Interactive Media, Gaming, and VR-AR Technologies

NKFN77

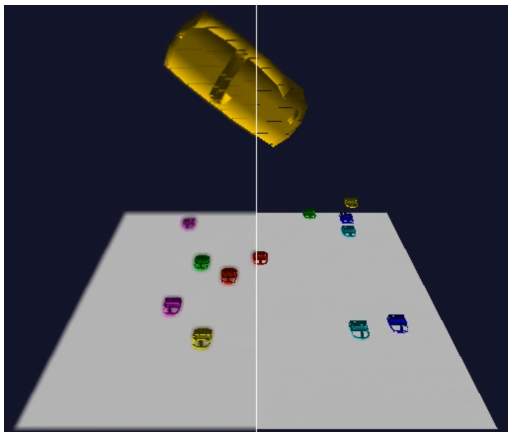

Fig. 1. Perspective projection implemented in the main scene

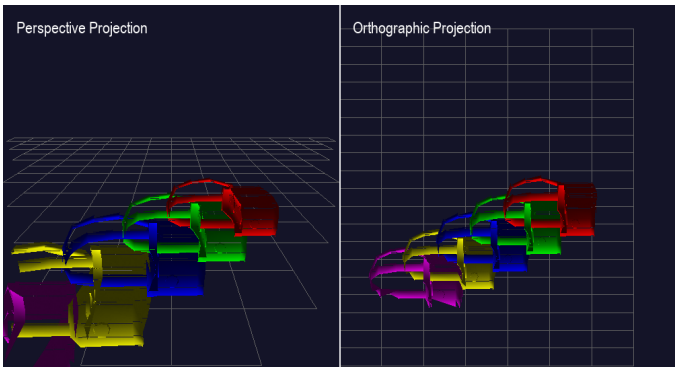

Fig. 2. Perspective and Orthographic projection

1 Simulation

1.0.1 Perspective projection

Perspective projection creates a realistic view of the scene and objects within it. Orthographic projection means that objects remain the same size regardless of the distance; it provides no sense of depth to the scene. Figure 1 shows the implemented perspective projection, with floor headsets further away from the camera appearing smaller to the viewer. Figure 2 shows a scene with headsets being rendered with orthographic and perspective projection, showing the need for perspective projection to add depth to the rendered scene.

2 Pose calculation: Dead Reckoning Filter

2.0.1 DRF: Gyroscope-only

A dead reckoning filter (DRF) estimates the current position and orientation of an object based on a previously known position combined with measurements of speed, heading and elapsed time. In the case of the rendering pipeline, gyroscopic data from the IMU dataset is used for angular velocity which are integrated over time to determine changes in direction. A pure gyroscopic DRF creates a latency-free simulation which is highly responsive to rotational movements which makes it particularly suitable for IMU datasets. In short-term renderings, the orientation is accurate but over time it begins to drift after accumulating small errors which are compounded over time. Error sources come from:

1) Bias error (constant offset) [1]
2) Scale factor errors (proportional to true angular rate)
3) Random walk noise (statistical variance in the signal) [2]

Due to simplicity in design, the computational efficiency of a gyroscope-only DRF is low compared to that of an enhanced DRF with accelerometer an magnetometer considerations. With only gyroscopic measurements to work with and no reference to gravity, there can be a loss in orientation of the headset to the ground plane over time. Figure 3 shows clear gyroscopic drift that occurs throughout a fully rendered scene, with

