OPEN ACCESS Freely available online
V. Berstis,
"3D Printed Acoustic Lens for Dispersing Sound,"
J. Audio Eng. Soc., vol. 66, no. 12, pp. 1082–1093, (2018 December.).
DOI: https://doi.org/10.17743/jaes.2018.0063

3D Printed Acoustic Lens for Dispersing Sound

VIKTORS BERSTIS, *AES Associate Member*
(aesjm@berstis.com)

IBM Corporation, Portland OR, USA

The conventional approach to achieving relatively uniform directional dispersion of sound from an audio monitor is to use drivers substantially smaller than the wavelengths of sound they are reproducing. However, it is desirable to use larger drivers to counteract difficulties in producing sufficient amplitude and linearity. Larger drivers emit nearly planar wave fronts that produce substantially larger amplitudes on axis, known as "beaming." With the advent of 3D printing technologies, it is possible to print acoustic lenses that have negative focal length, better dispersing the sound. The approach uses an array of physical channels to delay portions of the planar wave front shaping it into a spherical wave front having an apparent point source.

0 INTRODUCTION

The conventional approach to achieving relatively uniform directional dispersion of sound from an audio monitor is to use diffraction from drivers substantially smaller than the wavelengths of sound they are reproducing. Ideally one would like to have a point sound source emitting a spherical wave front in at least a 90 degree cone [1] or other dispersion shape. Most drivers, however, produce a nearly planar wave front when the frequencies of sound have a wavelength smaller than the size of the driver. Such drivers give a very noticeable beaming effect on axis. While one approach is to use an array of drivers mounted on a spherical surface, this is only practical for very large sound systems. In smaller systems, especially when more linearity is desired [2], a larger driver would be desirable so that it does not have to be driven close to its limits.

The earliest use of acoustic lenses to disperse sound appeared in the 1940s and 1950s with devices such as slant plates [3] used in some JBL speakers [4]. Diffraction horns [5] that narrowed the opening were also used. However, most interest seemed to be in focusing sound, especially in the ultrasound frequencies [6]. A range of methods appeared using materials with varying index of refraction [7, 8] or devices with a Fresnel lens to focus or defocus certain wavelengths [9]. Phased arrays using many drivers [10] was another alternative, but more expensive. Reflecting sound from ellipsoids [11] or other shapes have been described. More recently Bang Olufsen offers speakers with drivers pointing upwards toward a reflective surface symmetrical around the vertical axis [12].

This paper explores the design of a low cost lens constructed using a 3D printer.

1 DESIGN

The approach described here breaks up the wave front into an array of paths and delays the sound passing through each array element an appropriate amount to recombine as a spherical wave front. It is somewhat analogous to the phase plate approach but is not limited to creating an essentially cylindrical wave front. It is more similar to a phased array of emitters but accomplished by using a specific geometry of paths. Each sound path consists of an approximately nearly square, trapezoidal tube with dimensions of approximately 6 millimeters by 6 millimeters. The size is chosen to be substantially smaller than the wavelengths of sound to be used. Fig. 1 shows one set of tubes at different distances from the axis.

The first thought was to construct layers of tubes with a folding architecture to achieve the desired delays in portions of the wave front. Fig. 2 illustrates one layer of such a design somewhat reminiscent of Turbosound's Polyhorn-Flexarray [13]. However, each layer would be slightly different and the folds were likely to attenuate the signal more than desired.

Because the driver is symmetrical around the axis, instead of being folded, the paths could be lengthened by gradual twisting around the axis. The most straight forward design seems to be to have a flat surface in front of the driver and a spherical exit surface which results in a minimum of physical blocking of the wave front. Fig. 3 shows the final 3D-printed lens.

Taking advantage of the circular symmetry of the driver, the tubular paths are tilted with increasing angles from vertical, forming helixes, as they are placed further from the center axis. This extends each path such that the sound

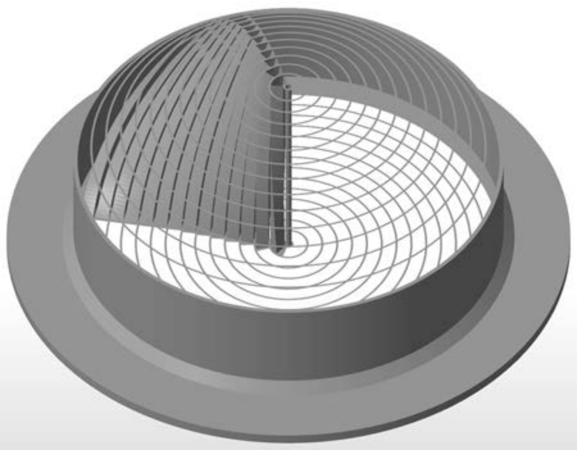


Fig. 1. Set of tubes of equal length at range of distances from axis.

Fig. 2. Initial idea to have custom layers of folded tubes.

Fig. 3. 3D printed acoustic lens.

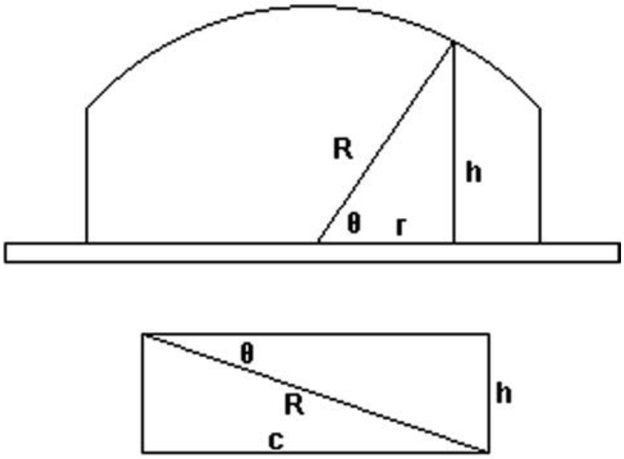


Fig. 4. Calculation of the geometry of the lens tubes.

is delayed the proper amount at the exit point to recombine with neighboring tubes to form the desired spherical wave front. Another way to imagine the structure is to think of a bundle of equal length flexible soda straws which would cover the front of the driver. Then each concentric layer is twisted around the axis. The soda straws will tilt and bend slightly. The tilts should be such that the openings form a spherical surface. Surprisingly the twist turns out to be the same angle around the axis for each concentric layer. If the exit surface were a different shape—flat, for example—the tube lengths and tilts would have to be adjusted accordingly to achieve the desired spherical wave front.

