

Gliederung der Abschlussarbeit: Netcode-Mechaniken in Echtzeitanwendungen

Alexander Seitz

29. April 2025

Inhaltsverzeichnis

1	Einleitung	3
1.1	Motivation	3
1.2	Zielsetzung der Arbeit	3
1.3	Aufbau der Arbeit	3
2	Theoretische Grundlagen	4
2.1	Netzwerke in Echtzeitanwendungen	4
2.1.1	Latenz, Paketverlust, Jitter, Tickrate	4
2.2	Architekturen in Multiplayer-Systemen	4
2.2.1	Client-Server-Modell	4
2.2.2	Peer-to-Peer-Architektur	4
2.2.3	Authority-Konzepte: Client vs. Server	4
2.3	Begriffserklärungen	4
2.3.1	Tickrate und Input Delay	4
2.3.2	Reconciliation, Prediction, Interpolation	4
3	Netcode-Mechaniken im Detail	5
3.1	Client-Side Prediction	5
3.1.1	Funktionsweise	5
3.1.2	Vorteile und Risiken (Rubberbanding, Desyncs)	5
3.2	Interpolation	5
3.2.1	Zielsetzung: Flüssige Darstellung	5
3.2.2	Techniken: LERP, Snapshot Buffering	5
3.3	Lag Compensation	5
3.3.1	Rewind-Mechaniken bei Treffererkennung	5
3.3.2	Trade-offs: Fairness vs. Komplexität	5

4	Umsetzung des Prototyps in Unity	6
4.1	Projektstruktur und Designentscheidungen	6
4.1.1	Aufbau der Zielumgebung	6
4.1.2	Verzicht auf automatische Synchronisation	6
4.2	Implementierung der Mechaniken	6
4.2.1	Client-Side Prediction beim Schießen	6
4.2.2	Interpolation beweglicher Objekte	6
4.2.3	Lag Compensation beim Hit-Scan	6
4.3	Technische Herausforderungen	6
4.3.1	Simulation von Latenz und Paketverlust	6
4.3.2	Fehlerquellen und Debugging	6
4.4	Visualisierungstools	6
4.4.1	Overlays: Prediction, Lag, Interpolation	6
5	Evaluation und Experimente	7
5.1	Testaufbau	7
5.1.1	Szenarien mit simulierten Netzwerkbedingungen	7
5.2	Beobachtungen	7
5.2.1	Einfluss auf Spielgefühl und Kontrolle	7
5.3	Vergleich verschiedener Konfigurationen	7
5.3.1	An/aus-Schalten von Prediction und Interpolation	7
5.3.2	Unterschiedliche Latenzstufen	7
5.4	Diskussion	7
5.4.1	Stärken und Schwächen der Mechaniken	7
5.4.2	Relevanz je nach Anwendungsszenario	7
6	Übertragbarkeit auf andere Anwendungsbereiche	8
6.1	Weitere Echtzeitanwendungen	8
6.1.1	Beispiel: Kollaborative Musiksoftware	8
6.2	Anpassung der Mechaniken	8
6.2.1	Prediction vs. Genauigkeit	8
6.2.2	Verschiebung der Prioritäten: Qualität vs. Reaktionszeit	8
7	Fazit und Ausblick	9
7.1	Zusammenfassung	9
7.2	Ausblick	9
7.2.1	Mögliche Erweiterungen (z.B. Rollback, KI)	9

1 Einleitung

1.1 Motivation

1.2 Zielsetzung der Arbeit

1.3 Aufbau der Arbeit

2 Theoretische Grundlagen

2.1 Netzwerke in Echtzeitanwendungen

2.1.1 Latenz, Paketverlust, Jitter, Tickrate

2.2 Architekturen in Multiplayer-Systemen

2.2.1 Client-Server-Modell

2.2.2 Peer-to-Peer-Architektur

2.2.3 Authority-Konzepte: Client vs. Server

2.3 Begriffserklärungen

2.3.1 Tickrate und Input Delay

2.3.2 Reconciliation, Prediction, Interpolation

3 Netcode-Mechaniken im Detail

3.1 Client-Side Prediction

3.1.1 Funktionsweise

3.1.2 Vorteile und Risiken (Rubberbanding, Desyncs)

3.2 Interpolation

3.2.1 Zielsetzung: Flüssige Darstellung

3.2.2 Techniken: LERP, Snapshot Buffering

3.3 Lag Compensation

3.3.1 Rewind-Mechaniken bei Treffererkennung

3.3.2 Trade-offs: Fairness vs. Komplexität

4 Umsetzung des Prototyps in Unity

4.1 Projektstruktur und Designentscheidungen

4.1.1 Aufbau der Zielumgebung

4.1.2 Verzicht auf automatische Synchronisation

4.2 Implementierung der Mechaniken

4.2.1 Client-Side Prediction beim Schießen

4.2.2 Interpolation beweglicher Objekte

4.2.3 Lag Compensation beim Hit-Scan

4.3 Technische Herausforderungen

4.3.1 Simulation von Latenz und Paketverlust

4.3.2 Fehlerquellen und Debugging

4.4 Visualisierungstools

4.4.1 Overlays: Prediction, Lag, Interpolation

5 Evaluation und Experimente

5.1 Testaufbau

5.1.1 Szenarien mit simulierten Netzwerkbedingungen

5.2 Beobachtungen

5.2.1 Einfluss auf Spielgefühl und Kontrolle

5.3 Vergleich verschiedener Konfigurationen

5.3.1 An/aus-Schalten von Prediction und Interpolation

5.3.2 Unterschiedliche Latenzstufen

5.4 Diskussion

5.4.1 Stärken und Schwächen der Mechaniken

5.4.2 Relevanz je nach Anwendungsszenario

6 Übertragbarkeit auf andere Anwendungsbereiche

6.1 Weitere Echtzeitanwendungen

6.1.1 Beispiel: Kollaborative Musiksoftware

6.2 Anpassung der Mechaniken

6.2.1 Prediction vs. Genauigkeit

6.2.2 Verschiebung der Prioritäten: Qualität vs. Reaktionszeit

7 Fazit und Ausblick

7.1 Zusammenfassung

7.2 Ausblick

7.2.1 Mögliche Erweiterungen (z.B. Rollback, KI)

Anhang

A.1 Code-Snippets

A.2 Screenshots und Visualisierungen

A.3 Testdaten

Literaturverzeichnis