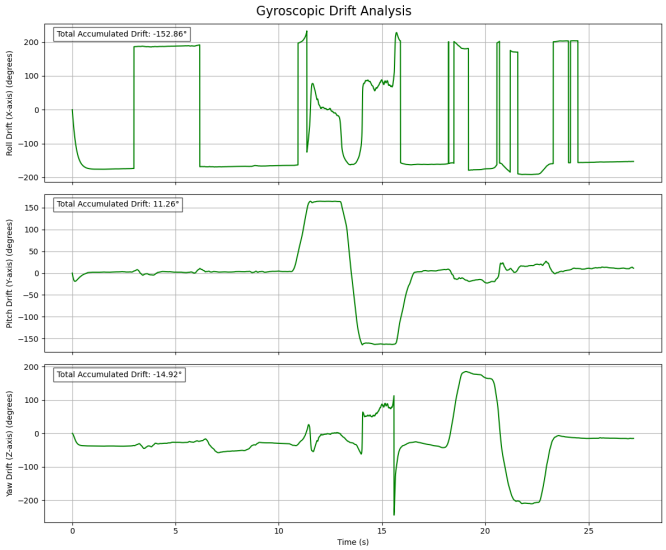

Fig. 3. Gyroscopic drift analysis

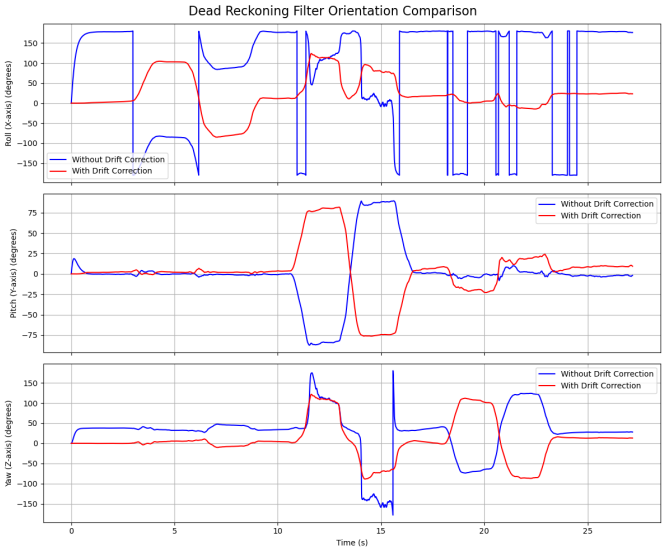

Fig. 4. Gyroscopic and accelerometer drift analysis

most drift occurring around the x-axis. The video titled `headset_simulation_gyro_only` shows the fully tracked motion of the headset with gyroscopic correction only.

2.0.2 DRF: Gyroscope and accelerometer fusion

A solution to gyroscopic drift is to incorporate accelerometer data, which factors in the direction of gravity and forces acting on the headset in all directions. The provided absolute reference compensates for the accumulated drift experienced by the gyroscope, providing long-term stability. Reference to gravity maintains a correct orientation relative to the ground plane, and bias is corrected in the long run. However, there is a significant computational difference to using accelerometer data as it requires double integration to obtain position, which adds a significant overhead to rendering compared to gyroscope-only DRFs. Alongside the computational overhead, the accelerometer causes the headset to be sensitive to all forces which makes it difficult to isolate gravity alone. This can lead to noise being introduced into rendering from small movements. Figure 4 shows the accelerometer data that decrease the drift accumulated in the simulation, but there is still accumulation that could be improved with a more sophisticated DRF approach. The alpha value, ranging from 0 to 1, determines the weighting towards gyroscopic over accelerometer data. A value closer to 1 gives more weighting to the gyroscope, and a value closer to 0 gives more weighting to the accelerometer. As the value gets closer to 0, the accelerometer data starts to add noise to the

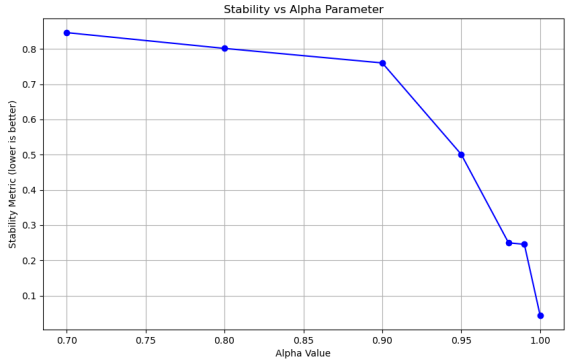

Fig. 5. Dead reckoning filter alpha value Vs simulation stability

simulation, causing the headset to jitter more aggressively. As the value gets closer to 1, the headset appears to be smoother, but drift accumulates over time and causes the headset to rotate unnecessarily towards the end. The experimentation of different values of alpha is shown in Figure 5, with lower values introducing large amounts of instability to the system (measured by a stability value closer to 1).

