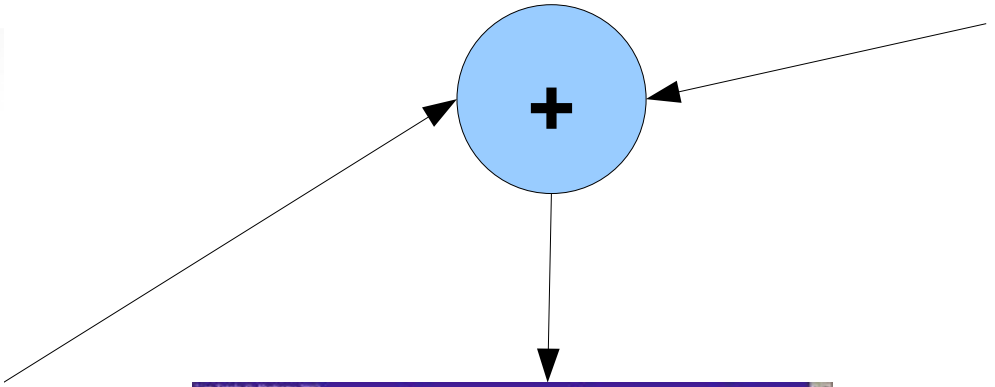
Wii Tetris GL

- Christian Benjamin Ries
- Informationstechnik / Datenverarbeitung
  - Vortrag über die Projektarbeit für die Vorlesung **”Grafische Datenverarbeitung”**, Prof. Dr. math. Bunse.

Die Idee

Wiimote



Computer

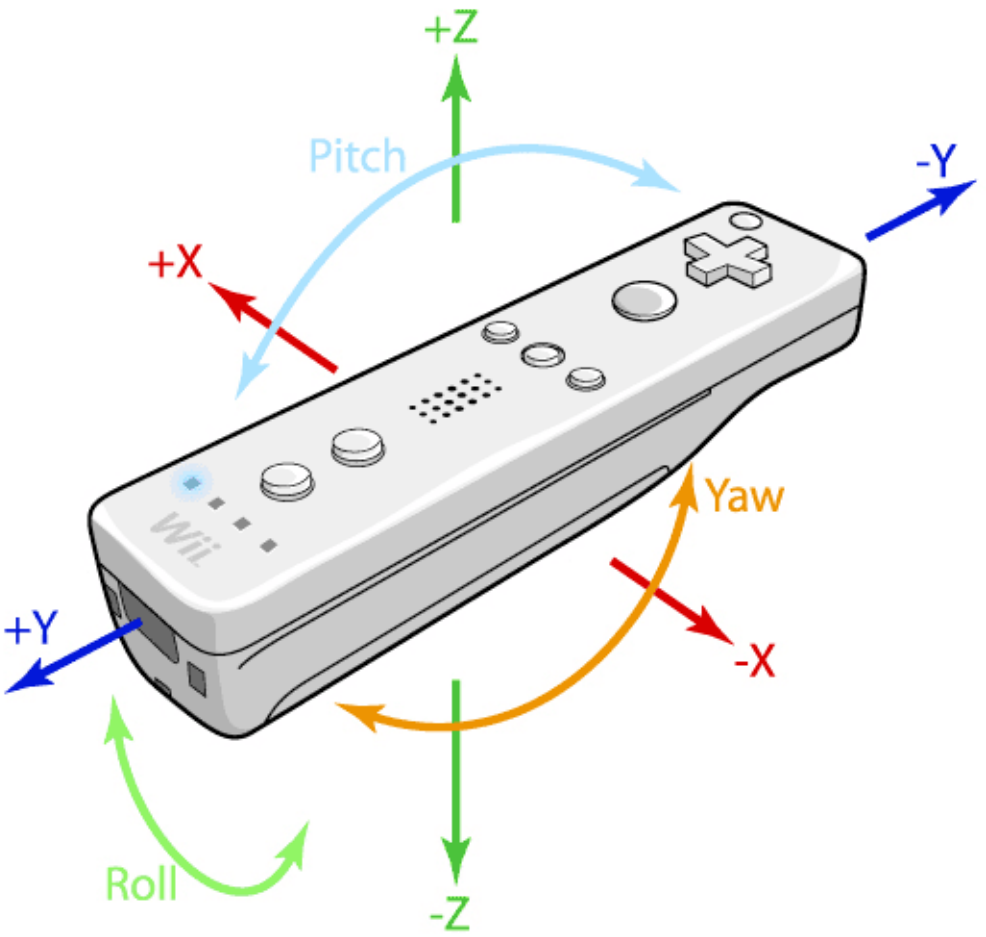
Resultat:

