



VEKTORIA-MANUAL SOUND



Prof. Dr. Tobias Breiner

vektoria

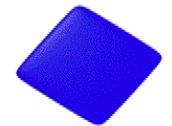
www.vektoria-engine.com



Sound

Inhalt

////SOUND ////ALLGEMEIN



////AMBIENT ////SOUND

////SOUND ////METHODEN

3D ////SOUND

////UEBUNGEN ZU ////SOUND



Sound

Wichtigkeit von Sound



Der stiefmütterlich behandelte Gehörsinn

Fallout-Teaser mit Sound



Der stiefmütterlich behandelte Gehörsinn

Fallout-Teaser ohne Sound



Sound

Sound Ratespiel



Sound

Sound Ratespiel



Sound in Szenegrafen

Sound ist normalerweise nicht Bestandteil von Szenegrafen. Schließlich behandelt ein Szenegraf – wie der Name schon sagt – lediglich die grafischen Aspekte einer Szene.

Allerdings macht es durchaus Sinn, Szenegrafen mit anderen Empfindungen zu erweitern:

- Force Feedback (Tastsinn, Vibration)
- Motion Bases (Gleichgewichtssinn)
- Sound & Musik (Gehörsinn)
- O-Devices (Olfaktorik)





CAudio

Ambient Sound

Hintergrundklang, der immer gleich laut erschallt, egal wo die Kamera steht.

Gut geeignet für:

- musikalische Untermalung
- akustische Benutzerführung

3D Sound

Raumklang der beliebig positioniert werden kann. und aus der entsprechenden Richtung erschallt. Je näher die Position dem Sound der Kamera, desto lauter. Gut geeignet für:

- tönende 3D-Objekte
- Effektklänge .





Abfrage-Methoden

Beide, sowohl Ambient- als auch 3D-Sound, werden durch eine einzige Klasse repräsentiert, sie heißt **CAudio**.

Eine Routine kann bei initialisiertem Sound feststellen, um was für eine Art Sound es sich handelt:

```
bool Is3D();    // Gibt true bei  
                // einem 3D-Sound aus,  
                // ansonsten false
```





CAudio

Ambient Sound

Hintergrundklang, der immer gleich laut erschallt, egal wo die Kamera steht.

Gut geeignet für:

- musikalische Untermalung
- akustische Benutzerführung

3D Sound

Raumklang, der beliebig positioniert werden kann. und aus der entsprechenden Richtung erschallt. Je näher die Position dem Sound der Kamera ist, desto lauter. Gut geeignet für:

- tönende 3D-Objekte
- Effektklänge



Init-Methode für Ambient Sound

Funktion gibt true aus, falls die Initialisierung erfolgreich war, ansonsten false

Initialisiert einen ambienten Klang

```
bool Init
```

```
(
```

```
char *stringwavFile,
```

Pfad zu wav- oder mp3-Datei

```
HWND hwnd,
```

Handle auf das aktuelle Fenster

```
);
```





Tick und Fini-Methoden

Finalisiert einen Sound, egal ob
Ambient Sound oder 3D-Sound.



```
void Fini();
```

Tick braucht man nicht aufzurufen, weder bei Ambient Sound (wo kein Tick benötigt wird) noch bei 3D-Sound (hier wird die Methode automatisch aufgerufen, wenn der Sound in die Szenegrafenhierarchie eingefügt wird.)





Gemeinsame Methoden für 3D- und Ambient Sound

Start-, Loop- und Stop-Methoden



```
void Start(); // Startet einen Klang,  
              // spielt ihn einmal ab  
  
void Loop();  // Startet einen Klang,  
              // spielt ihn immer wieder ab  
  
void Stop();  // Beendet einen  
              // gestarteten Klang
```





Regulationsmethoden

```
void SetVolume(float frVolume);  
// Modifiziert die Lautstärke  
// (0.0F=aus 1.0F=volles Rohr)
```

1

2

3

4

5





Sound

3D-Sound in Vektoria



CAudio

Ambient Sound

Hintergrundklang, der immer gleich laut erschallt, egal wo die Kamera steht.