2.0.3 DRF: Gyroscopic fusion, accelerometer and magnetometer fusion

A limitation to DRF with only gyroscopic and accelerometer data is that the accelerometer only detects linear forces that provide a reference to the vertical plane. As the gravity vector remains unchanged around the yaw axis, rotations cannot be corrected from accelerometer readings around the gravity vector. The solution to this is to factor

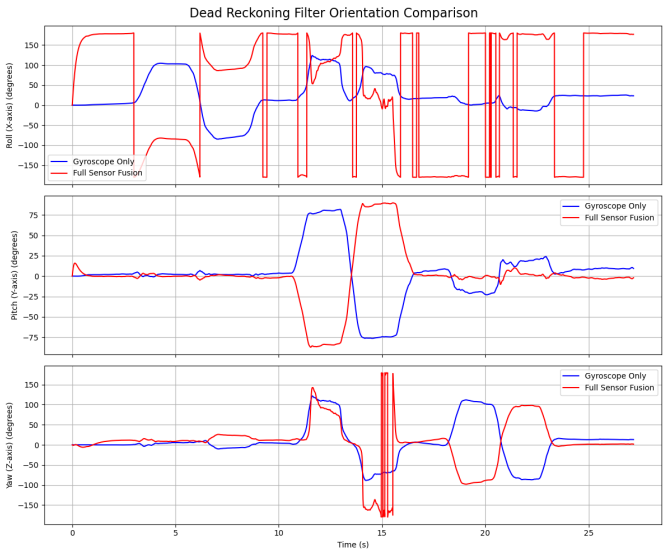

Fig. 6. Dead reckoning filter gyroscope, accelerometer, and magnetometer fusion

in the magnetometer readings present in the IMU data which provides a reference to Earth's magnetic field. This reduces drift around the Z-axis as there will be yaw correction. Combining accelerometer, magnetometer and gyroscopic readings means all axes will have drift correction, providing a stable heading over time. An adaptive buffer was implemented to aid with correction efforts which is used by the engine to average stored samples for each axes to determine if the headset is rotating. This is used in noise correction, and impacts the sensitivity the headset has to noise spikes. The adaptive method aids more bias to the gyroscope during rotation, and more bias to the accelerometer/magnetometer during stationary periods for drift correction. Figure 6 shows the correction from a full fusion approach which shows all axes have drift correction applied to them. The video titles `headset_simulation_sensor_fusion` shows the fully tracked motion of the headset with gyroscopic, accelerometer, and magnetometer correction.

2.0.4 Computational cost of different DRF implementations

A pure gyroscopic implementation of the DRF has the lowest computational cost of all implementations. A total of 35 operations are required per sample from the IMU dataset to process a gyroscopic DRF. The methods where these operations occur are outlined in Table 1, giving a total of 35 computations per sample for the gyroscope-only DRF. Table 2 shows the extra operations when the accelerometer and magnetometer data is incorporated into the DRF, totalling between 250-300 operations per sample including the 35 samples from the gyroscopic operations. The range of operations depends on a variety of different pathways in the accelerometer and magnetometer. These pathways occur if correction for an axis occurs in the particular frame or for when correction rates need to be adjusted. Table 3 shows the operations related to the rotation state buffer, a feature used by the adaptive alpha control.

2.0.5 Positional tracking

Positional tracking is possible by double integrating the accelerometer data present in the IMU dataset. But, the drift error rate can grow quadratically due to the double integration compared to the linear growth of the gyroscopic drift rate which is integrated only once. A solution to this issue in general application is detection of zero-velocity which can mitigate position drift, but this is not accessible with IMU data. Sample frequency can also be increased to improve simulation accuracy, but this adds significant computational overhead to the calculations [3].

