

swonder3Dq: Auralisation of 3D objects with Wave Field Synthesis

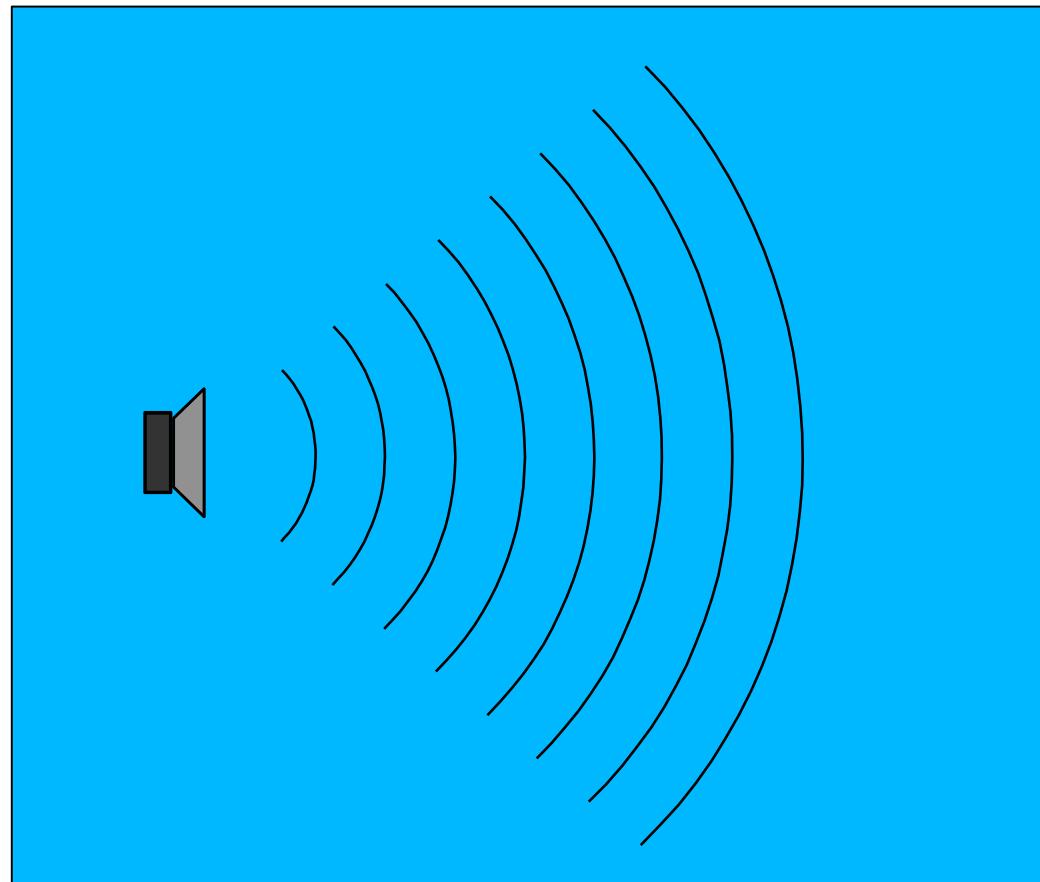
Marije A.J. Baalman
Fachgebiet Kommunikationswissenschaft
Technische Universität Berlin



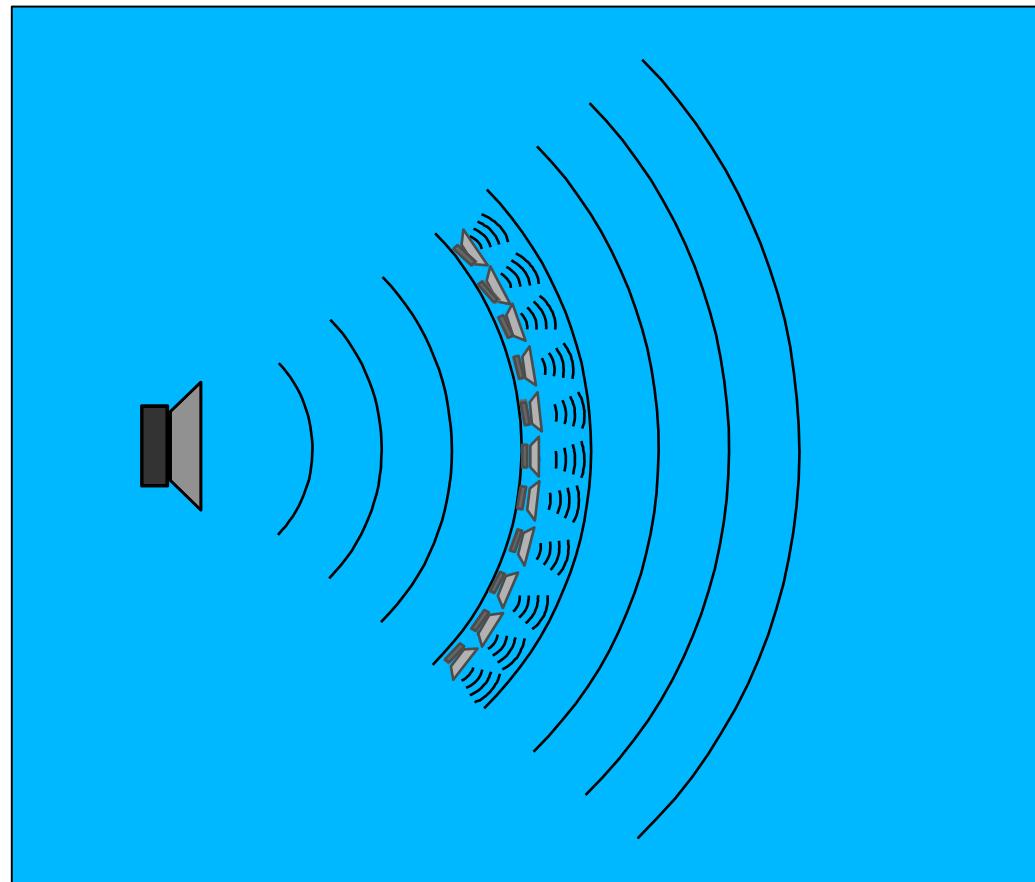
Overview

- Complex sound sources in WFS
- Implementation
 - 3D models
 - WFS calculation
 - user interface
 - engine
- First tests
- Future work

