azuma, R.: A Survey of Augmented Reality. In: Presence: Teleoperators and Virtual Environments 6. pp. 355–385 (1997).

18. Reisman, R.J., Feiner, S.K., Brown, D.M.: Augmented Reality Tower Technology Flight Test. In: International Conference on Human-Computer Interaction in Aerospace (HCI-Aero). , Santa Clara, CA (2014).
19. Ronald J. Reisman, David M. Brown: Augmented Reality Tower Technology Assessment. NASA Ames Research Center Technical Reports (2010).
20. De Crescenzio F., Bagassi S., Fantini M., Lucchi F.: Virtual Reality based HUD (Head Up Display) to simulate 3D conformal symbols in the design of future cockpits. Presented at the Council of European Aerospace Societies, Venice, Italy (24-28 October, 2).
21. Sara Bagassi, Francesca De Crescenzio, Francesca Lucchi, Franco Persiani: Innovation in Man Machine Interfaces: Use of 3D Conformal Symbols in the Design of future HUDs (Head-Up Displays). Presented at the 28th International Congress of The Aeronautical Sciences, Brisbane, Australia September 23 (2012).
22. SESAR JU: System Wide Information Management (SWIM), http://www.sesarju.eu/sesar-solutions/swim.
23. SESAR JU: SWIM Concept of Operations, https://www.eurocontrol.int/sites/default/files/publication/files/del08.01.01-d41-swim_conops.pdf.
24. SESAR JU: Time Based Separation, http://www.sesarju.eu/sesar-solutions/airport-integration-and-throughput/time-based-separation.
25. SESAR: 06.09.03 Remote & virtual TWR, https://www.atmmasterplan.eu/projects/16484.
26. N. Fürstenau, M. Schmidt, M. Rudolph, C. Möhlenbrink, A. Papenfuß, S. Kaltenhäuser: Steps Towards the Virtual Tower: Remote Airport Traffic Control Center (RAiCe). Presented at the ENRI International Workshop on ATM/CNS, Tokyo, Japan (2009).
27. D. Schulz-Rückert: Future of Aerodrome Traffic Control. Presented at the Aachen Aviation Convention, Aachen, Germany (2007).
28. Bimber, O., Raskar, R.: Spatial Augmented Reality: Merging Real and Virtual Worlds. A. K. Peters, Ltd., Natick, MA, USA (2005).
29. Peterson, S.: Very large format stereoscopic head-up display for the airport tower. In: In Proceedings of the Virtual Images Seminar, number 16. CNRS/Renault (2007).
30. Peterson, Stephen, Ella Pinska: Human Performance with simulated Collimation in Transparent Projection Screens. In: Proceedings of the Second International Conference on Research in Air Transportation (ICRAT) (2006).
31. Rolland, J.P., Baillot, Y., Goon, A.A.: A Survey of Tracking Technology for Virtual Environments. Fundamentals of Wearable Computers and Augmented Reality. 67–112 (2001).
32. Zhou, F., Duh, H.B.-L., Billinghurst, M.: Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR. In: Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on. pp. 193–202 (2008).
33. Nicola Masotti, Franco Persiani: Gaze-Coupled Perspective for Enhanced Human-Machine Interfaces in Aeronautics. Presented at the Conferences in Air Transport & Operations, Delft University of Technology, Delft, the Netherlands. July 20 (2015).
34. OpenGL: gluPerspective, https://www.opengl.org/sdk/docs/man2/xhtml/gluPerspective.xml.
35. Song Ho Ahn: OpenGL Projection Matrix, http://www.songho.ca/opengl/gl_projectionmatrix.html.
36. 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., Boston, MA, USA (1999).

37. Microsoft: DirectX Graphics and Gaming, https://msdn.microsoft.com/en-us/library/windows/desktop/ee663274(v=vs.85).aspx.
38. glFrustum - OpenGL, https://www.opengl.org/sdk/docs/man2/xhtml/glFrustum.xml.
39. Kooima, R.: Generalized Perspective Projection. J. Sch. Electron. Eng. Comput. Sci. (2009).
40. 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. pp. E1–9 – E1–16, Rimini, Italy (2001).