Wii Controller (Wiimote) steuerbare/spielbare Version einer Tetris Implementierung für den heimischen Computer.

Wii Tetris GL

- Was ist die Wii?
  - fernsehgebundene Spielkonsole von Nintendo
  - seit Ende 2006 verfügbar
  - Steuerung der Spielaktivitäten über eine Art Fernbedienung mit Bewegungssensoren
  - mehr als 25 Millionen verkaufte Spieleinheiten bis Ende 2007

Wii Tetris GL



- Frei wählbare Bewegungsrichtung (X-, Y- und Z-Achse).
- Frei wählbare Rotationsrichtung (Pitch, Roll, Yaw).

Wii Tetris GL

- Frei wählbare Bewegungsrichtung (X-, Y- und Z-Achse).
- Frei wählbare Rotationsrichtung (Pitch, Roll, Yaw).

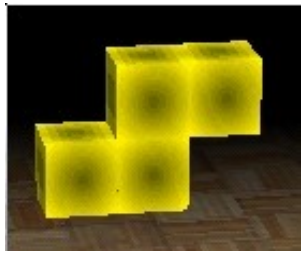
Zudem:

- **Präzisionssteuerung** durch Steuerungskreuz, lässt 360° Drehungen zu.

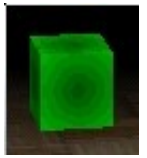
Wii Tetris GL

- Was ist Tetris?
  - Ein Computerspiel, bei dem von oben fallende Quadranten passend angeordnet werden müssen.
  - Bei gefüllter Reihe wird diese entfernt und oberhalb befindliche rücken nach unten.
  - Die Punktzahl ist vom Setzen der Steine abhängig.
  - Existiert in zahlreichen Varianten (Dr. Mario, Triptych, Blockout, TetriNET).

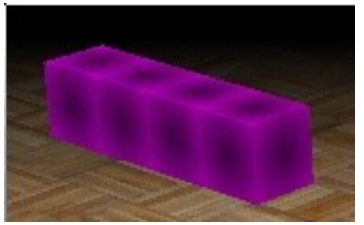
Die Spielsteine



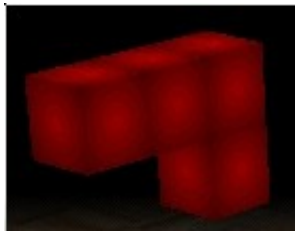
kurvenförmige Struktur



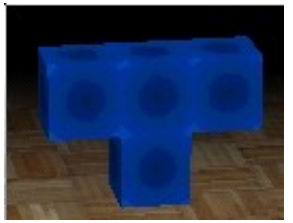
einzelner Stein



Linie an Steinen



einfaches Rechteck



pyramidenähnliche Struktur

→ Hinzufügen von weiteren Steinen mit einer geringen Anzahl an Zeilen Quelltext möglich!