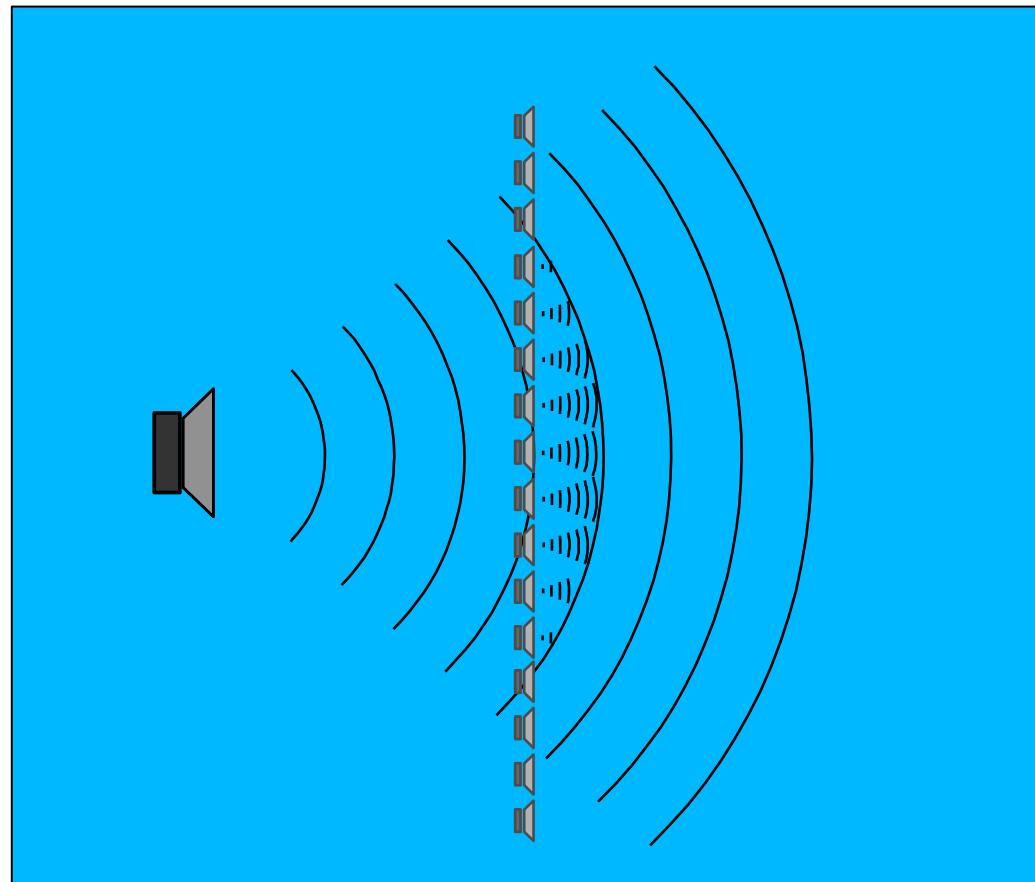
Principle of Huygens



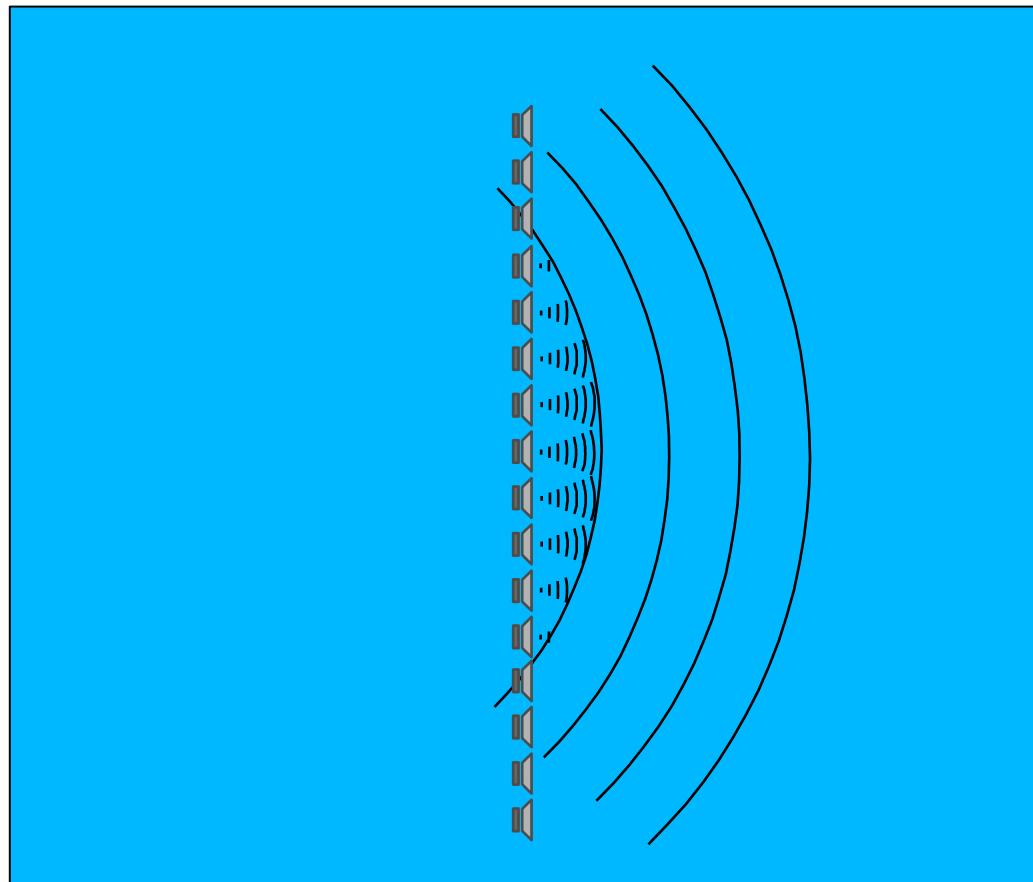
Principle of Huygens



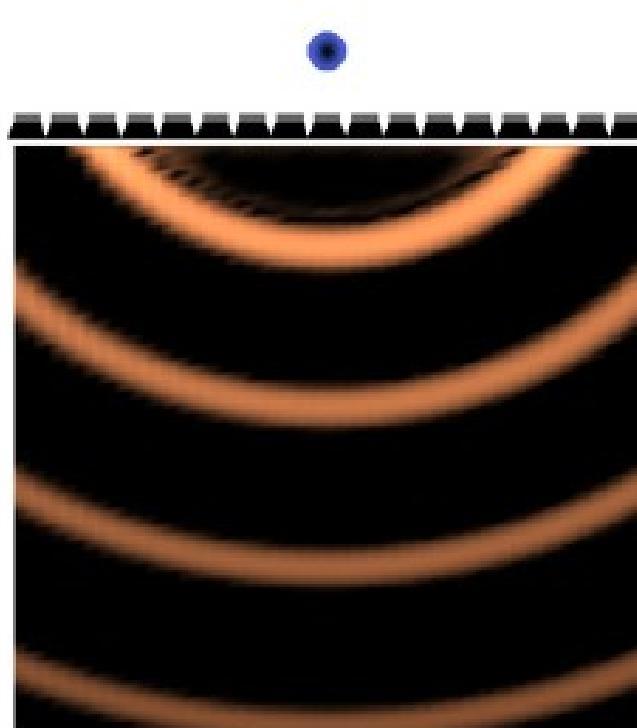
Wave Field Synthesis



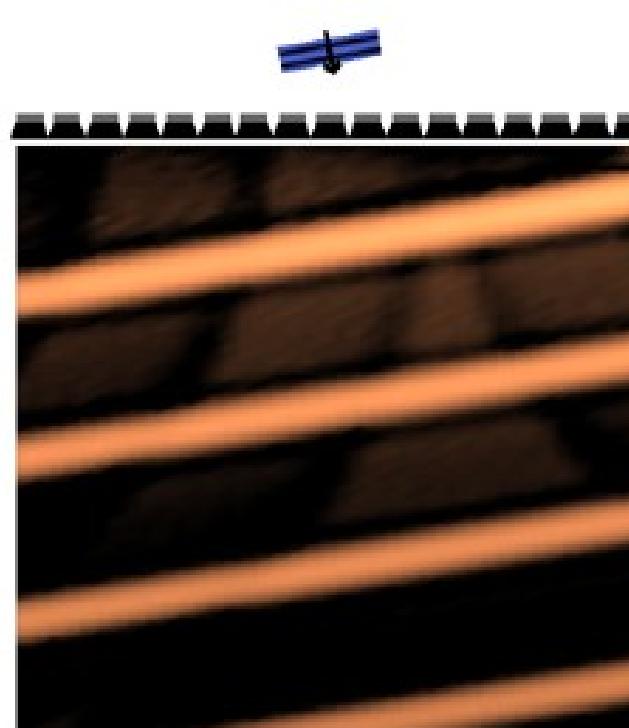
Wave Field Synthesis



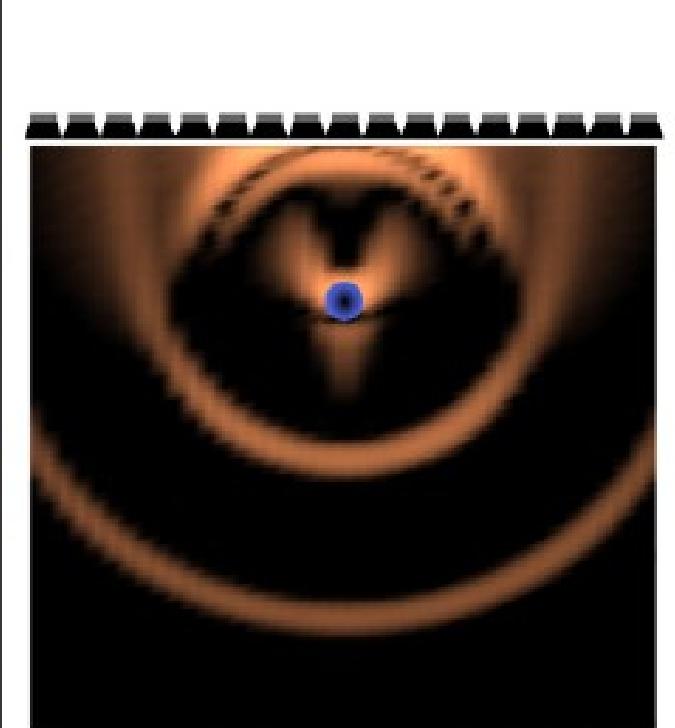
Source types



Virtual point source



Plane wave



Virtual point source situated in front of the loudspeaker array

Complex sound sources

- Currently implemented source types:
 - point source
 - plane wave
- “Ad hoc” solutions for reproduction of large sources
 - Virtual panning spots
 - Multiple point sources
- Start of research for reproduction of sound sources with radiation characteristic

Source model

- Object whose vibration is known or defined on the surface
- The geometry of the object is known or defined