The calculation of the geometry is as follows. Looking at Fig. 4, the surface of the lens is spherical with radius R from the center of the flat area on the bottom of the lens. The exit point of any tube would be at height $h = R \sin \theta$. The tubes form a portion of a helix at distance $r = R \cos \theta$ from the axis. The helix is flattened as shown by the rectangle in Fig. 4. All tubes have the same length *R* because the sound enters as a plane wave front on the flat side of the lens and exits as a spherical wave front on the spherical surface of the lens. Given $h = R \sin \theta$, the helical path of the tube projected onto the flat bottom of the lens has length $c = R \cos \theta$. Because *c* is the same as the distance from the axis *r*, the angle of the overall twist of all tubes is constant at $180c/(r \pi) = 180R \cos \theta /(\pi R \cos \theta) = 57.3$ degrees, or one radian.

The structure is generated by a computer program [see Appendix] that described the entire lens structure using about 1.1 million triangles (their coordinates and normal vectors) as a Standard Tessellation Language (STL) file. This file type is typically used by 3D printer software to describe objects to be printed. The 3D printer software slices the object into horizontal layers, in this case about 0.2 mm thick, and generates a G-code (RS-274) file that directs the XYZ motions and filament feed of the 3D printer. The printer melts the plastic material (acrylonitrile butadiene styrene—ABS) to a temperature of about 240°C and ejects it from a nozzle with a 0.5 mm diameter hole to slowly lay down the material in threads, layer by layer. The printing process for this lens takes about two weeks of time on an upgraded, open-source Lulzbot TAZ-3 printer.

2 MEASUREMENTS

2.1 Wave Front Delay

The goal was to measure if the lens converted the plane wave front into a spherical wave front. We first measured the wave front produced by the 10-inch sealed back driver (Beyma 10MCF400Nd) without the presence of the lens. A 150 hz square wave signal was fed into the driver.

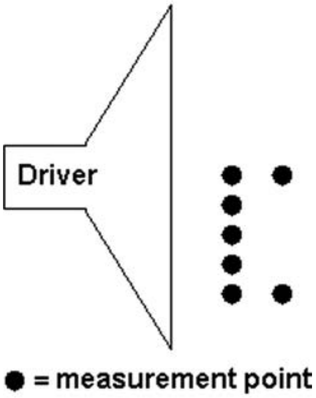


Fig. 5. Setup for measuring without lens (see Table 1).

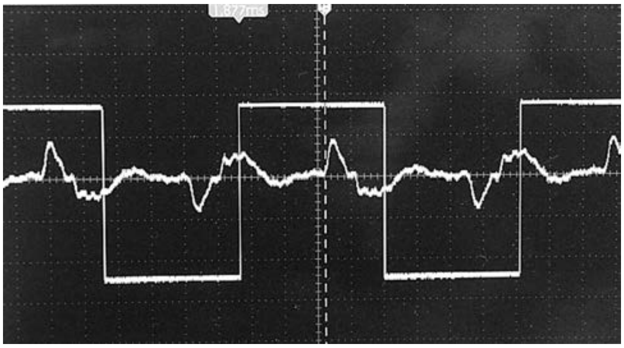


Fig. 6. Square 150 hz input and driver response.

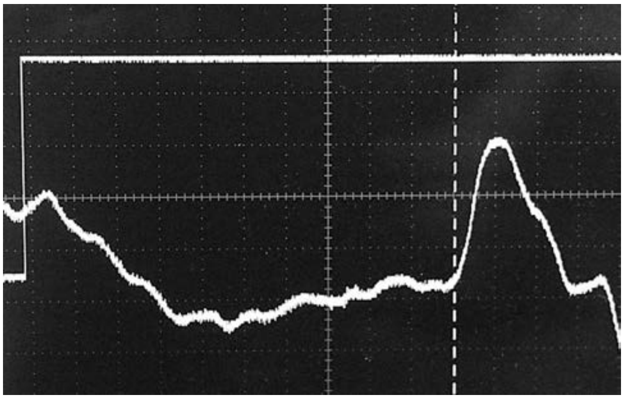


Fig. 7. Measuring delay by finding knee of response.

Table 1. Relative delay without lens

Millimeters from axis	Signal delay, milliseconds at 55 mm from driver	Signal delay, milliseconds at 80mm from driver
0	1.640	1.680
25	1.640	
50	1.640	
75	1.640	
100	1.640	1.700

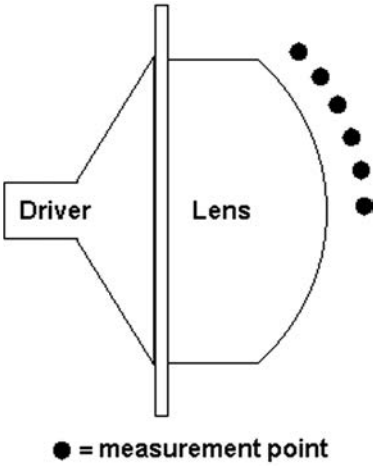


Fig. 8. Setup for measuring with lens (see Table 2).

Table 2. Relative delay with lens

Angle from axis in degrees	Signal delay, milliseconds at 45 mm from lens
0	1.740
10	1.740
20	1.765
30	1.785
40	1.780
50	1.780

Table 3. Response at 3 khz with and without lens.