-.- Spielvorführung -.-

Wii Tetris GL

- Konzepte
  - Steine hinzufügen, → Anzeigen
  - Programmablauf, → Anzeigen
  - Linien entfernen, → Anzeigen
  - Kollisionserkennung, → Anzeigen

Wii Tetris GL

Hinzufügen eines neuen Bausteins: (siehe Datei **forms.h → Anzeigen**)

```
/** Die Einsen geben an, an welcher Position, sich ein Kubus (Cube) befindet.
 */
static int __rechteck[] = {
	0, 0, 1,
	1, 1, 1
};

enum FORMSID { FRM_RECT };

/** Struktur, die alle Bausteine beinhaltet.
 */
static struct forms {
	int * frm;
	int cols;
	int rows;
	int depth;
	cColor color;
	FORMSID id;
	int points;
} forms[1] = {
	 { __rechteck, 3, 2, 1, cColor(COLOR_RED), FRM_RECT, 50 }
};
```

Wii Tetris GL

Hinzufügen eines neuen Bausteins:

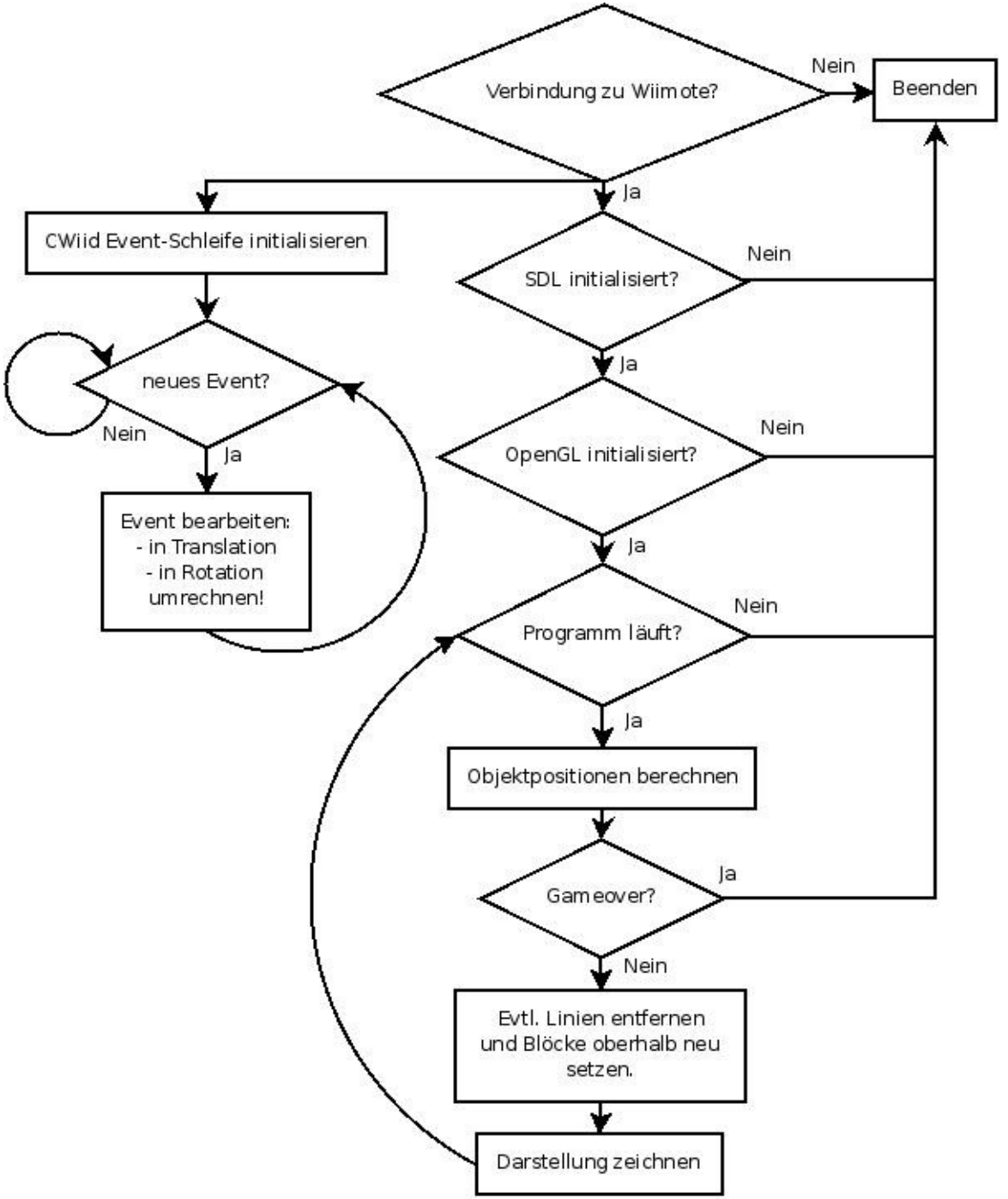
--> zurück!

```
/** Hinzufuegen des oben definierten Bausteins.
 */
Tetris::glObjectMatrix baustein;

struct forms * f = &forms[0];

baustein.position() = cVec<float>(0.0f, 20.0f, 0.0f);

baustein.setMatrix( f->frm, f->cols, f->rows,
                        f->depth, f->color, f->id, 0 );

baustein.rotateX(90.0f);
baustein.rotateY(90.0f);
baustein.rotateZ(90.0f);

baustein,setCubeSectorArea(
      cVec<float>( PLAYGROUND_STARTX, PLAYGROUND_STARTY, PLAYGROUND_STARTZ ),
      cVec<float>( PLAYGROUND_STARTX, PLAYGROUND_STARTY, PLAYGROUND_STARTZ ),
      1.0f );
```