- The surface vibration can be divided in a source signal $S(\omega)$ and a filtering function $G(\omega, r)$ that is dependent on frequency and position on the surface

Elevation

- Current WFS-implementation only takes points in account within the horizontal plane
- For 3D objects points outside of this plane are also relevant
- For this a derivation of the WFS-driving function is necessary for points outside of the horizontal plane

Geometry for elevated points

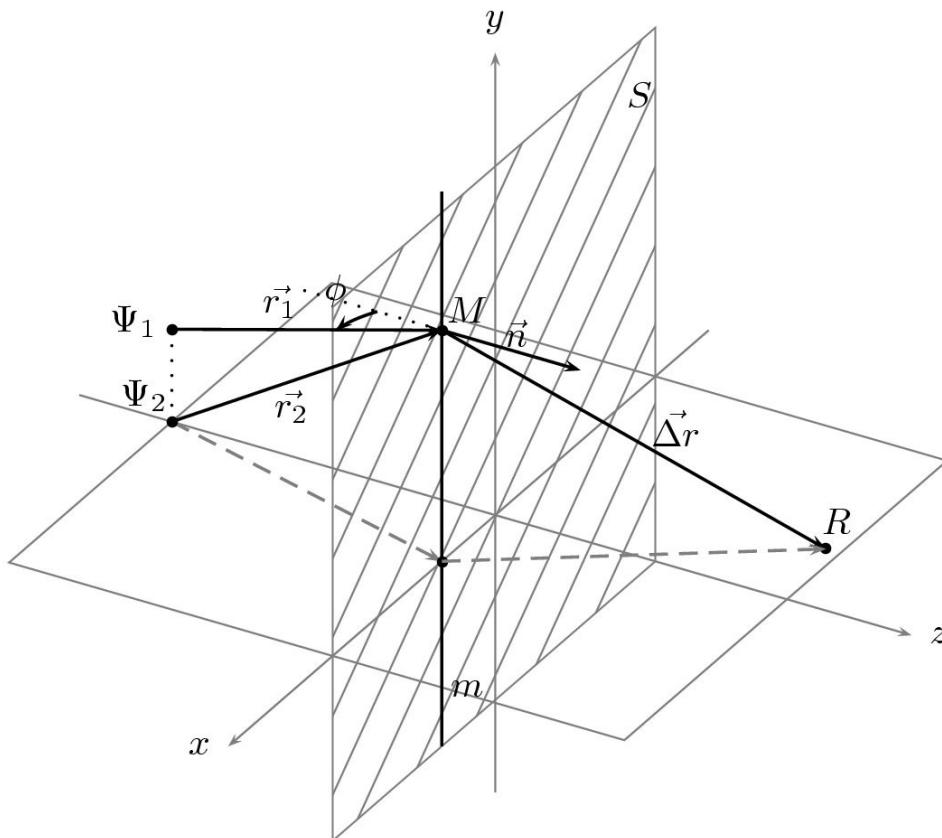


Figure 1. Geometry for the derivation of the $2\frac{1}{2}$ D-operator. Ψ_1 and Ψ_2 are points from the source distribution, \vec{r}_1 and \vec{r}_2 the vectors to a point M on the integration line m . \vec{n} is the normal on the plane S , $\vec{\Delta r}$ is the vector from a point M on the integration line to the receiver point R .