3 Physics

3.0.1 Collision detection

Figure 7 shows the collision spheres created around each headset that were used to detect collisions between the headsets and the boundary of the scene in the fully rendered scene. A collision between two spheres was handled with two main methods: 1. `check_collision` and 2. `resolve_collision`. These two methods allowed for a headset to detect a collision between the scene boundary or another headset's sphere, then based off the collision normal and elasticity coefficients applied to each object, the velocities of each object would then be updated. In practice, these collisions appeared like in the attached video titled `collision` which shows four headsets moving close to each other and rebounding based off the angle of collision between them. A more sophisticated collision system could be implemented with a multi-spherical method where different parts of the headset model would have spheres assigned to them. For example, a sphere would be assigned to the front of the headset, to the left hand strap, to the right hand strap, and to the rear strap. This would grant further coverage of the headset, and allow for different collision normals and elasticity coefficients to be implemented on the

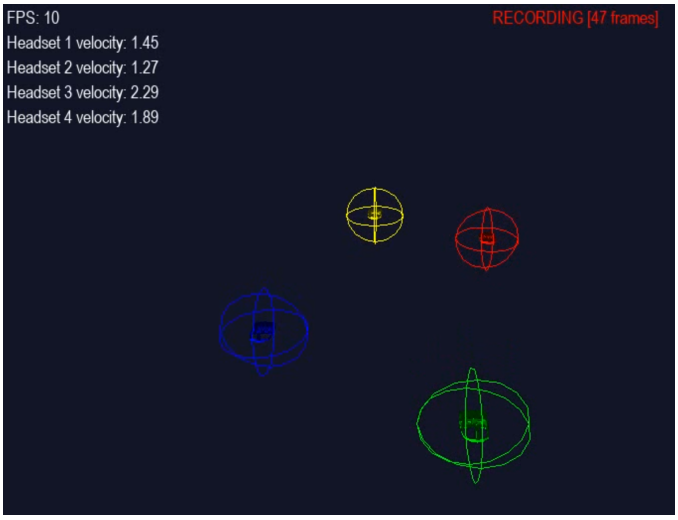

Fig. 7. Scene showing four headsets with draw collision spheres

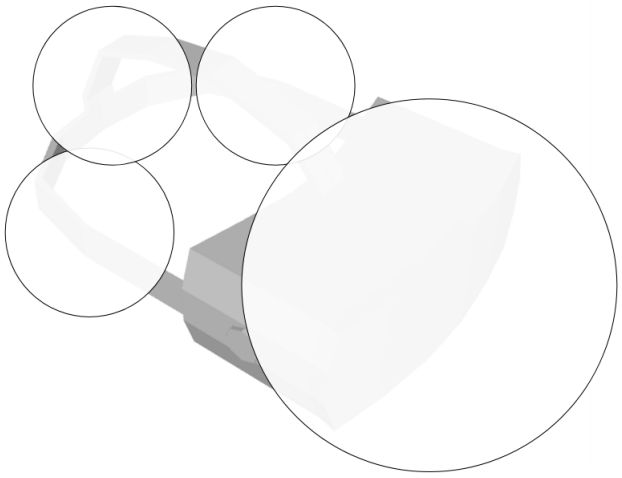

Fig. 8. Improved collision detection spheres for the headset object

headset. For example, a higher elasticity coefficient could be assigned to the front of the headset as this would have an impact in real-life that would cause the objects to bounce off each other. However, a hard object hitting the soft straps at the side or back would cause little to no rebound force as the strap would absorb most of the impact, leading to the rebounding headset to have little velocity in the opposing direction. The collision detection described would increase the computation required by a factor of 4 due to the number of spheres going from 1 to 4, which is a negligible performance difference as it is $O(n)$. Figure 8 shows an example of the design described above.

3.0.2 Friction

Friction was implemented by a velocity dampener coefficient which was applied to a floor headsets horizontal (`x` and `z` axes) velocity during the simulation. The friction coefficient is applied to the `x` and `z` velocity components during the simulation, causing a headsets horizontal velocity to decrease over time. Equations 1 and 2 show the application of the friction coefficient to the velocity components of an object in the simulation.

$$Vx' = \mu \cdot Vx \tag{1}$$

$$Vz' = \mu \cdot Vz \tag{2}$$

4 Post-processing

4.0.1 Depth-of-field

Depth-of-field was implemented to distinguish headsets further away from the camera to be blurred to create a sense of depth. Figure 1 shows the headsets furthest away from the camera to be blurred more than headsets closer to the camera on the left hand side. A version with no depth of field at all is show on the right hand side. Monocular depth cues were used to create a blur effect with distance, mimicking human eye functionality.