Wii Tetris GL

Statistik

- Hauptspeichernutzung ca. 1,7 MB
- CPU Ausnutzung z.Z. bei 100%
  (wird dran gearbeitet!!!)
- Grafikkartenspeicherausnutzung
  unbekannt!

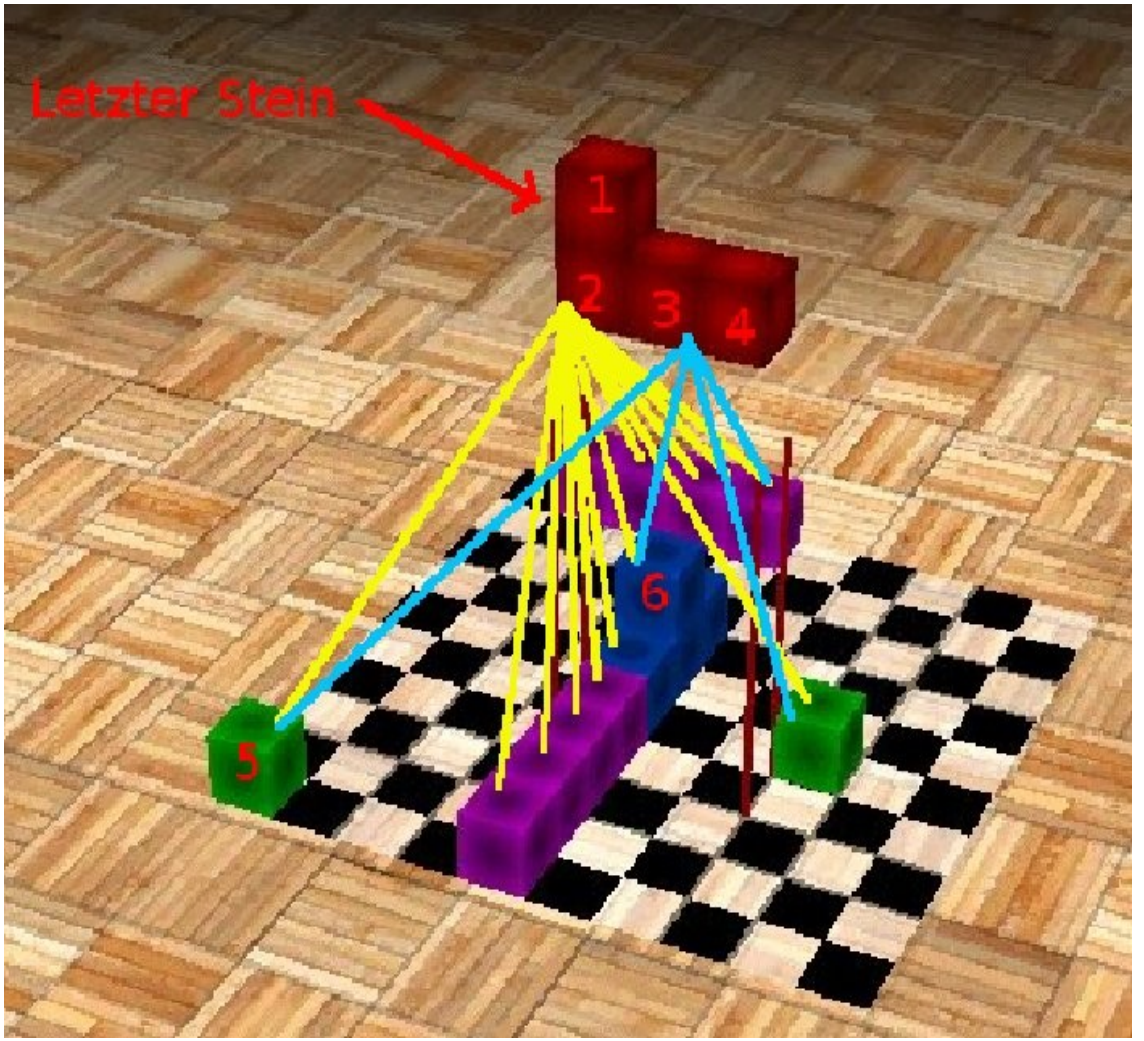


--> zurück!

- **Überprüfung auf Linien**
  - benötigt viel Rechenzeit
  - bei einem Raum von der Dimension 10*10*20-X, der komplett gefüllt ist, sind ca. **4.000.000** Überprüfungen notwendig!!! (diese Situation kann nicht vorkommen, da nur der letzte Stein mit allen Übrigen überprüft wird!)
  - **'Anzahl der Steine' * (10^2 * 20 – 1)** max. Überprüfungen
  - Rechenaufwand steht im Verhältnis zu der Anzahl an gesetzten Bausteinen.

Wii Tetris GL

- Nur der letzte Stein wird mit allen anderen Steinen überprüft!
- Stein X wird auf Gleichheit mit den Achsen der anderen Steine auf der Y-Z und Y-X Achse geprüft.
- Bei Gleichheit Einreihung in einen Vektorcontainer und bei der Anzahl von 10 Elementen wird die Reihe entfernt.
- Die Steine über dieser Linie werden N Zeilen nach unten gesetzt.