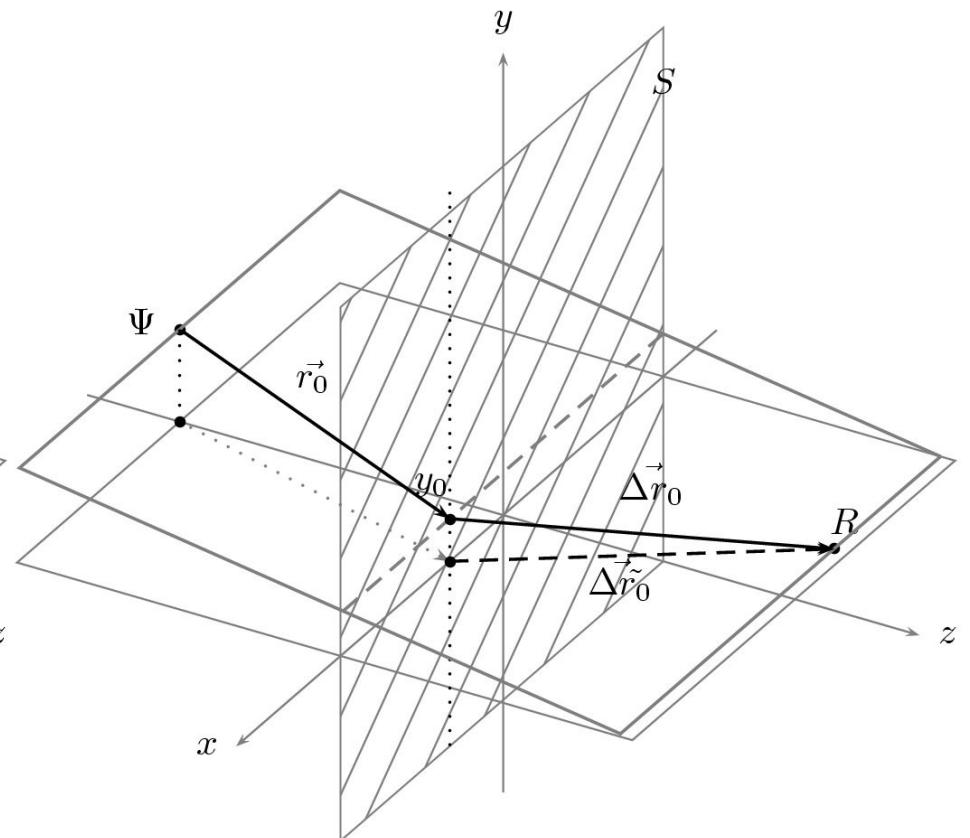


Figure 2. The stationary point y_0 lies on the cross-section of plane S and the plane through Ψ and R . In practice $\vec{\Delta r}_0$ will in fact be $\vec{\Delta r}_{\tilde{0}}$.

Implementation

- General design
- 3D models
- WFS calculation
- Refinement
- User interface
- Engine

Design

- Project
 - Object
 - Mesh
 - Vertices (points on the object)
 - Filters
 - Location
 - Translation
 - Rotation
 - Scale

3D models

- several existing libraries, data formats and viewers for *mesh* data
 - GNU Triangulated Surface (gts) library
 - mview
 - geomview
 - INRIA, medit
- criteria:
 - open source
 - easily extendible: identifiable vertex points

3D models

- *GeomView*
 - external control possible
 - available in Linux distributions
- *mview*
 - written using Qt-Libraries
 - easily extendible
- *gts*
 - vertex points not identified, so difficult to add filter nodes

WFS calculation

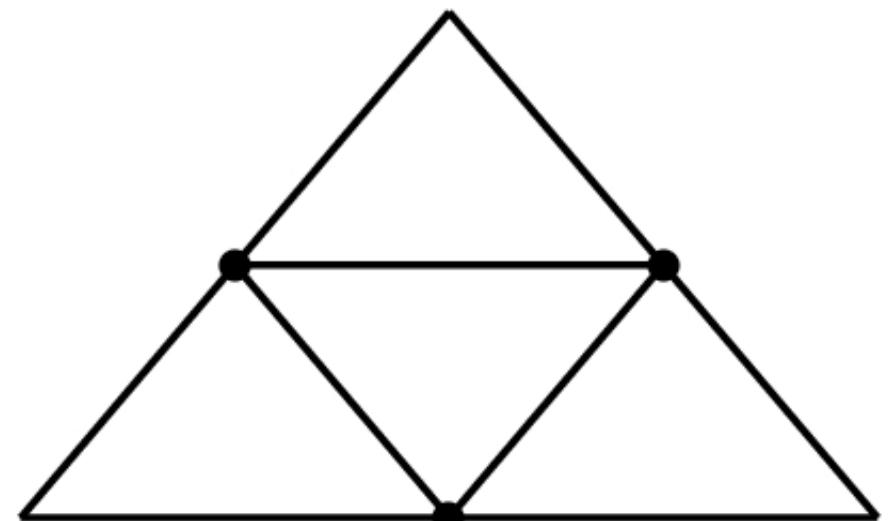
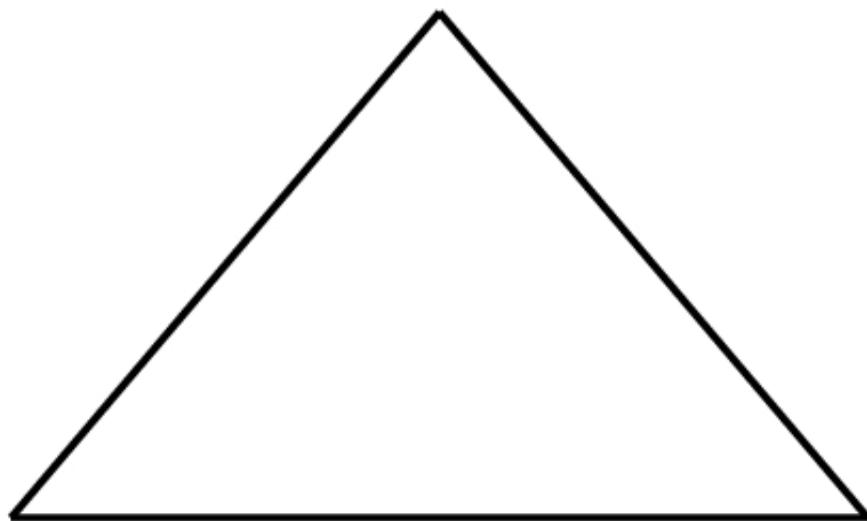
- For each object
 - for each location
 - transform mesh
 - for each point (& for each speaker)
 - check visibility
 - calculate stationary point
 - calculate delay and attenuation
 - convolution with filter
 - save to disk