Degrees off axis	Sound at one meter distance in dB (no lens)	Sound at one meter distance in dB (with lens)
0	0.0	−9.7
15	−5.5	−11.8
30	−14.7	−12.5
45	−22.2	−14.4

A measurement microphone (Behringer ECM8000) was placed at a distance of 55 mm from the front plane of the driver at several places in the plane parallel to the front of the driver as illustrated by Fig. 5. The knee of the leading edge of the rising signal received was observed using an oscilloscope (Figs. 6 and 7) to measure the relative delay with respect to a fixed point on the input signal at different distances from the axis. As shown in Table 1, the equal values show that the wave front is planar at that distance. Two measurements were made at a greater distance to confirm that the setup was able to properly resolve the difference in wave front arrival time.

Next, the acoustic lens was installed covering the front of the driver. The same square wave signal was used as input. The output wave front was measured at spherical positions at a radius of 170 mm from the virtual point source point of the sound as illustrated by Fig. 8. This was 45 mm away from the surface of the acoustic lens. The radius of curvature of the front of the lens was 125 mm. The approximately equal readings were consistent with an essentially spherical wave front as shown by Table 2.

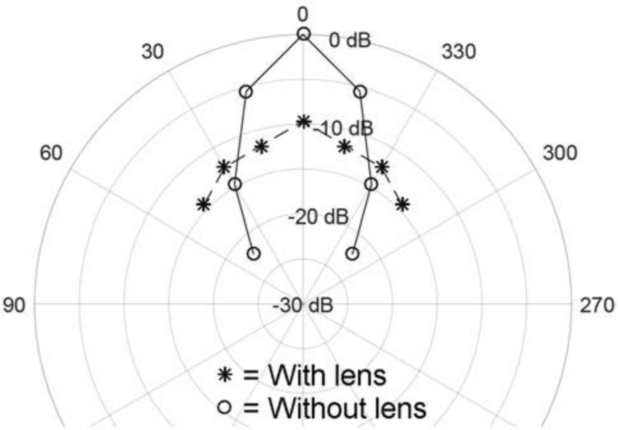


Fig. 9. Response at 3 hkz with and without lens.

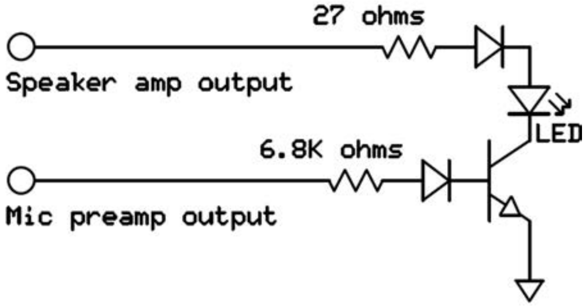


Fig. 10. Schematic of wave front illuminator.

2.2 Beaming Measurement

We had brief access to a semi-anechoic chamber to measure the relative amplitudes of the signal at different angles from the axis, with and without the lens installed. The midrange driver's useful frequency range is about 300–3000 hz. We chose to measure at 3 khz, the frequency at which beaming would be the most problematical. The microphone's distance from the driver was one meter. The measurements show a distinct reduction of on-axis beaming of the signal per Table 3 and Fig. 9.

2.3 Visualizing of Wave Front Shape

To illustrate the actual wave front being emitted, a simple circuit was constructed to perform a logical "and" function between the positive going input sine wave to the driver and the positive going signal received by the microphone as shown in Fig. 10. The LED is taped to the tip of the microphone. The overall volume is adjusted to just start to illuminate the LED. By moving the microphone to different positions, one can easily see the peaks and troughs of the virtual standing wave front created by the circuit. Ideally an x-y scan mechanism would be constructed using a stepper motor table that would position the microphone at an array of points in a plane and record the LED intensities. For expediency we simply photographed the wave front using a 30-second camera exposure set to f22 and ASA 800 while manually waving the microphone with LED, back and forth approximately in a plane perpendicular to the camera's axis. The wave front at 3 khz appears fairly flat without the lens

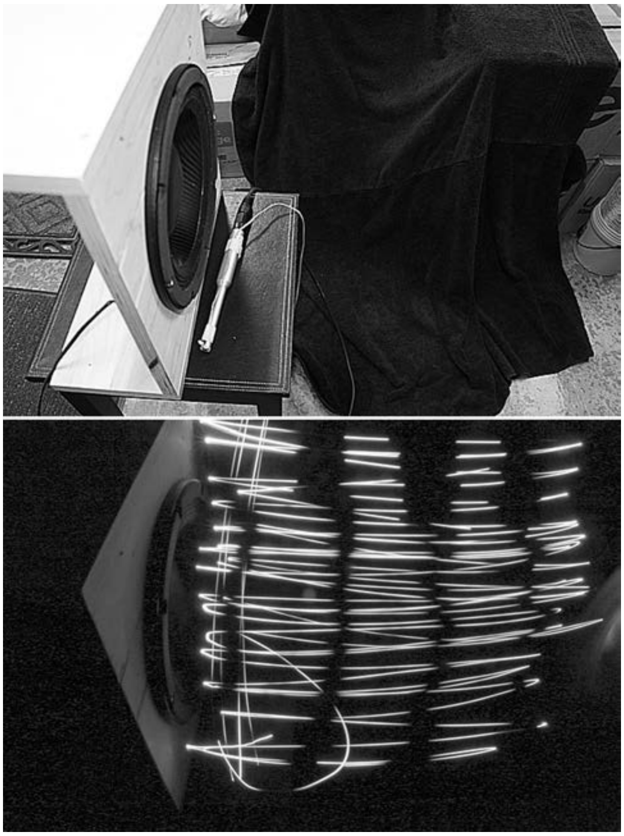


Fig. 11. Photo of wave front without lens.

