Avoiding sound problems (V1.0) AT AN

Problem:

You shoot against an opposite wall and hear no impact sound.
This problem is caused by level design and it can be avoided by remembering some points during level building.

Some basics first: (sorry, but must be)

While creating the BOA Vis, D3Edit calculates the sound ranges inside the level and put them into a list.
This list is saved with the level and Descent3 uses this list inside the game to create the sounds.

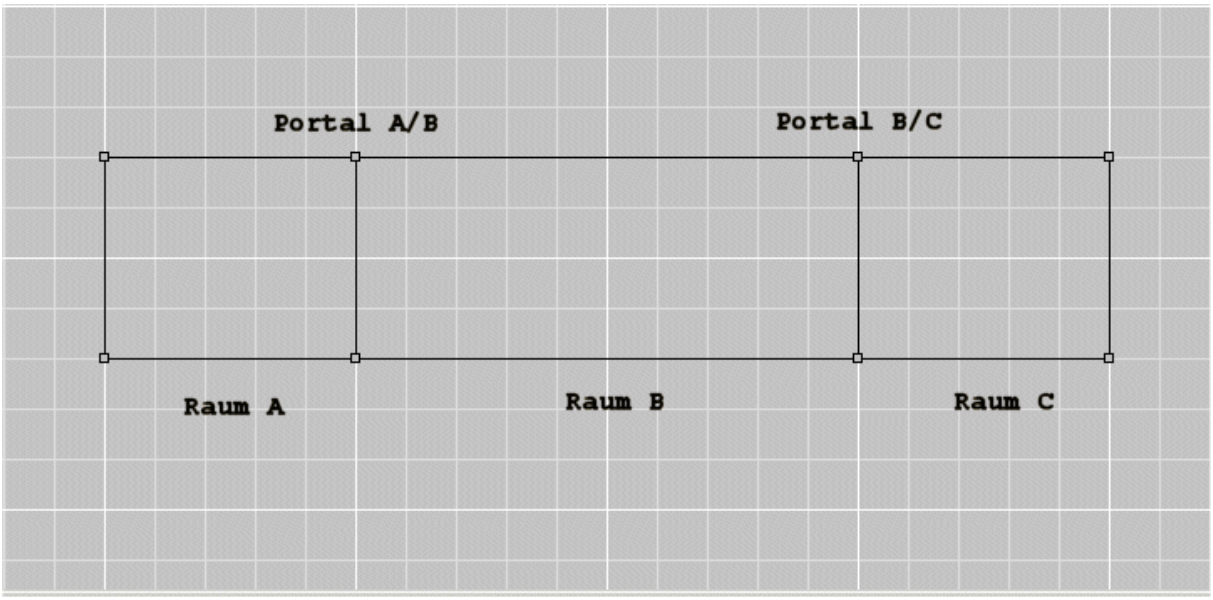
While calculating the distance D3Edit checks for a distance larger than 400 units.
If the distance exceeds this limit, this distance entry is set to invalid.
If Descent3 finds this invalid distance entry it will not generate an impact sound.

How is this distance list calculated?

D3Edit checks for every room what is the most distant room "visible".
While checking this, D3Edit adds the distances between Start- and most distant End-Room.
Then this distance is placed inside the list if valid.

How does that 'looking from room to room' work?

Let's see with an example level. Here we have 3 Rooms in a row (A, B, C) which are connected by portals.
(Room A -> B; Room C -> B; Room B -> A and Room B -> C)
So Room A has *one* Portal, Room B *two* Portals and Room C *one* Portal.