Visibility check

- Sound from points on the source that are at the backside (seen from the speaker) of the object, will not be heard by that speaker
 - calculate line segment between point and speaker
 - calculate crossings with surfaces
 - if there is a crossing with one of the object surface, point is obscured
- *Diffraction of waves is neglected!*

Refinement

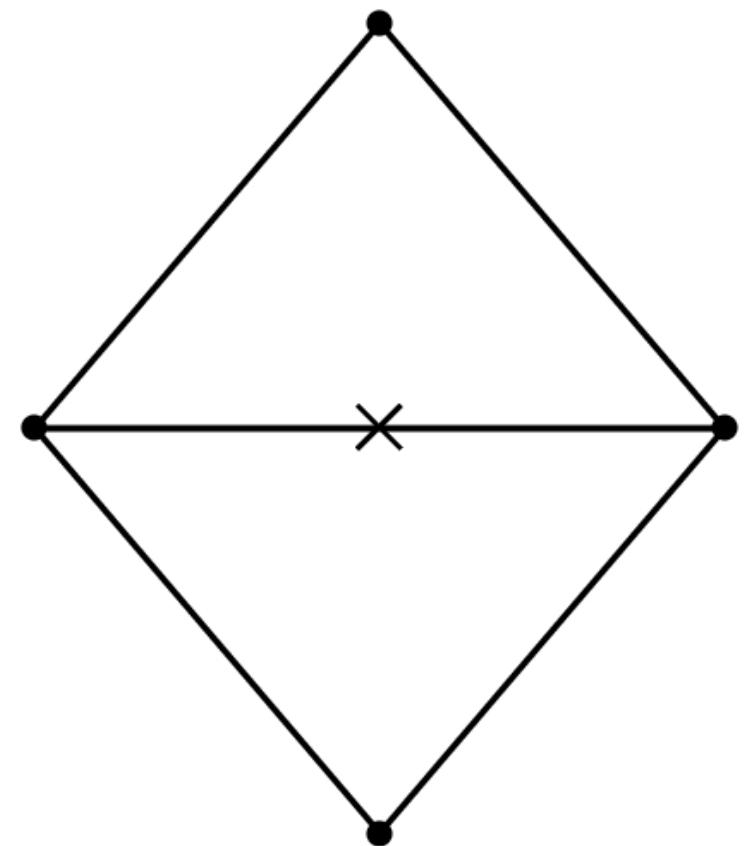
- *midvertex insertion*



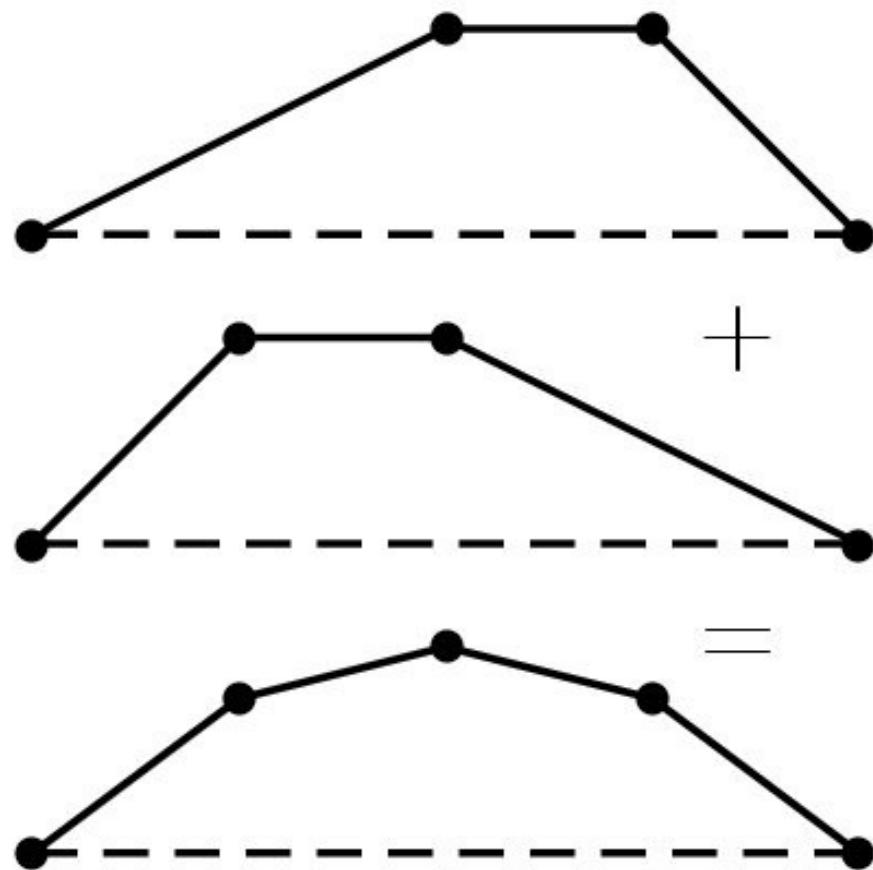
Filter average

- *inverse distance weighting*

$$Z_j = \frac{\sum_{i=1}^n \frac{Z_i}{h_{ij}^\beta}}{\sum_{i=1}^n \frac{1}{h_{ij}^\beta}}$$