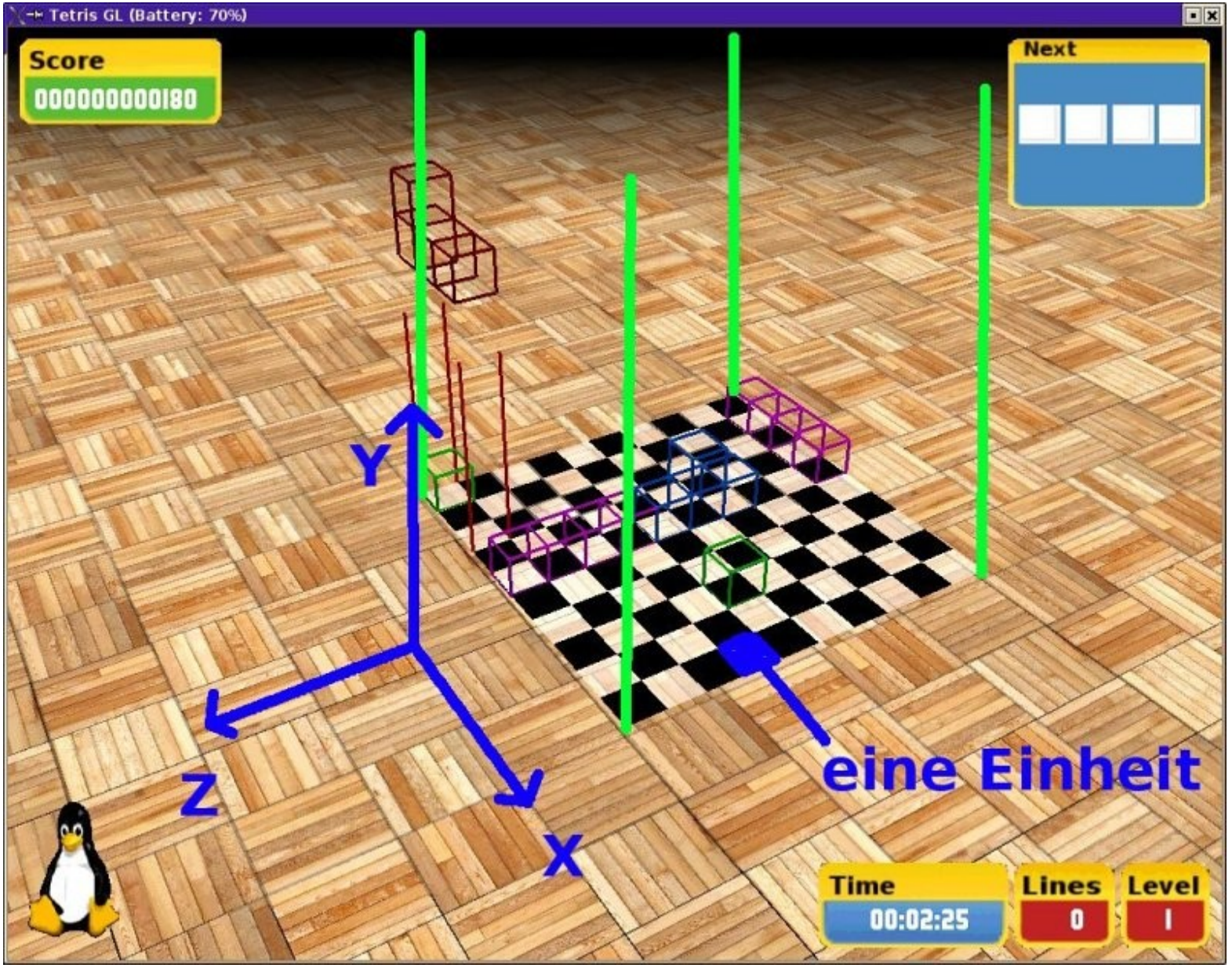


Quelltext anzeigen!

--> zurück!

- **Kollisionserkennung, Voraussetzung**
  - Die Steine fallen FPS unabhängig, damit auf unterschiedlichen Rechnern keine Geschwindigkeitsunterschiede existieren.
  - Dadurch unterschiedliche Schrittweite der Steine.
  - Eingrenzung der Bewegung auf einen kleinen Raum.
  - Schritte auf der X und Z Achse +- 1 Einheit.