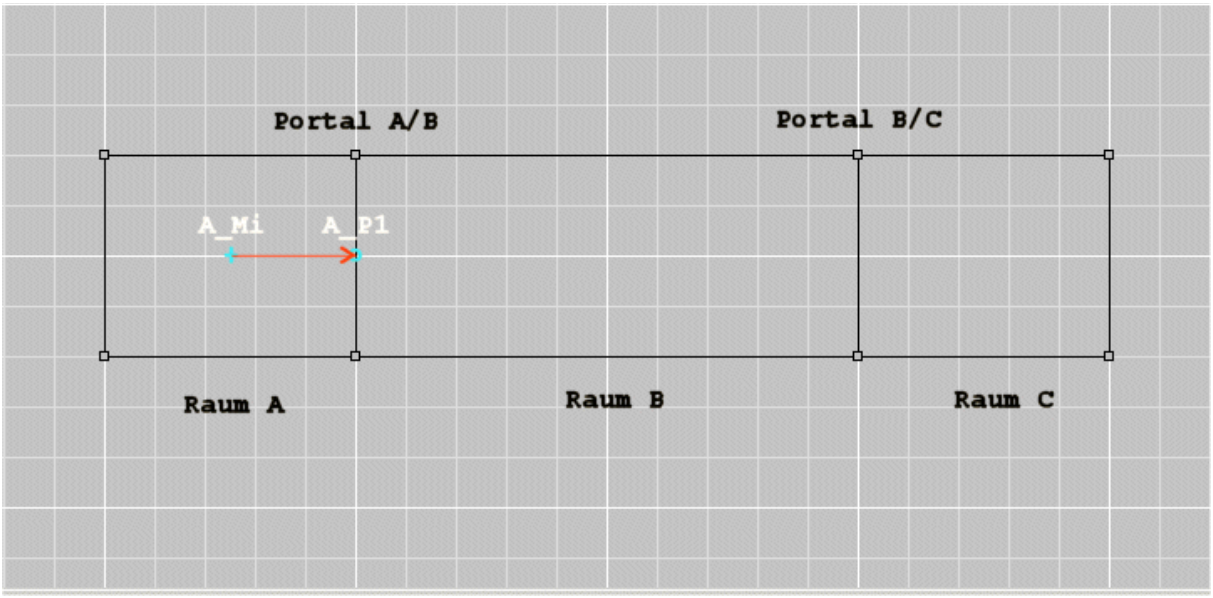
D3Edit calculates for each Room the max line of sight first.
And it puts for every Room (it's the Start-Room) the corresponding End-Room into the table.
In our example level we will get this table:

Start Room	End Room
A	C
B	A
B	C
C	A

How is that -Room to Room distance- calculated now?

Let D3Edit begin with Room A as Start-Room.
Using the table generated above the distance is calculated from the center of Room A to the center of Room C.
First step is calculating the distance between the center of Room A and the first Portal of Room A.

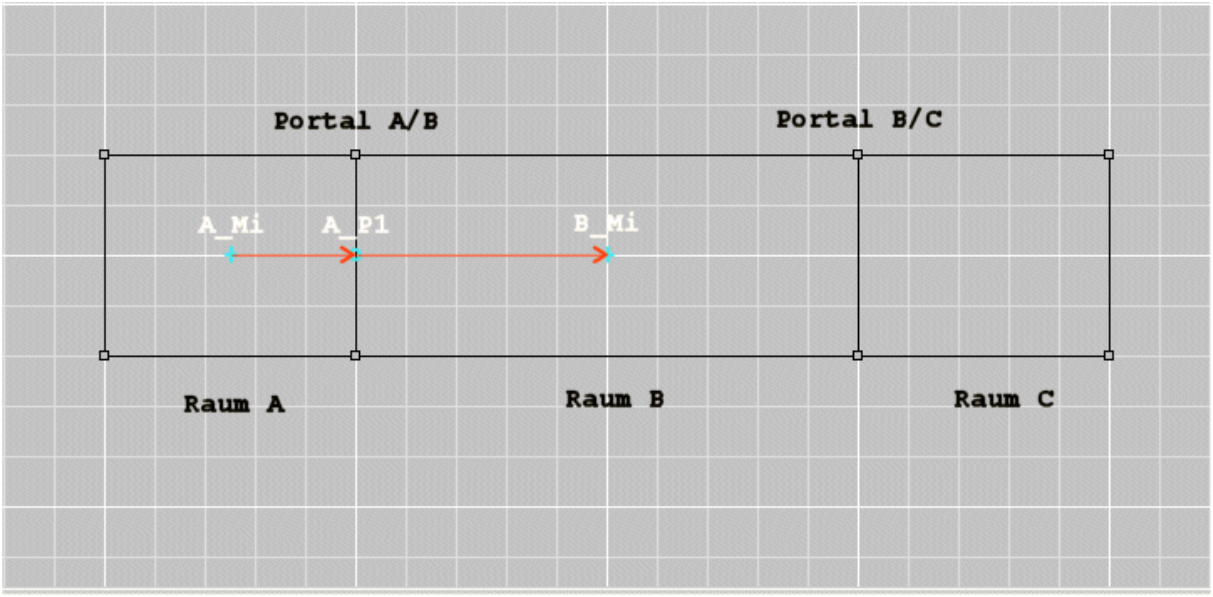
Distance AC is set to 0.
Then a direct line between Room A center towards the center of the portal A/B is taken.