The depth-of-field filter utilises three parameters:

1) Focal distance (D_f): Where objects appear perfectly sharp
2) Focal range (R_f): Range around focal distance where objects remain relatively sharp
3) Blur strength (B_s): Global multiplier for blur intensity

A piecewise function is applied to each pixel's Z-buffer value. Equation 3 shows the blur effect applied to pixels within the focal range and equation 4 shows the blur effect applied to pixels outside the focal range. The effect this produces is a continuous blur transition with gradual falloff after the focal range, becoming more pronounced beyond it. The blur effect is generated based off a 2D Gaussian kernel, where the final object colour is calculated based off the weighted average of pixels in the neighbourhood [4]. D_p is the measurement of the pixel's depth from the camera according to the Z-buffer

$$Blur(Dp) = Bs \times \frac{|Dp - Df|}{Rf} \times 0.5 \tag{3}$$

$$Blur(Dp) = Bs \times \left(0.5 + \min\left(1.0, \frac{|Dp - Df| - Rf}{Rf}\right)\right) \tag{4}$$

Depth-of-field allows for a viewers brain to naturally prioritise headsets closer to the camera, allowing the engine to bring a users attention to certain parts of the scene which can increase user engagement [5]. Overall, the blur gradient makes the scene more photo-realistic, and immerses the user more into the scene than the non-implemented approach seen on the right hand side of Figure 1 [6].

REFERENCES

[1] Ian Beavers. The case of the misguided gyro — analog devices, 2017.

[2] Brian Douglas. Fusing a mag, accel, and gyro to estimate orientation — understanding sensor fusion and tracking, part 2, 2019.

[3] Kurt Seifert and Oscar Camacho. Implementing positioning algorithms using accelerometers, 2007.

[4] P. Rokita. Generating depth of-field effects in virtual reality applications. *IEEE Computer Graphics and Applications*, 16:18–21, 03 1996.

[5] Pingping Wen, Fei Lu, and Ahmad Zamzuri. Using attentional guidance methods in virtual reality laboratories reduces students' cognitive load and improves their academic performance. *Virtual reality*, 28, 05 2024.

[6] Saeed Safikhani, Vinzenz Gattringer, Michael Schmied, Johanna Pirker, and Selina Christin Wriessnegger. The influence of realism on the sense of presence in virtual reality: Neurophysiological insights using eeg. *Multimodal Technologies and Interaction*, 8:104–104, 11 2024.

Operation	Computation cost per sample
Bias correction	3 subtractions
Vector creation	1 object allocation
Quaternion	1 object allocation
Quaternion multiplication	16 multiplication / addition operations
Scale quaternion derivative	4 multiplications
Integration	4 multiplications and adds
Normalisation	5 operations + 1 square root

TABLE 1
Total operations for gyroscopic DRF, in order of method

Operation	Computation cost per sample
Bias and scale correction	6 operations
Magnitude calculation	4 operations + 1 square root
Adaptive alpha calculation	10 conditional and arithmetic
Vector normalisation	3 divisions
Quaternion transformations	2 quaternion multiplications (32 operations)
Vector normalisation and operations	15 operations
Cross product calculation	9 operations
Dot product and angle	8 operations + acos
Correct quaternion	10 operations + sin/cos
Apply correction	16 operations + normalisation
Similar vector processing	20 operations
Transform to global frame	2 quaternion multiplications (32 operations)
Projection calculations	15 operations
Heading calculations	20 operations + trig functions
Heading correction	15 operations + trig
Apply heading correction	16 operations + normalisation
SLERP quaternion	30 operations + trig functions

TABLE 2
Total operations for gyroscopic, accelerometer, and magnetometer DRF, in order of method

Current rotation calculation	4 operations + square root
Average calculation	5N operations
State transition logic	5 conditional operations

TABLE 3
Total operations for the rotational state tracking where N is the buffer size (Set to 10 by default), in order of method