Filter average



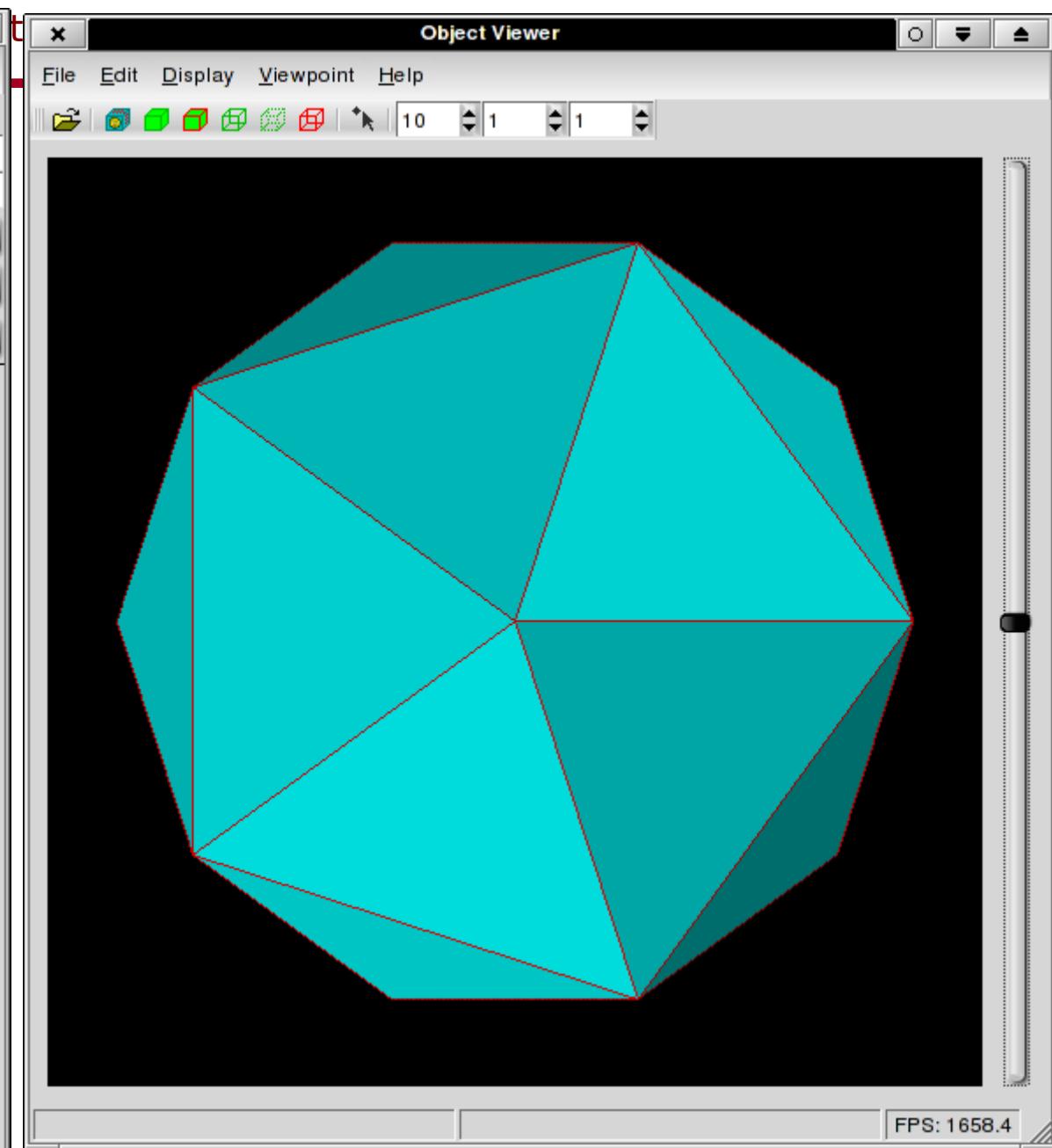
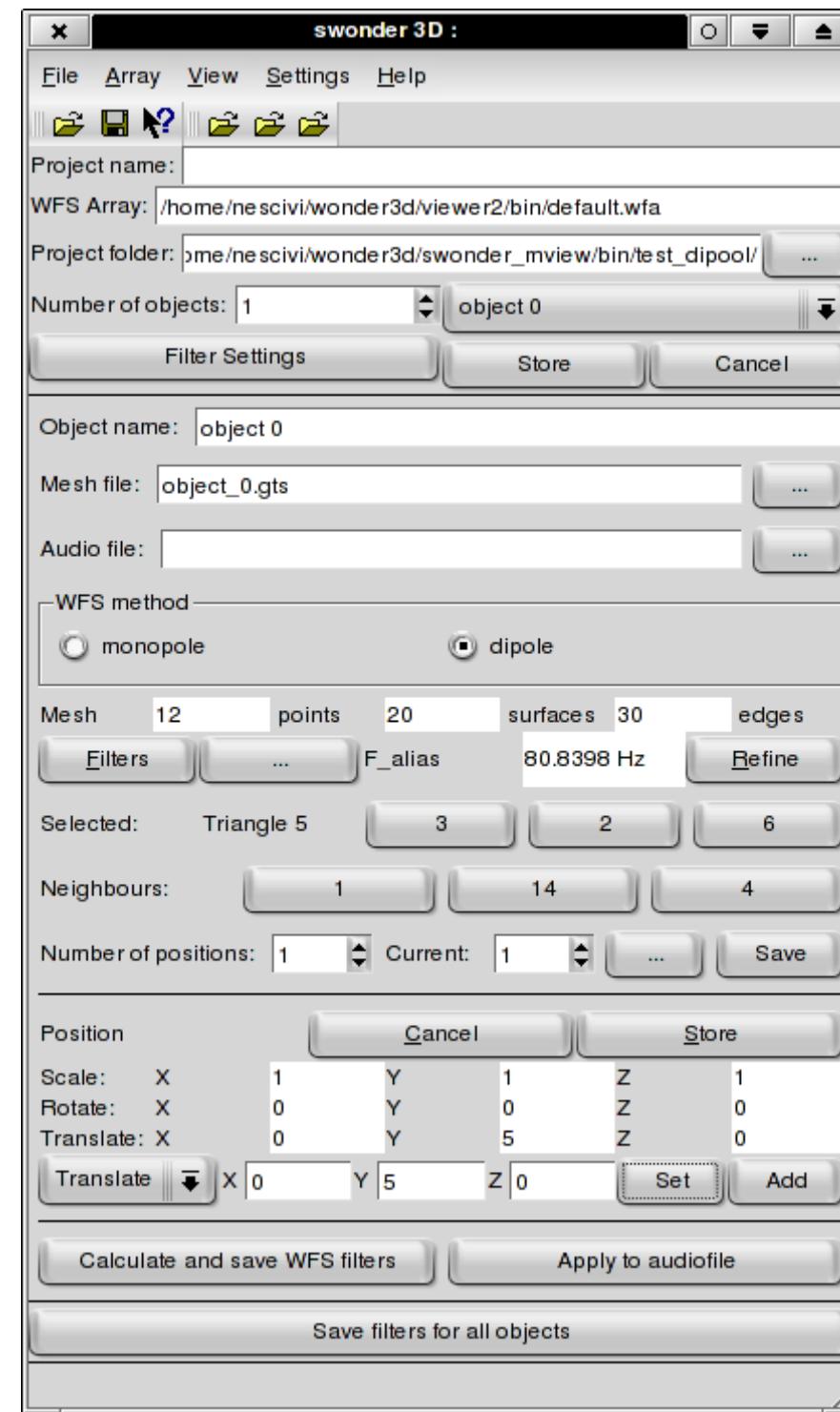
Software *swonder3Dq*