Can't this line hit the Portal face the distance is invalid.
Distance AC = invalid.

If success, the resulting distance is stored.
Distance AC = distance AC + (distance A_Mi -> A_P1)

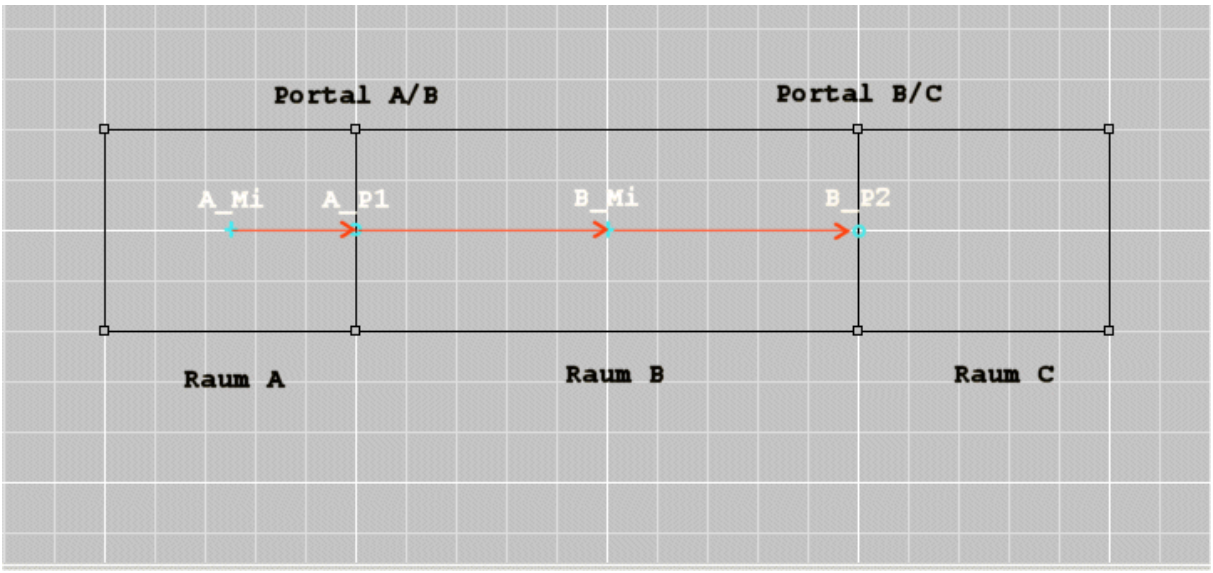
Next step is to take a direct line between Portal-Center A/B to the center of Room B.



Can't this line hit the center of Room B the distance is invalid.
Distance AC = invalid.

