Andrea Padula

REPORT

The first filter that I created is the VR complex. During the the semester we got the opportunity to create a VR filter. However, the performance of this filter were very slow and since there weren't any constraints in the computational speed I applied very little to not existing optimization. Therefore, I decided to actually create another VR complex filter, but this time try to optimize as much as possible using the algorithm found in the paper that I presented in class. In the paper the VR complex was calculated mainly with 3 methods I didn't implemented with all three, but just with one the Incremental VR.
The paper is very precise and direct in saying that to to construct a proper VR complex, it is better to divide its construction in two phases:

1. Compute a neighborhood graph
2. Expand the graph to include higher-dimensional simplices

In the first part I tried different ways to implement it.
The first idea was to basically loop through our vertices and see which vertices are within the range. In this first idea there wasn't any specific optimization in finding the vertices, but the optimization was done to help the the second phase of the construction of the VR.
The second idea that I had was to create a linked list instead of a vector in order to optimize the performance of the code. However, even though the performance were better in the first part, by 3% to 5% , the second part suffered by a lost of performance of 10-15%. It is important to underline the fact that in my testing most of the times the second phase of the construction of a VR complex is actually the one that takes longer. The third that I did was to implement a spacing hashing, which actually is supposed to speed up the performance by a lot. However, due to time constraints and finals in other class I wasn't able to successfully implement this option. Therefore for phase 1 I compute the graph basically by looping through each vertex and check if they are within the radius given to the user.
Here the result in performance of the filter in comparison with the filter not optimize
Radius 0.25

FILENAME	**GRAPH in sec**			**VR EXPANSION in sec**			**TOTAL TIME**	
	Slow	**Fast**		**Slow**	**Fast**		**Slow**	**Fast**
Small Annulus	0.0003	0.0001		0.004	0.002		0.004	0.002
Annulus	0.01	0.004		8.3	16.8		8.3	16.8
Torus	0.04	0.02		0.51	0.03		0.56	0.06
Dragon	47.1	32.6		38.9	0.01		85.9	32.6
shape2	113.8	7.76		X	X		X	X

Radius 0.5

FILENAME	GRAPH in sec			VR EXPANSION in sec			TOTAL TIME	
	Slow	Fast		Slow	Fast		Slow	Fast
Small Annulus	0.0006	0.0001		0.07	0.06		0.07	0.06
Annulus	0.02	0.004		72.8	206.9		72.8	206.9
Torus	0.05	0.03		9.2	3.6		9.2	3.6
Dragon	46.4	31.9		232.6	0.43		279.0	32.3

Radius 0.7

FILENAME	GRAPH in sec			VR EXPANSION in sec			TOTAL TIME	
	Slow	Fast		Slow	Fast		Slow	Fast
Small Annulus	0.0009	0.0002		0.13	0.25		0.13	0.25
Annulus	0.03	0.005		342.01	1000.3		342.0	1000.3
Torus	0.07	0.03		40.4	30.1		40.5	30.2
Dragon	47.1	30.1		751.4	2.99		798.5	33.1