Gut geeignet für:

- musikalische Untermalung
- akustische Benutzerführung

3D Sound

Raumklang der beliebig positioniert werden kann. und aus der entsprechenden Richtung erschallt. Je näher die Position dem Sound der Kamera, desto lauter. Gut geeignet für:

- tönende 3D-Objekte
- Effektklänge





Spezielle Methoden für 3D-Sound

Init Methode für 3D-Sound



Funktion gibt true aus, falls die Initialisierung erfolgreich war, ansonsten false

Initialisiert einen Punktsound mit 3D-Raumklang

```
bool Init3D  
(  
    char *stringWavFile,  
    float fRadius  
);
```

Pfad zu wav- oder mp3-Datei

Radius der max. Soundreichweite










Spezielle Methoden für 3D-Sound

Spezielle 3D-Sound Methoden



1 
`void SetRadius(float fRadius);`
`// Modifiziert den Wirkradius`
`// in Units eines 3D-Klantes`

2 
3 
4 
5 
`void SetDoppler(float fFactor);`
`// Setzt Dopplerstärke`
`// fFactor=1.0: physikalisch richtiger`
`// Doppler-Effekt (bei 1 Unit = 1 Meter)`
`// fFactor=0.0: kein Doppler-Effekt;`
`// Default = 1.0F`





3D Sound



3D Sounds sollten werden stets an ein Placement angehängt und werden durch die darüber liegende Placement-Hierarchie positioniert. Sonst hört man nix.

In der Klasse Placement gibt's dafür folgende Methode:

```
void AddAudio(CAudio * paudio);
```

Und um ihn wieder abzuhängen, existiert die inverse Methode:

```
bool SubAudio();
```

Sie gibt true aus, wenn das Abhängen geklappt hat.





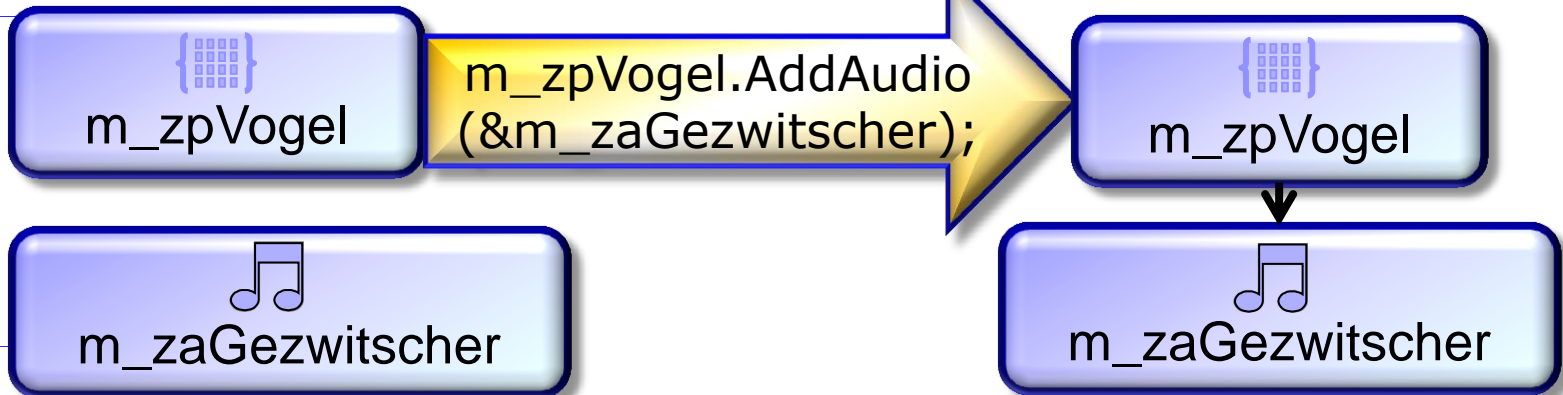
3D Sound - Beispiel



Game.h: `CPlacment m_zpVogel;
CAudio m_zaGezwitscher;`

Game.cpp: `m_zaGezwitscher.Init("Gezwitscher.wav",
(Init) 3.0F);
m_zpVogel.AddSound(&m_zaGezwitscher);
m_zaGezwitscher.Loop();`

...





Übungen zu Sounds

Übung zu Ambient Sounds



Lassen Sie eine Hintergrundmusik laufen!

1

Freiwillige Zusatzaufgabe für die schnellen Nerds:

Lassen Sie die Musik periodisch an- und abschwellen!

2

3

4

5



Übung zu 3D-Sounds



Erzeugen Sie ein beliebiges 3D-Objekt mit einem 3D-Sound!