If success, the resulting distance is added to the distance stored already.
Distance AC = distance AC + (distance A_P1 -> B_Mi)

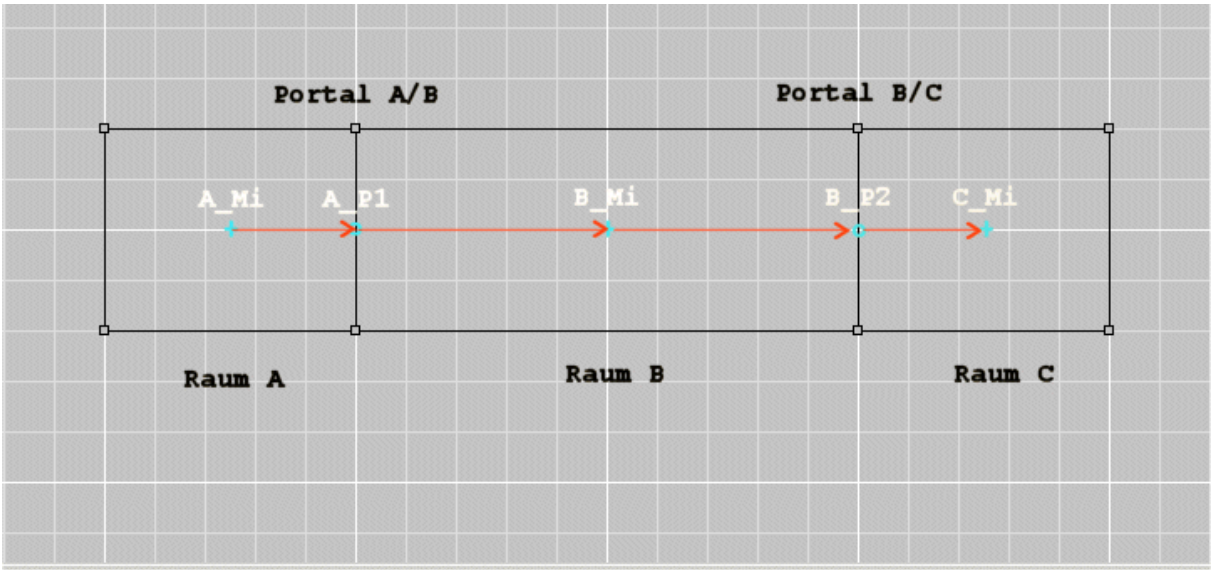
Next step is to take a direct line between Room B center towards Portal-Center B/C.



Can't this line hit the Portal-Center B/C, the distance is invalid.
Distance AC = invalid.

If success, the resulting distance is added to the distance stored already.
Distance AC = distance AC + (distance B_Mi -> B_P2)

Next step is to take a direct line between Portal-Center B/C towards Room-Center C.



Can't this line hit the center of Room C the distance is invalid.
Distance AC = invalid.

If success, the resulting distance is added to the distance stored already.
Distance AC = distance AC + (distance B_P2 -> C_Mi)

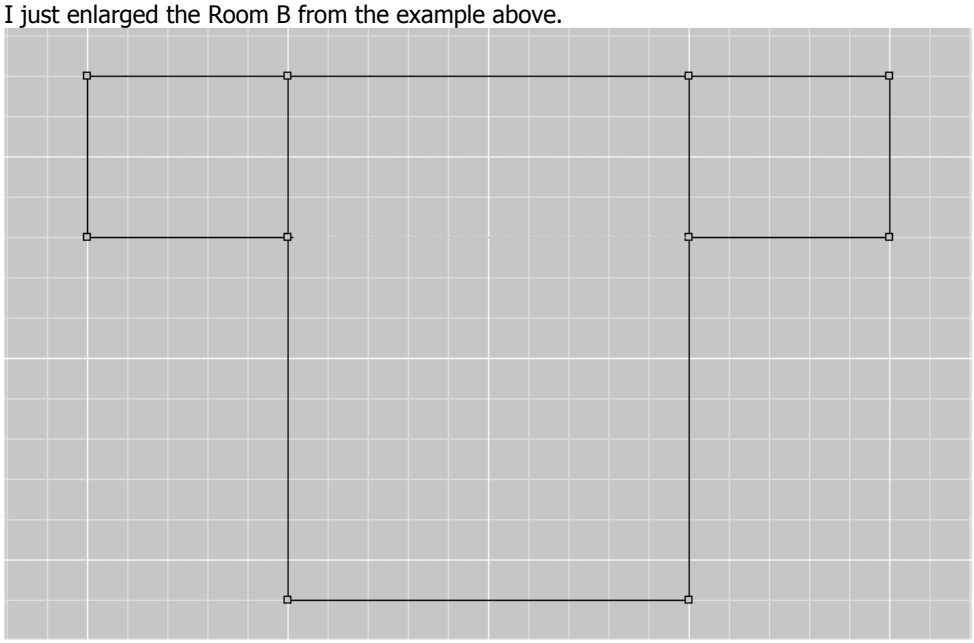
D3Edit now checks if distance AC > 400.
If the result would be yes:
Distance AC = invalid.
Otherwise the resulting distance is stored into the list at A/C.
There is only one portal to check for Room A so calculating distances for Room A is finished now.