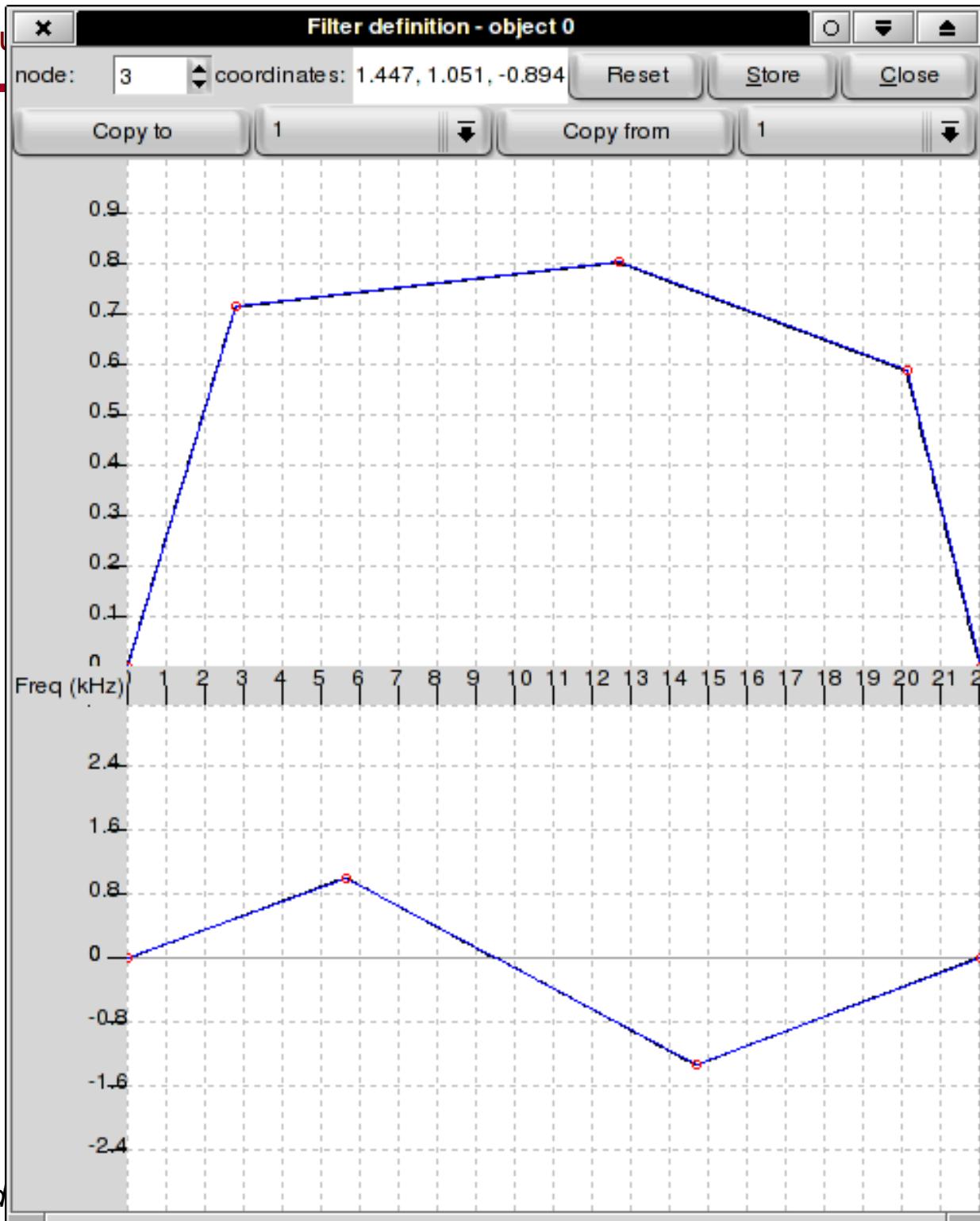
- Graphical user interface to define a project and do the calculations of the filters