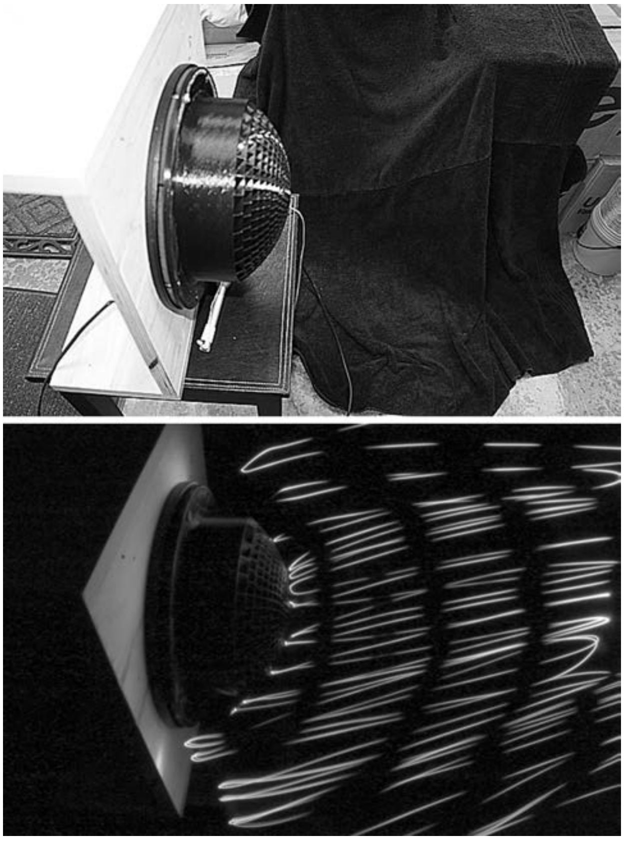
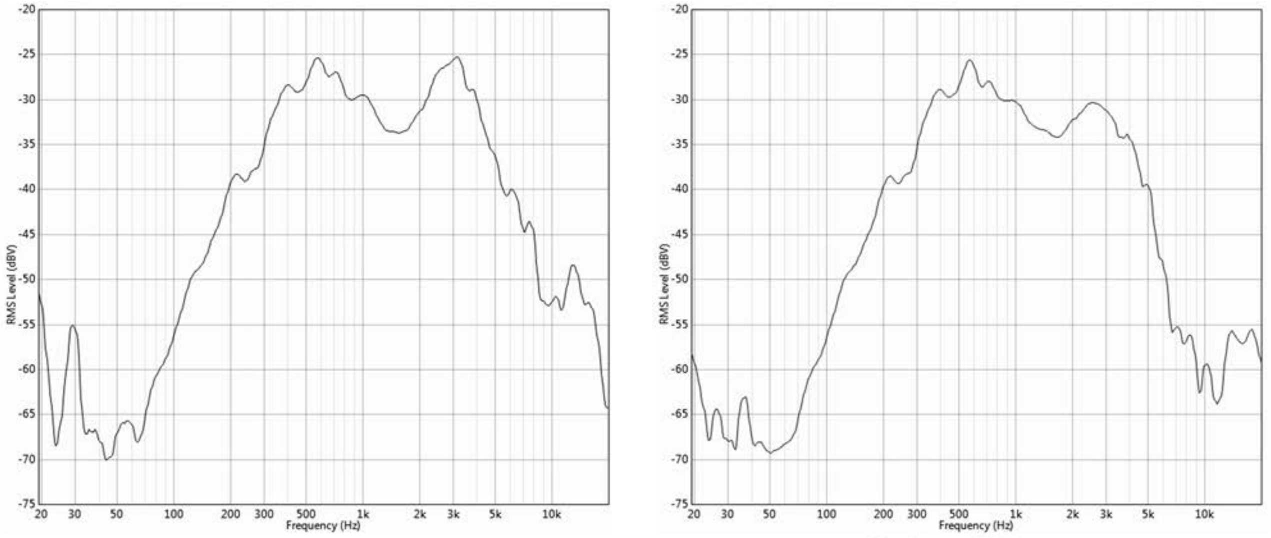
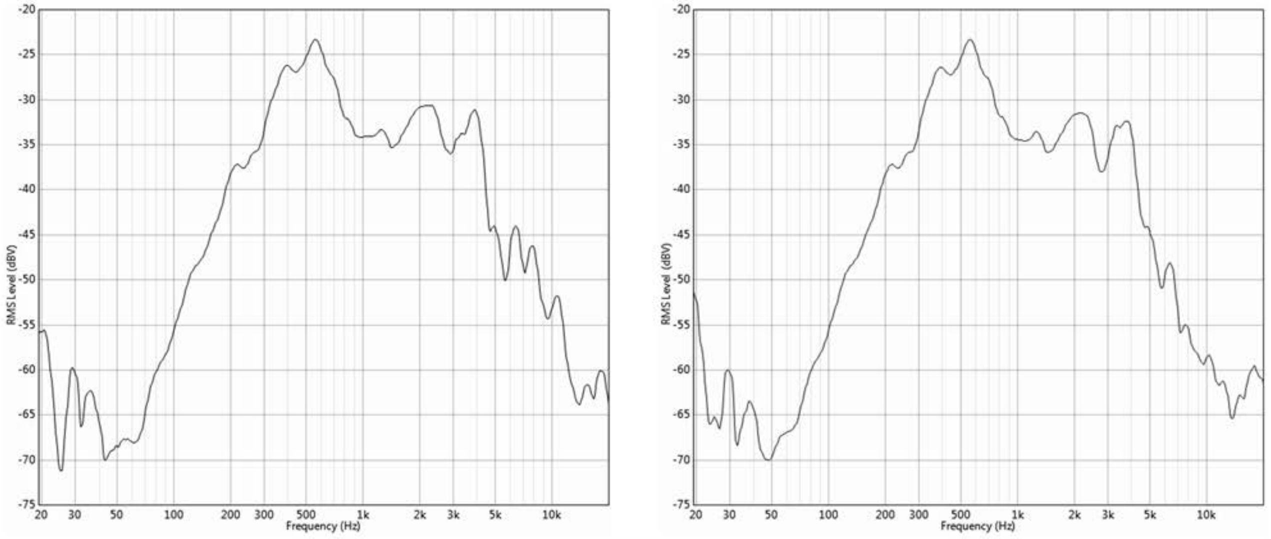


Fig. 12. Photo of wave front with lens.