D3Edit switches to Room B and calculates, as described for Room A, the distances between B/A and B/C.
If Room C distance calculating is ready we got all max sound distances.
These values are stored inside the level while saving it. If Descent3 loads the level these values are used.

OK, but where is the problem now?

In the example above we could hear an impact by shooting from Room A to Room C if the calculated distance is < 400. Otherwise not. (too far away to hear something).

But have a look to this level design now:
I just enlarged the Room B from the example above.

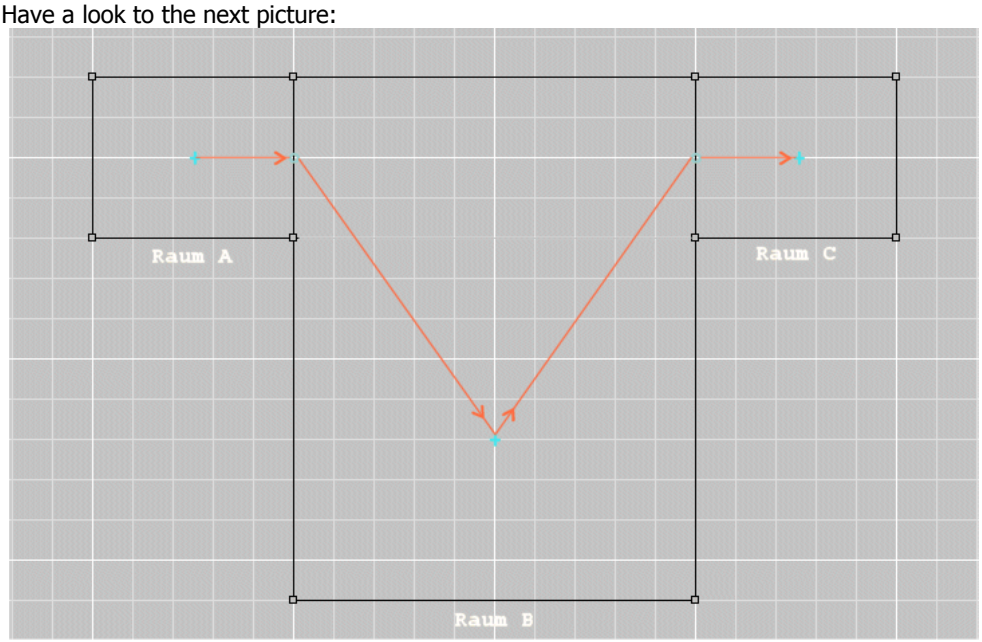


Same as before, we shoot from A to C but.. we can't hear the impact from Room C now!
A second shot, a direct hit on the floor before the Portal B/C is, and we can hear the impact again.
Now we fly into Room B and shoot into Room C. We hear the impact.
Now we shoot back, from Room C to Room A, no impact hearable again!
A second shot direct in front of Portal B/A is hearable, fly into Room B shoot into Room A too.