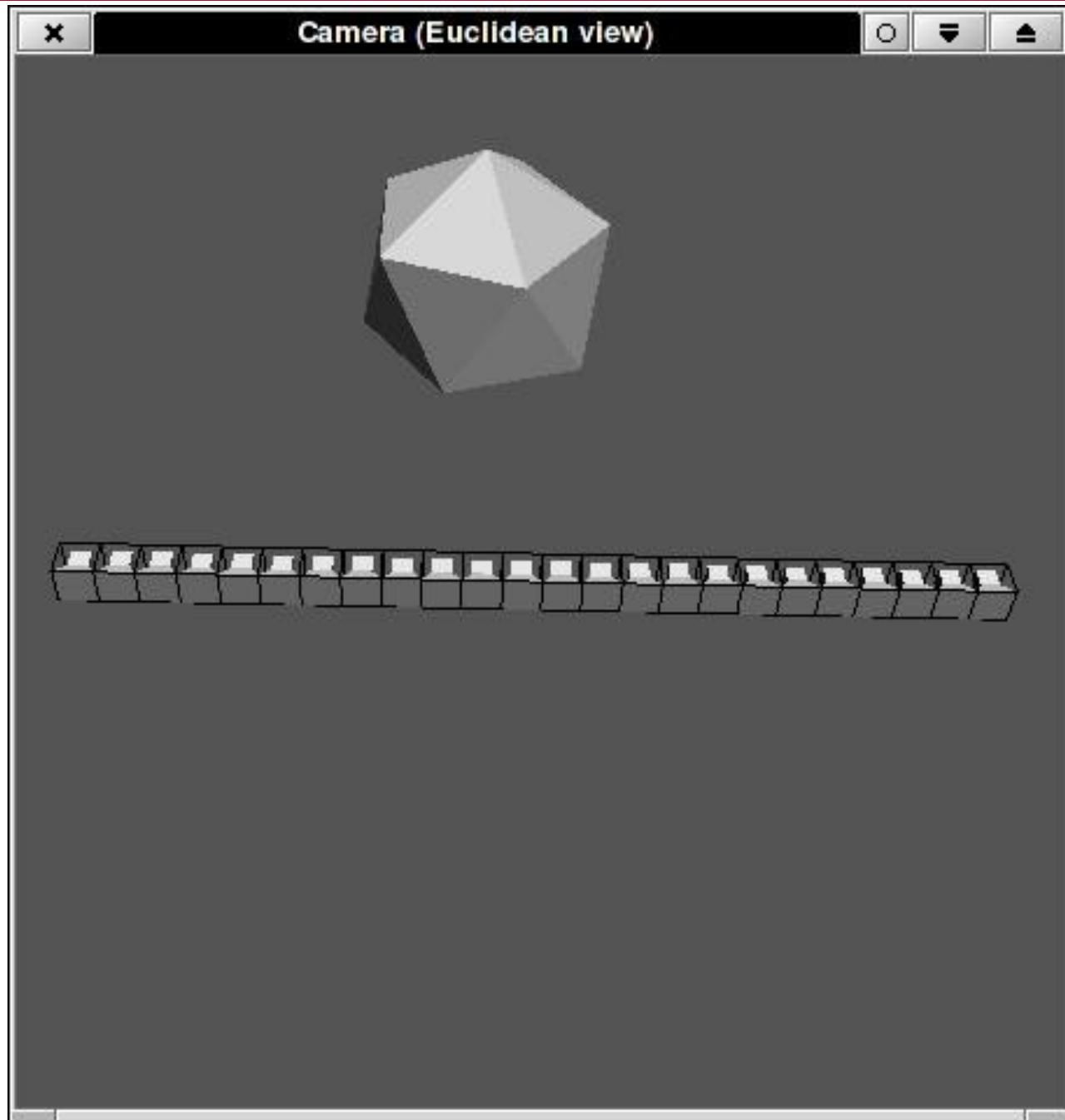
swonder3Dq

- Command line program to control the engine and viewer
 - Controllable with OpenSoundControl (OSC)

swonder3d_engine







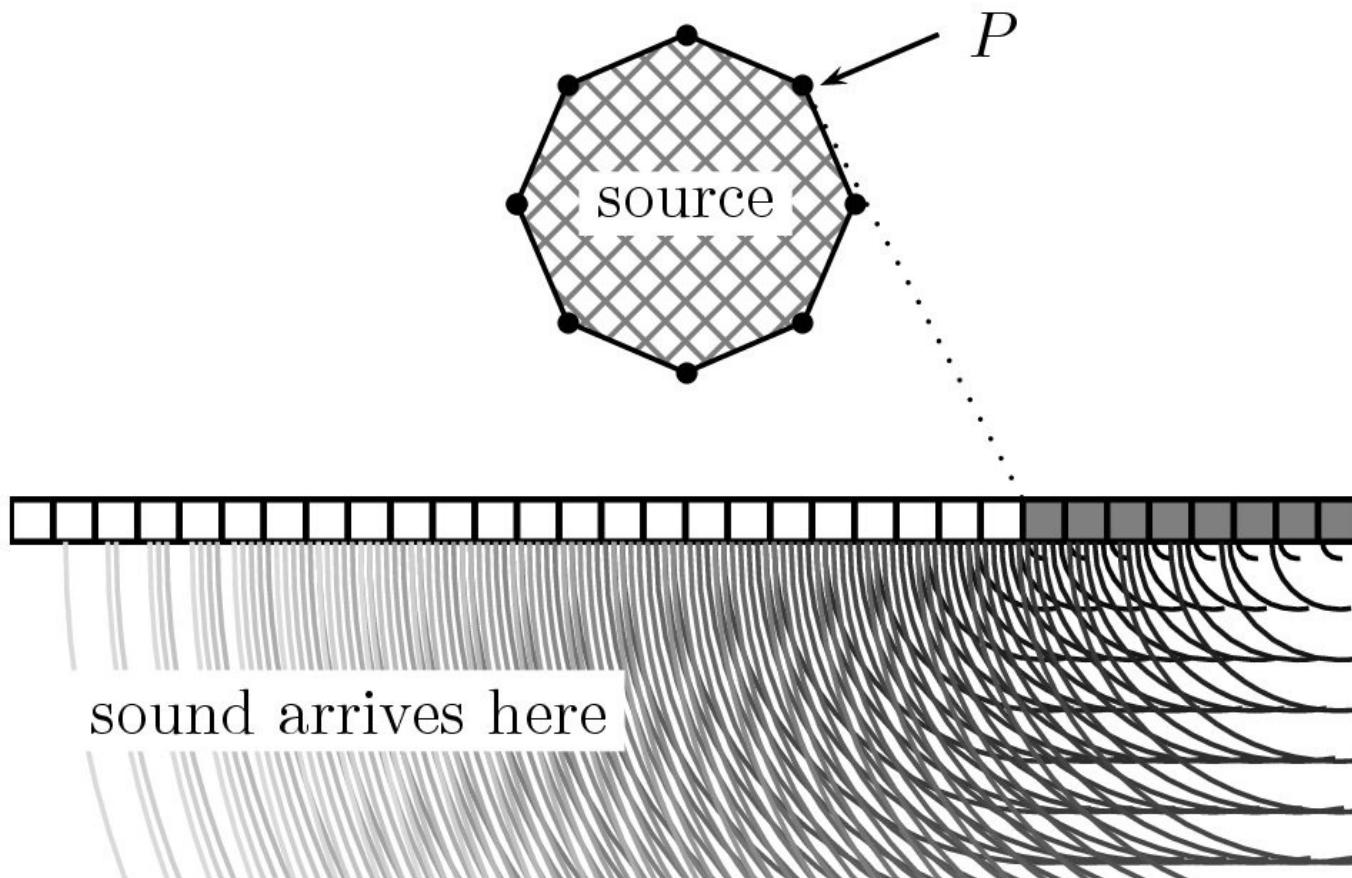
OSC control over engine

- /start
- /stop
- /project - filename
- /change – object, location
- /mute – object
- /client – host, port
- /info – *about renderer
status, project, object,
location*
- /geomview/start
- /geomview/stop
- /geomview/project
- /geomview/array
- /geomview/top
- /geomview/front
- /verbose
- /quit

First tests

- With a 24 speaker prototype setup
- This approach does give a stronger spatial impression
- Problem with neglecting diffraction

Problem



Future work

- Study of diffraction and implementation
- Listening tests
- Usability tests by working with composers
- release on sourceforge:

<http://swonder.sourceforge.net>