No Lens, 0 degrees

No Lens, 15 degrees



With Lens, 0 degrees

With Lens, 15 degrees

Fig. 13. Frequency response.

(Fig. 11) and curved with it (Fig. 12). The sound intensity is higher where the null bands are narrower. Where the hand motion was not perfectly in a plane, you can see a few lines where it should be all dark.

2.4 Effect on Frequency Response

We made a cursory measurement of frequency response with and without the lens on axis and 15 degrees off axis. This was not performed in a proper anechoic chamber and we noted that at higher frequencies slight placement changes caused rather dramatic changes in the curves. We will have to leave better measurements to others who have proper facilities and equipment. Fig. 13 shows our results. In the desired frequency range of 300–3000 hz, with the lens, the response is fairly similar on and off axis compared to not using a lens.

3 FUTURE INVESTIGATIONS

As demonstrated by the measurements, this approach has potential to reduce the beaming effect from drivers. More careful measurement of the wave front emitted by the driver could be used to tune the design of the lens. Some of the sound paths through the lens could be lengthened or shortened to more precisely consider anomalies in the shape of the driver's emitted wave front. Likewise paths could be constricted or expanded in various ways to adjust for variations in wave front amplitude at different points. This lens approach may be particularly useful in reducing the pronounced beaming effects typical in planar speakers employing electrostatic membrane drivers, for example.

Although this lens was used as a dispersion device (somewhat analogous to an optical concave lens), the approach could also be used to focus a plane wave sound source. This would require a geometry that would need to delay to

Fig. 14. Practical speaker installation with lenses over midrange and tweeter drivers.

a greater amount, the sound paths which are closer to the axis. The inner tube layers would have to be coiled more than the outer ones. This lens design approach is not readily suitable for imaging (similar to optical camera imaging) because it assumes the sound is entering the lens from a fixed source. It is, however, possible to design acoustic lenses with our approach that preferentially distribute the sound in specific directions, horizontally, for example.

While resonance has not been a noticeable issue, introducing spherical layers of empty breaks in the tubes, shortening each tube's length, may be a solution in situations where this becomes a problem.

4 CONCLUSIONS

In a practical speaker installation (Fig. 14) this acoustic lens reduced the on-axis beaming effect by reshaping the driver's planar wave front into a spherical one. Subjective impressions from listeners were very positive. 3D printing opens up the possibility creating a range of such lenses for various purposes, particularly changing the shape and nature of emitted wave fronts.

4 ACKNOWLEDGMENTS

Many thanks go to Dr. Mark Martin of Audio Precision Inc. for his valuable assistance. Also, thanks to my wife Dr. Sylvia Gray for her encouragement and tolerance of extensive technical intrusions into living spaces.

5 REFERENCES

[1] J. Kates, "Optimum Loudspeaker Directional Patterns," *J. Audio Eng. Soc.*, vol. 28, pp. 787–794 (1980 Nov.). http://www.aes.org/e-lib/browse.cfm?elib=3948

[2] P. Kantrowitz, "Distortion of High-Frequency Loudspeakers," *J. Audio Eng. Soc.*, vol. 10, no. 4, pp. 310–317 (1962 Oct.). http://www.aes.org/e-lib/browse.cfm?elib=835

[3] E. Geddes, "Acoustic Lens, Their Design and Application," presented at the *61st Convention of the Audio Engineering Society* (1978 Nov.) convention paper 1401. http://www.aes.org/e-lib/browse.cfm?elib=2953

[4] JBL "Professional Series Acoustic Lens Family" (1954). http://www.jblpro.com/pub/obsolete/Acoustic_Lens_Family1.pdf

[5] JBL "Professional Series 2397 Diffraction Horn" (1954). http://www.jblpro.com/pub/obsolete/2397.pdf

[6] C. Chang, K. Firouzi, K. K. Park, A. F. Sarioglu, A. Nikoozadeh, H. Yoon, S. Vaithilingam, T. Carver, and B. T. Khuri-Yakub, "Acoustic Lens for Capacitive Micromachined Ultrasonic Transducers," *J. Micromechanics and Microengineering*, vol. 24, no. 8, (2014 July). https://doi.org/10.1088/0960-1317/24/8/085007

[7] M. Hurwitz, "Acoustic Lens System" US Patent 4,001,766 (1977 Jan.).

[8] G. Maimbourg, A. Houdouin, T. Deffieux, M. Tanter, and J. Aubry, "3D-Printed Adaptive Acoustic Lens as a Disruptive Technology for Transcranial Ultrasound Therapy Using Single-Element Transducers," *Physics in Medicine & Biology*, vol. 63, no. 2 (2018 Jan.). https://doi.org/10.1088/1361-6560/aaa037

[9] B. Hadimioglu, E. Rawson, R. Lujan, M. Lim, J. Zesch, B. Khuri-Yakub, and C. Quate, "High-Efficiency Fresnel Acoustic Lenses," *IEEE Ultrasonics Symposium,* vol. 2, pp. 579–582 (1993 Oct.). https://doi.org/10.1109/ULTSYM.1993.339544 https://www.researchgate.net/publication/3570074/download

[10] D. Keele, "Time/Phase Behavior of Constant Beamwidth Transducer (CBT) Circular-Arc Loudspeaker Line Arrays," presented at the *139th Convention of the Audio Engineering Society* (2015 Oct.), convention paper 9387. http://www.aes.org/e-lib/browse.cfm?elib=17944