What happens, the distance between Room A and Room C didn't changed in this design. So what?

Time to remember that we can measure distances inside Descent3 by using the Massdriver.
So we look from Room A to Room C and (i.e.) we see a 250 units distance.
This is absolutely ok, it's below the magic 400 units border but why can't we hear the impact?

The answer is really simple knowing the way the max sound distances are calculated.
Have a look to the next picture:



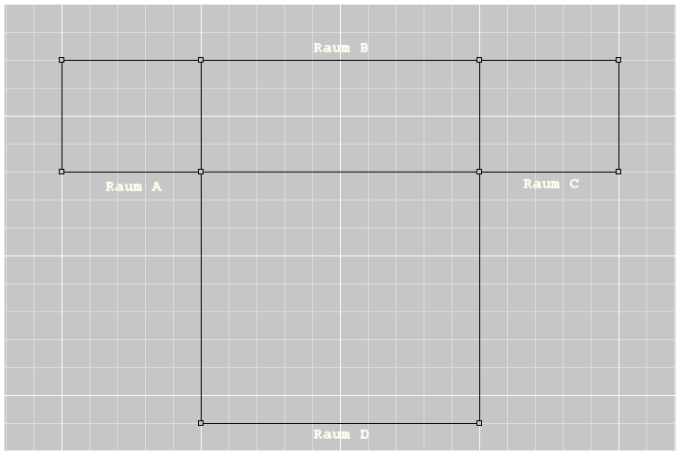
I moved the center of Room B a little more down to show the problem much better.

Do you remember? Distances are calculated from Room center to next Portals.
Well that's exactly the problem now. Although the direct line between Room A and Room C inside this example is just 250 units far, D3Edit calculates more than the doubled distance, as can easily be seen inside the picture.
This is why the distance is more than 400 units. Distance AC = invalid.
And this causes no impact sound.

Any solution?

Yes there is one. We must change our level design.
Simple way is to cut Room B and produce a new Room D which we attach to Room B then.

This picture shows the changed level:

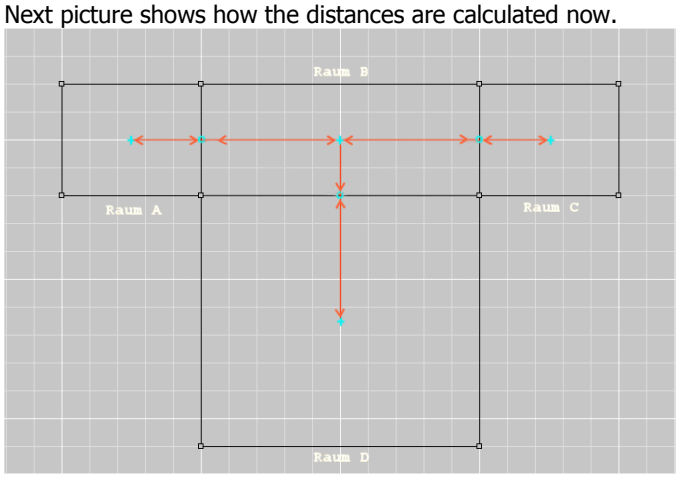


D3Edit generates the following new sight table for this level:

Start Room	End Room
A	C
A	D
B	A
B	C
B	D
C	A
C	D

Inside this level we can hear an impact by shooting from A to C and C to A now.
Inside the game we wouldn't see any difference in level design, the new level looks the same as the first one.

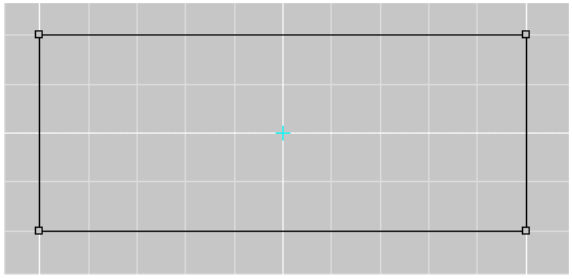
Next picture shows how the distances are calculated now.