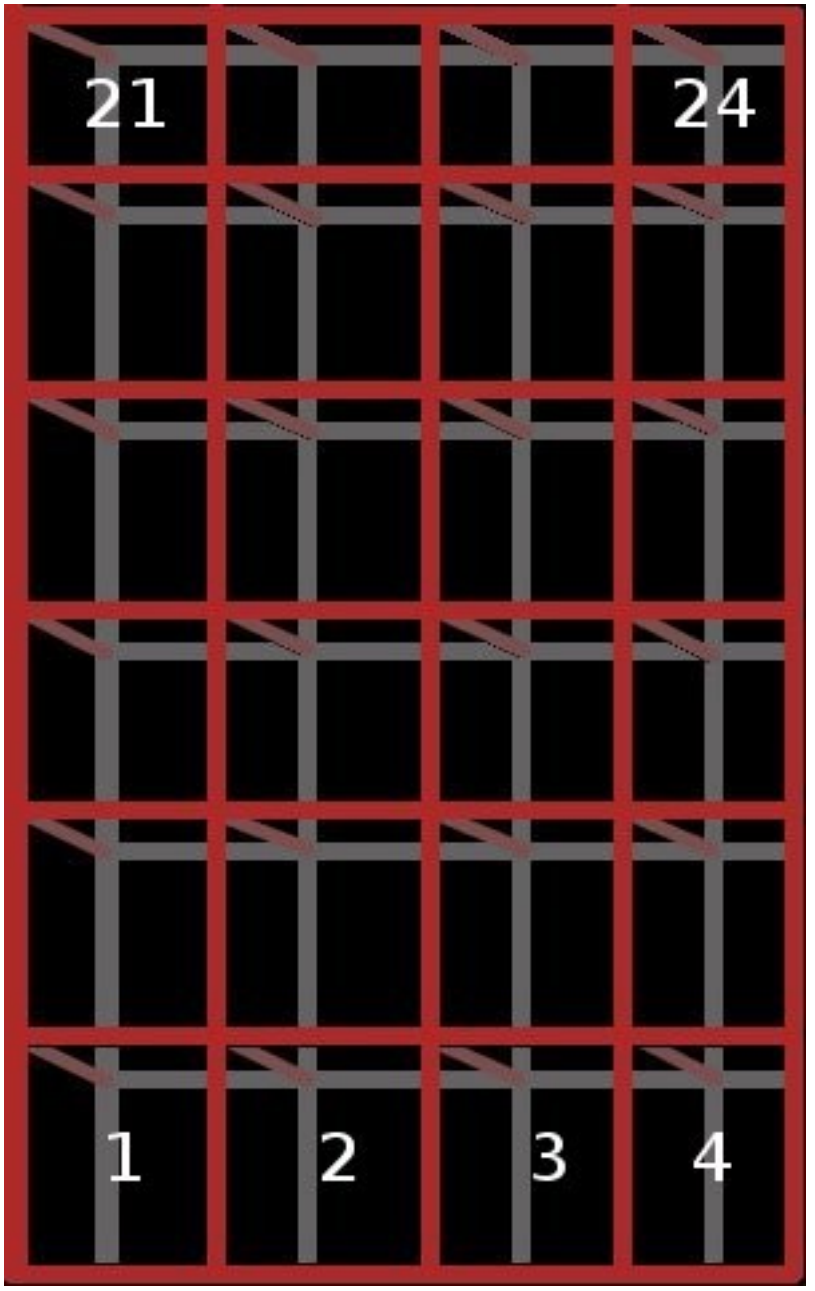
Wii Tetris GL



- **Kollisionserkennung, Implementierung**
- **auf der Y-Achse!**
  - wenn sich ein Stein auf der Y-Achse eines anderen befindet und der kleinste Y-Wert des runterfallenden Steines kleiner als der größte Y-Wert des anderen ist, dann sind diese Steine miteinander kollidiert!

Wii Tetris GL

- **Kollisionserkennung, Voraussetzung**
- **auf der X- und Z-Achse**
  - Der Raum wird in Sektoren geteilt.
  - Die Maße eines Sektors sind 1x1x1.
  - Jeder Sektor besitzt eine eindeutige ID.



- **Kollisionserkennung, Implementierung**
- **auf der X- und Z-Achse**
  - Nach jedem Schritt der Bausteine werden die Sektoren neu berechnet.
  - Beim Bewegen des Bausteins vom Benutzer werden die Sektoren von der nächsten Stelle überprüft.
  - Bei gleichen Sektoren findet eine Kollision statt und es folgt keine Bewegung.

- **Probleme bei der Entwicklung**
  - **Rundungsfehler**
    - 0.5f == 0.5f, diese Bedingung ist nicht immer erfüllt!
    - 0.1f Rundungsfehler beachten!
  - **Regel der großen 3**
    - Kopierkonstruktor, Destruktor und überladender Zuweisungsoperator
    - Alle drei müssen implementiert sein!

Wii Tetris GL

- Verbesserungswürdig:
  - Berechnungsaufwand
  - Speichernutzung
  - Fallbewegung
  - Steuerung (z.B. Kalibrierung)
  - Spiellogik (z.B. Multiplayer, Fun-Blocks ”Explosion” oder ”Randomize-Mode”)

Danke für Ihre Aufmerksamkeit!

Fragen?