[11] M. Ferralli, D. Moulton, D. Brugger, J. White, and S. Hebrock, "Wide Dispersion Frequency Invariant Acoustic Lens," presented at the *79th Convention of the Audio Engineering Society* (1985 Oct.) convention paper 2267. http://www.aes.org/e-lib/browse.cfm?elib=11488

[12] Bang & Olufsen "Beolab-20." https://www.bang-olufsen.com/en/collection/speakers/beolab-20

[13] Illustration of Turbosound Flexarray/Polyhorn. http://europe.nxtbook.com/nxteu/turbosound/brochure/index.php?startid=8#/10

APPENDIX

This is the SNOBOL4 program that generated the STL file for the lens:

```
* program to generate stl file for acoustic lens
* this version has round base for midrange driver
   pi = 3.141592653
* scale factor
   scale = 1.00
* sizes are in millimeters
* main height and hemisphere radius and
*  hemisphere radius center z point
   mh = 105.0
   hr = 125.00
   hrcz = mh - hr
* wave’s exiting channel width
   wave = 6.0
* partition thickness
   pth = 1.0
* center hole radius
   cr = wave / 4.0
* outer lens radial size
   or = 95.0
* adjust radial size to proper multiple to
*  make each concentric layer uniform
   orn = CONVERT((or - pth - cr) / (pth + wave),’INTEGER’)
   or = pth + cr + orn * (pth + wave)
* outer flange radial size
   ofr = 133.0
* outer flange height
   ofh = 4.7
* mounting hole center distance from overall center
   mhc = 124.0
* mounting hole radius
   mhr = 4.2
* offset to make coordinates not negative  (rs * scale)
   offset = 135.0 * scale
* degree granularity
   d0 = 1.0
   dn = 360.0 / d0
   dn = CONVERT(dn,’INTEGER’)
getdlp  EQ(REMDR(360 * 64,dn)) :S(gotd)
   dn = dn + 1 :(getdlp)
gotd  d = 360.0 / dn
* define angle adjust routine
   DEFINE(’aa(a)’)
* define other drawing routines
   DEFINE(’section(sr1,sa1,sz1,sr2,sa2,sz2,sr3,sa3,sz3,’
+    ’sr4,sa4,sz4)dn,n,dd,sr14d,sr23d,sz14d,sz23d,’
+    ’tx3,ty3,tz3,tr3,ta3,tx4,ty4,tz4,tr4,ta4,sx1,’
+    ’sy1,sx2,sy2’)
   DEFINE(’rect(rx1,ry1,rz1,rx2,ry2,rz2,rx3,ry3,rz3,rx4,’
+    ’ry4,rz4)’)
   DEFINE(’facet(fx1,fy1,fz1,fx2,fy2,fz2,fx3,fy3,fz3)Vx,’
+    ’Vy,Vz,Wx,Wy,Wz,Nx,Ny,Nz,N’)
* routine to make sure there is at least a zero after a
*  decimal point in the output for FreeCAD
   DEFINE(’PAD(x)’)
   OUTPUT(’print3d’,7,,’lens.stl’)
* first line of STL file:
   print3d = "solid simplicity"
***********************************************************************
* make center cylinder first using d degree chunks
   radius = cr; z = hr + hrcz
   section(cr,0.0,0.0,cr,0.0,z,cr,360.0,z,cr,360.0,0.0)
   rad = cr + pth
* make top cap for center
   section(cr,0.0,z,rad,0.0,z,rad,360.0,z,cr,360.0,z)
```

```
* make bottom cap for center
   z = 0.0
   section(cr,360.0,z,rad,360.0,z,rad,0.0,z,cr,0.0,z)
* previous height
   ph = hr + hrcz
* previous radius
   pr = rad
******************************************************
* now do layers of twisted channels
lplayer
   m = m + 1
* make last layer partition thickness larger
   NE(m,14) :S(nochng)
   pth = pth * 2.0
   orn = CONVERT((or - pth - cr) / (pth + wave),’INTEGER’)
   or = pth + cr + orn * (pth + wave)
   radiusAT = radiusA + pth
nochng
* this radial layer
* layer’s inner radius
   radius = pr
* layer’s outer radius
   radiusA = radius + wave
* layer’s outer wall radius
   radiusAT = radiusA + pth
* layer’s inner height
   ih = ph
* layer’s outer height
   oh = cos(arcsin(radiusA / hr)) * hr + hrcz
* sound’s exit height
   eh = MIN(oh + wave / 2.0,(ih + oh) / 2.0)
* sound path desired length
   spdl = SQRT( ((radius + radiusA) / 2.0) ** 2 + eh * eh)
* tilt angle of channel
   tilt = 90.0 - arcsin(eh / spdl)
   tiltD = sin(tilt) / cos(tilt)
* calculate total horizontal twist angle
   hta = aa(eh * tiltD * 360.0 / (pi * (radius + radiusA)))
* radial separator thickness increases with tilt angle
   rth = pth / cos(tilt)
* tangential separator thickness
   tth = pth
* vertical angle delta
   vad = d
* vertical height delta
   vd = vad * 2.0 * pi * radius / (360.0 * tiltD)
* channel angle span for walls and overall
   anglei = 360.0 * (wave + rth) / ((radius + radiusA) *
+     pi * cos(tilt))
   angled = CONVERT(anglei,’INTEGER’)
   anglenn = 360 / angled
   angled = aa(360.0 / anglenn)
   anglen = CONVERT(360 / angled,"INTEGER")
   angled = aa(360.0 / anglen)
   anglet = aa(360.0 * rth / ((radius + radius) * pi))
   anglet = d EQ(anglet) :F(noatwarn)
noatwarn