It works because the direct line between Room A and Room C is less than max 400 units now.

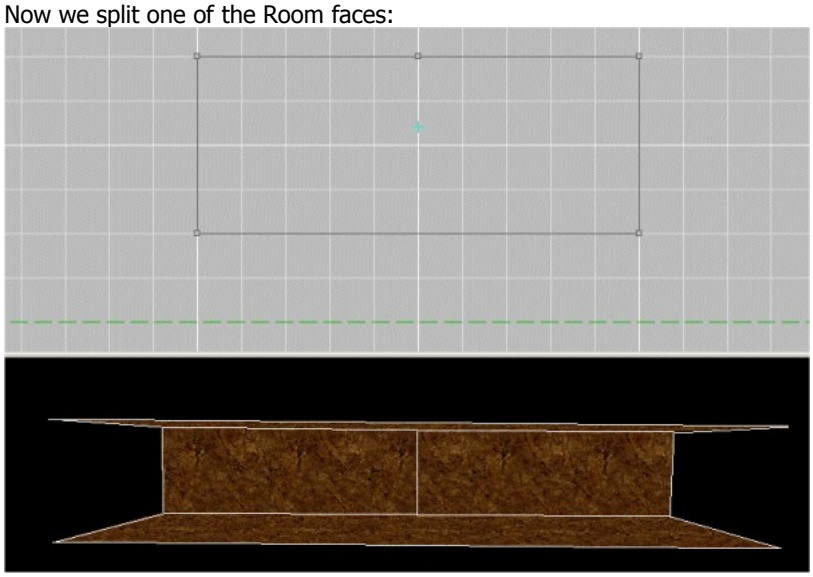
One more problem...

One D3Edit function isn't often used although it could prevent sound problems.
It's the -Show Room Center- function.
Is this turned on, we can see a little cross where Descent3 sees the Room center.
You can find it inside all Ortho Views and it shows the mass centre.
Why do I called it 'mass centre' and not Room Center or middle now?
Have a look to the next pictures:



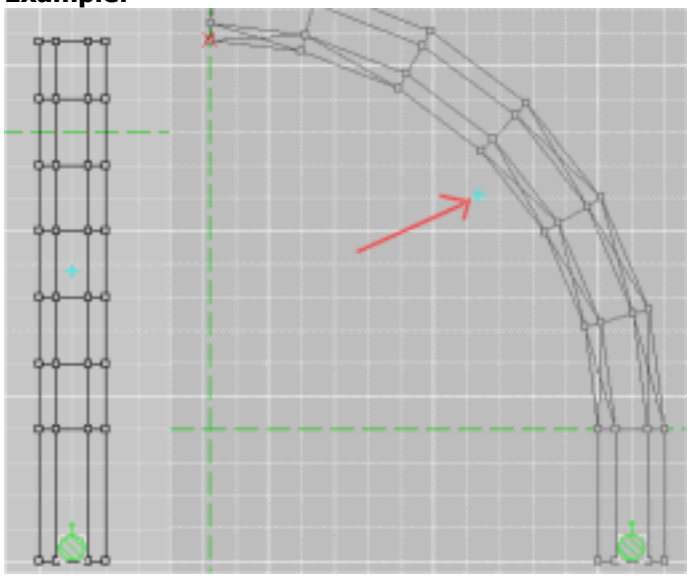
Inside this Room the 'mass centre' is at the center/middle as expected.