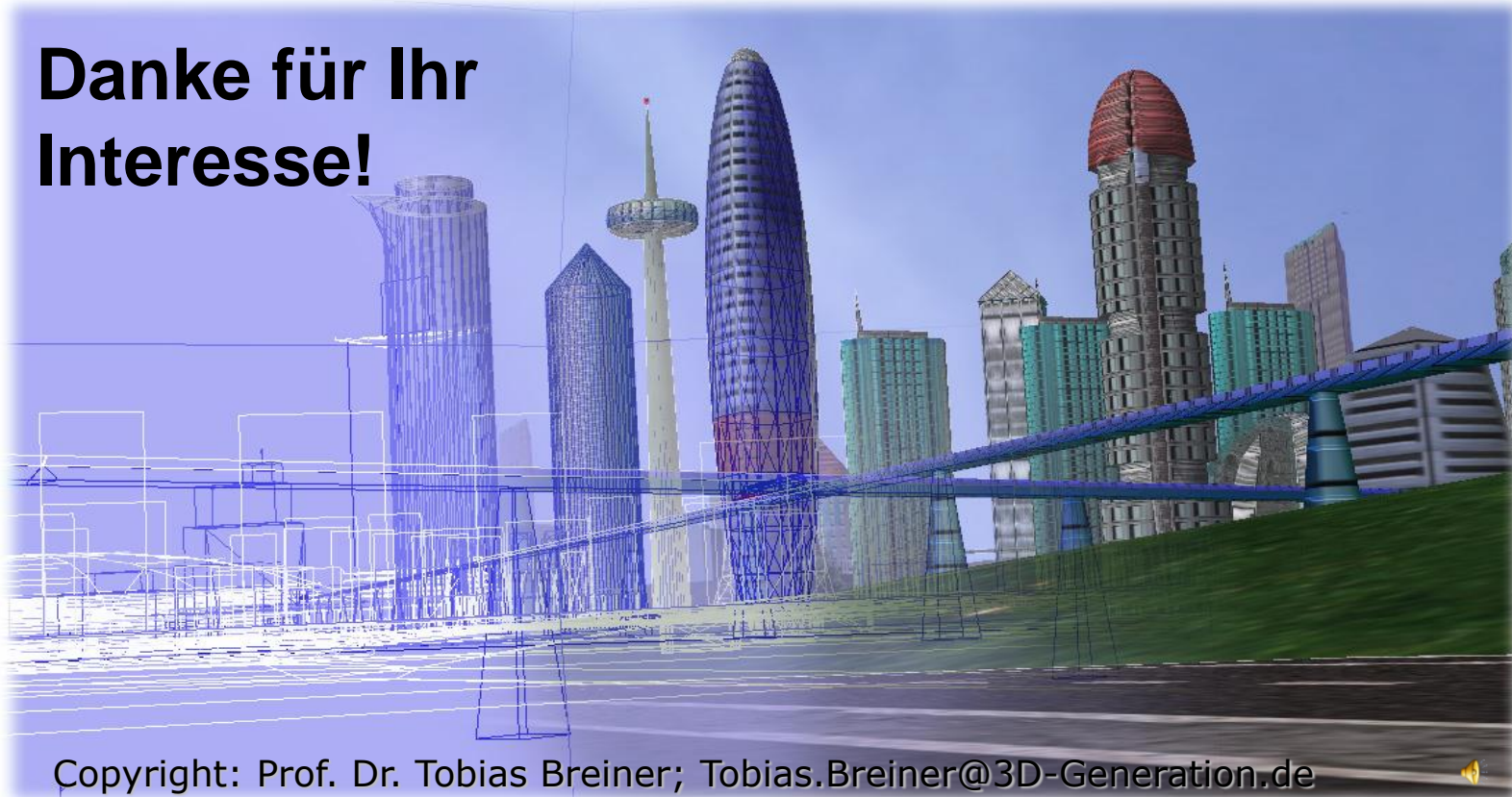
Freiwillige Zusatzaufgabe für die schnellen Nerds:

Justieren Sie den Doppler-Effekt!



///GAME ///OVER

**Danke für Ihr
Interesse!**



Copyright: Prof. Dr. Tobias Breiner; Tobias.Breiner@3D-Generation.de



///PROF. ///DR. ///TOBIAS ///BREINER
///HS ///KEMPTEN

23 VON 23
///SOUND