   anglew = angled - anglet
* angle remainder to add d by d
   angler = 360 - anglen * angled
* outward top cap all the way around
   r1 = radiusA;  a1 = 0.0;  z1 = oh
   r2 = radiusAT; a2 = 0.0;  z2 = oh
   r3 = radiusAT; a3 = 360.0; z3 = oh
   r4 = radiusA;  a4 = 360.0; z4 = oh
* cap:
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
```

```
* outward bottom cap all the way around
   r1 = radiusA;  a1 = 360.0; z1 = 0.0
   r2 = radiusAT; a2 = 360.0; z2 = 0.0
   r3 = radiusAT; a3 = 0.0;  z3 = 0.0
   r4 = radiusA;  a4 = 0.0;  z4 = 0.0
* cap:
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
   anglea = 0.0; anglerem = angler
   channel = 0
* do next concentric layer
lparound angle = anglea
   angledi = angled; anglewi = anglew
   EQ(anglerem) :S(noangleadjust)
   angledi = angled + d; anglewi = anglew + d
   anglerem = anglerem - d
noangleadjust
   channel = channel + 1
   za = ih; zb = oh; zc = oh; zd = ih
   zad = oh - vd; zbd = oh - vd; zcd = oh - vd; zdd = oh - vd
   n = 0
* radial wall chunk 1 - clockwise side
   r1 = radius;  a1 = angle;  z1 = za;
   r2 = radiusA; a2 = angle;  z2 = zb
   r3 = radiusA; a3 = angle + hta; z3 = 0.0
   r4 = radius;  a4 = angle + hta; z4 = 0.0
* wall:
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
* radial wall chunk 2 - outer wall
   r1 = radiusA; a1 = angle;     z1 = zb
   r2 = radiusA; a2 = angle + anglewi;  z2 = zc
   r3 = radiusA; a3 = angle + anglewi + hta; z3 = 0.0
   r4 = radiusA; a4 = angle + hta;   z4 = 0.0
* wall:
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
* radial wall chunk 3 - counterclockwise side
   r1 = radiusA; a1 = angle + anglewi;  z1 = zc
   r2 = radius;  a2 = angle + anglewi;   z2 = zd
   r3 = radius;  a3 = angle + hta + anglewi; z3 = 0.0
   r4 = radiusA; a4 = angle + hta + anglewi; z4 = 0.0
* wall:
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
* counterclockwise side bottom cap
r1 = radius;  a1 = angle + hta + anglewi + anglet; z1 = 0.0
r2 = radiusA; a2 = angle + hta + anglewi + anglet; z2 = 0.0
r3 = radiusA; a3 = angle + hta + anglewi;   z3 = 0.0
r4 = radius;  a4 = angle + hta + anglewi;   z4 = 0.0
* wall:
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
* radial wall chunk 4 - inner wall
   r1 = radius; a1 = angle + anglewi;  z1 = zd
   r2 = radius; a2 = angle;     z2 = za
   r3 = radius; a3 = angle + hta;   z3 = 0.0
   r4 = radius; a4 = angle + anglewi + hta; z4 = 0.0
* wall:
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
* side cap inner part
   r1 = radius;  a1 = angle + anglewi; z1 = zd
   r2 = radiusA; a2 = angle + anglewi; z2 = zc
   r3 = radiusA; a3 = angle + angledi; z3 = zc
   r4 = radius;  a4 = angle + angledi; z4 = zd
* cap:
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
   anglea = anglea + angledi
   GE(anglea,360.0) :F(lparound)
* check layer loop
   pr = radiusAT; ph = oh
   LT(radiusA,or) :S(lplayer)
* make outside main wall
```

```
   radius = pr; r1 = radius; a1 = 0.0
   z1 = ph;   r2 = radius; a2 = 0.0
   z2 = ofh + ofh; r3 = radius; a3 = 360.0
   z3 = ofh + ofh; r4 = radius; a4 = 360.0
   z4 = ph
* Vertical outer wall:
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
* make outside main wall fillet
   radius = pr
   r1 = radius;  a1 = 0.0;  z1 = ofh + ofh
   r2 = radius + ofh; a2 = 0.0;  z2 = ofh
   r3 = radius + ofh; a3 = 360.0; z3 = ofh
   r4 = radius;  a4 = 360.0; z4 = ofh + ofh
* Vertical outer wall fillet:
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
* inner radius for bottom
   iradius = radius
* radius starting from fillet
   radius = radius + ofh
* make outer flange
* outer flange wall:
   r1 = ofr; a1 = 0.0;  z1 = ofh
   r2 = ofr; a2 = 0.0;  z2 = 0.0
   r3 = ofr; a3 = 360.0; z3 = 0.0
   r4 = ofr; a4 = 360.0; z4 = ofh
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
* outer flange top:
   r1 = radius; a1 = 0.0;  z1 = ofh
   r2 = ofr;  a2 = 0.0;  z2 = ofh
   r3 = ofr;  a3 = 360.0; z3 = ofh
   r4 = radius; a4 = 360.0; z4 = ofh
   section(r1,a1,z1,r2,a2,z2,r3,a3,z3,r4,a4,z4)
* outer flange bottom:
   section(iradius,a4,0.,r3,a3,0.,r2,a2,0.,iradius,a1,0.)
* make 8 mounting holes
   holea = 0.0
holelp
   mhx = mhc * cos(holea); mhy = mhc * sin(holea)
   angle = 0.0
* making hole at holea
drilllp angled = angle + 5.0
x1 = mhx + mhr * cos(angle); y1 = mhy + mhr * sin(angle);
x2 = mhx + mhr * cos(angle); y2 = mhy + mhr * sin(angle);