Now we split one of the Room faces:



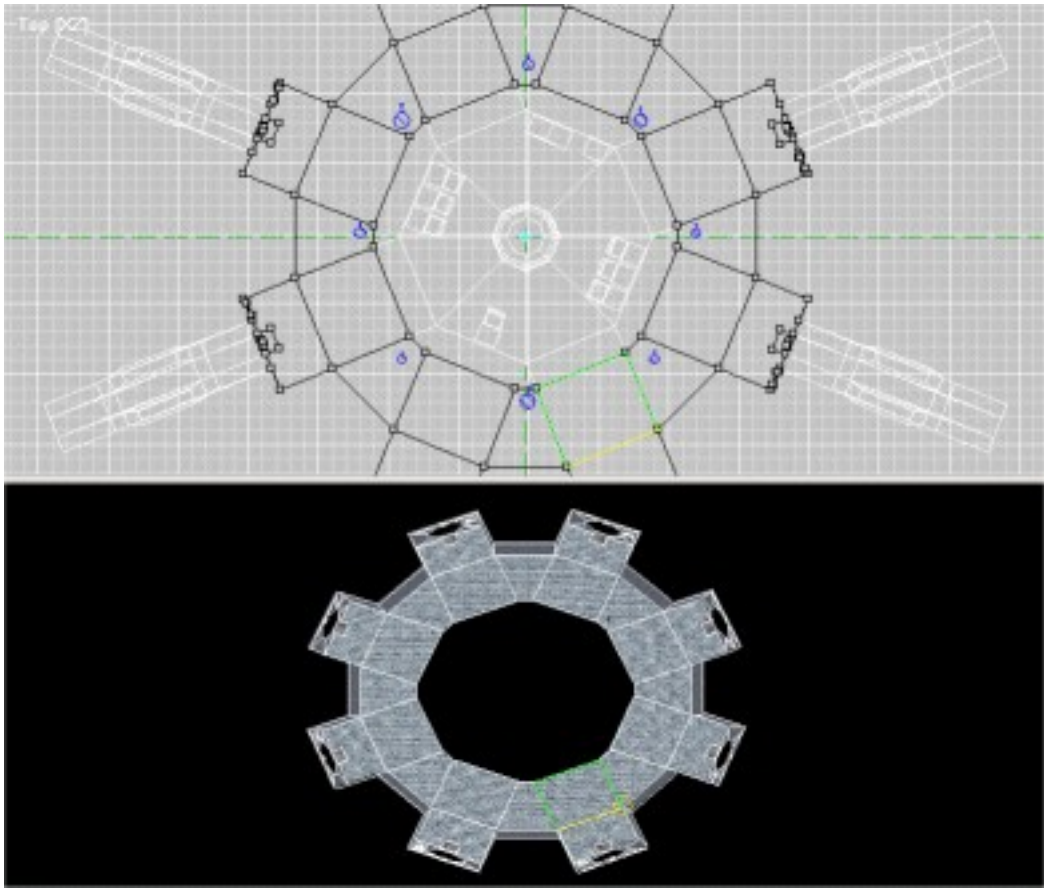
Although the room has the same shape as before the 'mass centre' moves towards the new vertices.
Calculating the Room center is more calculating mass.
A face with lot's of verts pulls the 'mass centre' to it.
This feature may produce sound faults or help to wipe them out.

Example:



When we bent this tunnel the 'mass centre' moved outside the tunnel. (red arrow)
This is fatal for the sound calculation. We would not have a direct line from each Portal towards the Room Center now. One solution would be to cut this bended tunnel into several pieces. But when the distance from 'mass centre' to the inside of the tunnel is not too large we can help us by inserting more vertices at the opposite side of the tunnel to move the 'mass centre'.
Other way could be to insert a monitor and using the function - Merge Object into Room- then.
The resulting new Faces and Verts could move the 'mass centre' into the tunnel again and the sound will work again.

A nice example about not working sounds:



The 'mass centre' is perfectly in the middle, but not inside the correct room ;)

Portals:

The newer D3Edit versions bear in mind that Portal textures must be porous to prevent sound problems inside a level, so the Palmleaf1 texture is set to Portal faces automatically.
This is the only one out of the Descent3 stock textures which is porous for lines of sight and enables correct sound calculation. If you plan to cut off the sound behind a portal you have to set a different texture onto that Portal face(s).
Keep in mind that the same Portal texture situation is relevant to BNodes too.