x3 = mhx + mhr * cos(angled); y3 = mhy + mhr * sin(angled);
x4 = mhx + mhr * cos(angled); y4 = mhy + mhr * sin(angled);
z1 = 0.0; z2 = ofh; z3 = ofh; z4 = 0.0; x5 = mhx; y5 = mhy
* Vertical center wall of mounting hole:
   rect(x1,y1,z1,x2,y2,z2,x3,y3,z3,x4,y4,z4)
* bottom and top of mounting hole
   facet(x5,y5,z1,x2,y2,z1,x3,y3,z1)
   facet(x5,y5,z2,x3,y3,z2,x2,y2,z2)
   angle = angled LT(angled,360.0) :S(drilllp)
   holea = holea + 45.0 LT(holea,315.0) :S(holelp)
* done with mounting holes
   :(done)
* draw curved surface based on polar coordinate corner points
* center of curvature is around z axiz at d angle increments
* so that all facets match boundaries
section dd = d; dd = -d LT(sa4 - sa1)
   dn = CONVERT(ABS((dd * 0.25 + sa4 - sa1) / d),'INTEGER')
   sr14d = ABS(d / (sa4 - sa1)) * (sr4 - sr1)
   sr23d = ABS(d / (sa3 - sa2)) * (sr3 - sr2)
   sz14d = ABS(d / (sa4 - sa1)) * (sz4 - sz1)
   sz23d = ABS(d / (sa3 - sa2)) * (sz3 - sz2)
   n = 0
sectionflp
   sx1 = sr1 * cos(sa1); sy1 = sr1 * sin(sa1)
   sx2 = sr2 * cos(sa2); sy2 = sr2 * sin(sa2)
   EQ(n,dn - 1) :F(sectionnae)
```

```
   tx3 = sr3 ∗ cos(sa3); ty3 = sr3 ∗ sin(sa3); tz3 = sz3
   tr3 = sr3; ta3 = sa3
   tx4 = sr4 ∗ cos(sa4); ty4 = sr4 ∗ sin(sa4)
   tz4 = sz4; tr4 = sr4; ta4 = sa4
   :(sectionae)
sectionnae
   tx3 = (sr2 + sr23d) ∗ cos(sa2 + dd)
   ty3 = (sr2 + sr23d) ∗ sin(sa2 + dd)
   tz3 = sz2 + sz23d; tr3 = sr2 + sr23d
   ta3 = sa2 + dd
   tx4 = (sr1 + sr14d) ∗ cos(sa1 + dd)
   ty4 = (sr1 + sr14d) ∗ sin(sa1 + dd)
   tz4 = sz1 + sz14d; tr4 = sr1 + sr14d; ta4 = sa1 + dd
sectionae
   EQ(sa1,sa2) :S(sectionrect)
   EQ(sz2,tz3) :S(sectionrect)
   section(sr2,sa2,sz2,tr3,ta3,tz3,tr4,ta4,tz4,sr1,sa1,sz1)
   :(sectionnext)
sectionrect
   rect(sx1,sy1,sz1,sx2,sy2,sz2,tx3,ty3,tz3,tx4,ty4,tz4)
sectionnext
   n = n + 1 LT(n,dn - 1) :F(RETURN)
   sa1 = sa1 + dd; sa2 = sa2 + dd
   sz1 = sz1 + sz14d; sz2 = sz2 + sz23d
   sr1 = sr1 + sr14d; sr2 = sr2 + sr23d
   :(sectionflp)
∗ draw a rectangle using two triangle facets
rect  facet(rx1,ry1,rz1,rx2,ry2,rz2,rx3,ry3,rz3) :F(FRETURN)
   facet(rx1,ry1,rz1,rx3,ry3,rz3,rx4,ry4,rz4) :F(FRETURN)
   :(RETURN)
∗ draw a triangle facet
facet  Vx = fx2 - fx1; Vy = fy2 - fy1; Vz = fz2 - fz1
   Wx = fx3 - fx2; Wy = fy3 - fy2; Wz = fz3 - fz2
   Nx = Vy ∗ Wz - Vz ∗ Wy; Ny = Vz ∗ Wx - Vx ∗ Wz
   Nz = Vx ∗ Wy - Vy ∗ Wx
   N  = SQRT(Nx ∗ Nx + Ny ∗ Ny + Nz ∗ Nz)
   Nx = Nx / N; Ny = Ny / N; Nz = Nz / N
   print3d = "facet normal " PAD(Nx) " " PAD(Ny) " " PAD(Nz)
   print3d = "outer loop"
   print3d = "vertex" PAD((fx1 + offset) ∗ scale)’ ’
+PAD((fy1 + offset) ∗ scale) ’ ’ PAD(fz1 ∗ scale) :F(FRETURN)
   print3d = "vertex " PAD((fx2 + offset) ∗ scale)’ ’
+PAD((fy2 + offset) ∗ scale) ’ ’ PAD(fz2 ∗ scale) :F(FRETURN)
   print3d = "vertex " PAD((fx3 + offset) ∗ scale)’ ’
+PAD((fy3 + offset) ∗ scale) ’ ’ PAD(fz3 ∗ scale) :F(FRETURN)
   print3d = "endloop"; print3d = "endfacet"
   facetcount = facetcount + 1 :(RETURN)
∗ angle adjust : make angle nearest multiple of d
aa  aa = CONVERT((a + 0.5 ∗ d) / d,’INTEGER’) ∗ d :(RETURN)
PAD  PAD = x + 0.0
∗ FreeCAD needs to see a trailing zero after a decimal point
   PAD RPOS(1) "." = ".0"; PAD ’-’ :(RETURN)
∗ last line of STL file:
done  print3d = "endsolid simplicity"
END
```

THE AUTHOR

Viktors Berstis

Viktors Berstis is a Senior Software Engineer at IBM and an IBM Master Inventor with over 180 US patents. He started programming computers in 1965 and his experience at IBM includes architecting the System/38 - AS/400, designing and developing various software and silicon compilers, research on high-level automated integrated circuit design while at the TJ Watson Research Center, and OS/2. He has been the technical lead and scientist for WorldCommunityGrid.org since its start where he helps researchers apply donated spare computing power from computing devices around the world to support projects helping humanity by finding cures for diseases and solving environmental problems. With mathematics, physics, and engineering degrees from the University of Michigan, he is a senior member of the IEEE. His hobbies include posting his piano playing on youtube, designing and building all sorts of scientific gadgets, 3D stereoscopic photography, radio controlled aircraft, cosmology, exploiting solar energy, and 3D printing.