PART2

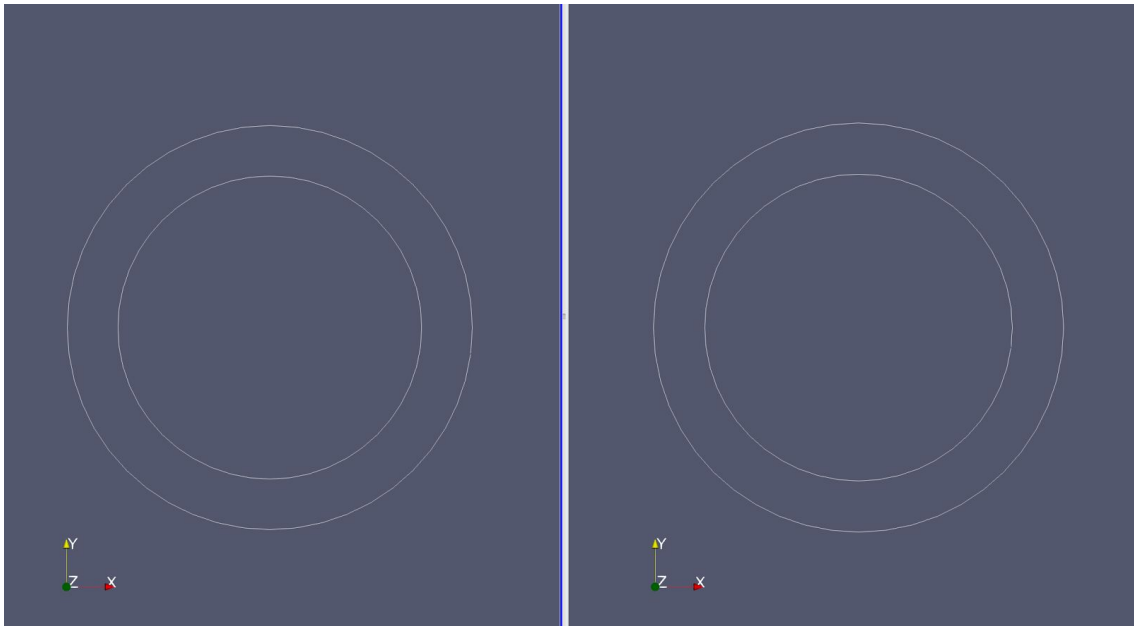
CECH COMPLEX VS VR COMPLEX

The second filter that I construct for my final project is the Cech complex. This filter was definitely harder to construct compare to the VR complex, mostly because we didn't implement it during the class, but also because I consider the actual computation to calculate the Cech complex a little bit more difficult to understand.

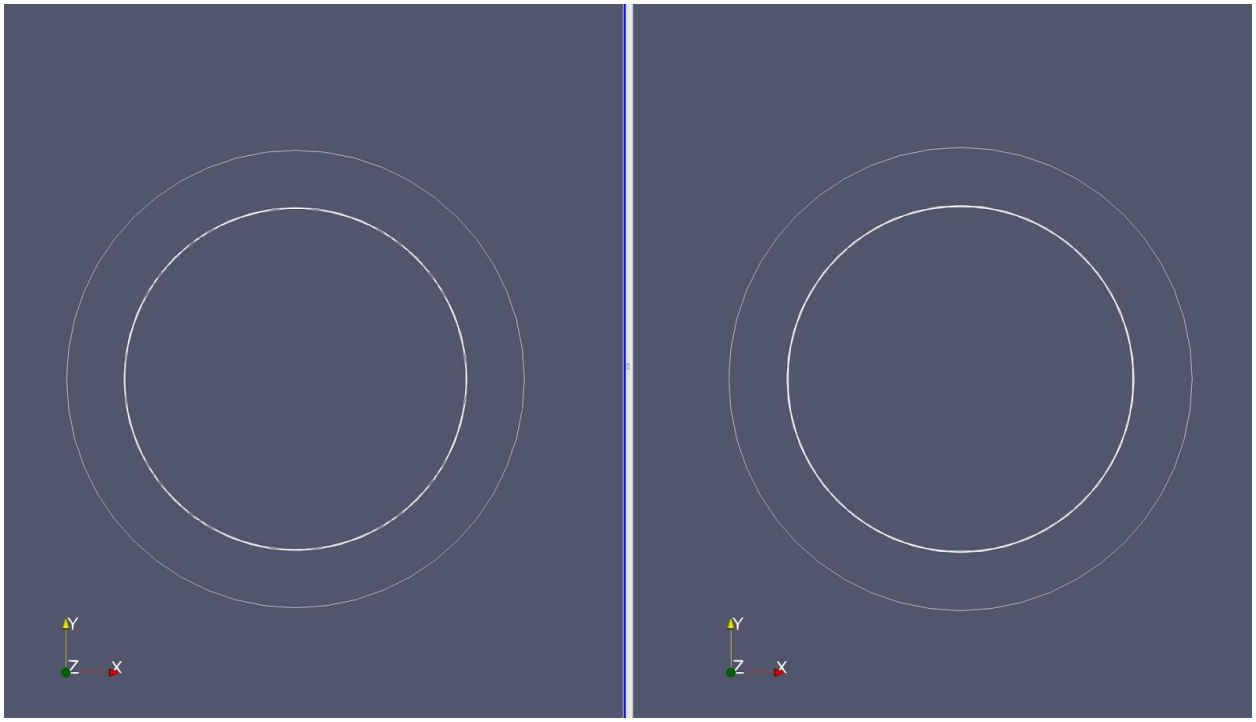
My implementation followed the implementation used by the other paper that I presented in class. This paper is not trying to be particularly efficient or super fast but it just trying to construct the cech complex. It is important to underline the fact that the author of the paper used this code for just 2d implementation, but they assure that this code will work for higher dimension and I confirm that that's the case. To experiment this filter I basically used the question for the assignment 2 but used them with the VR complex. Moreover in each experimentation I tried to compare the result that I got with the VR complex to kind see the similarities and differences between the complex.

The first experiment that I did was with the small annulus to see the difference visually.

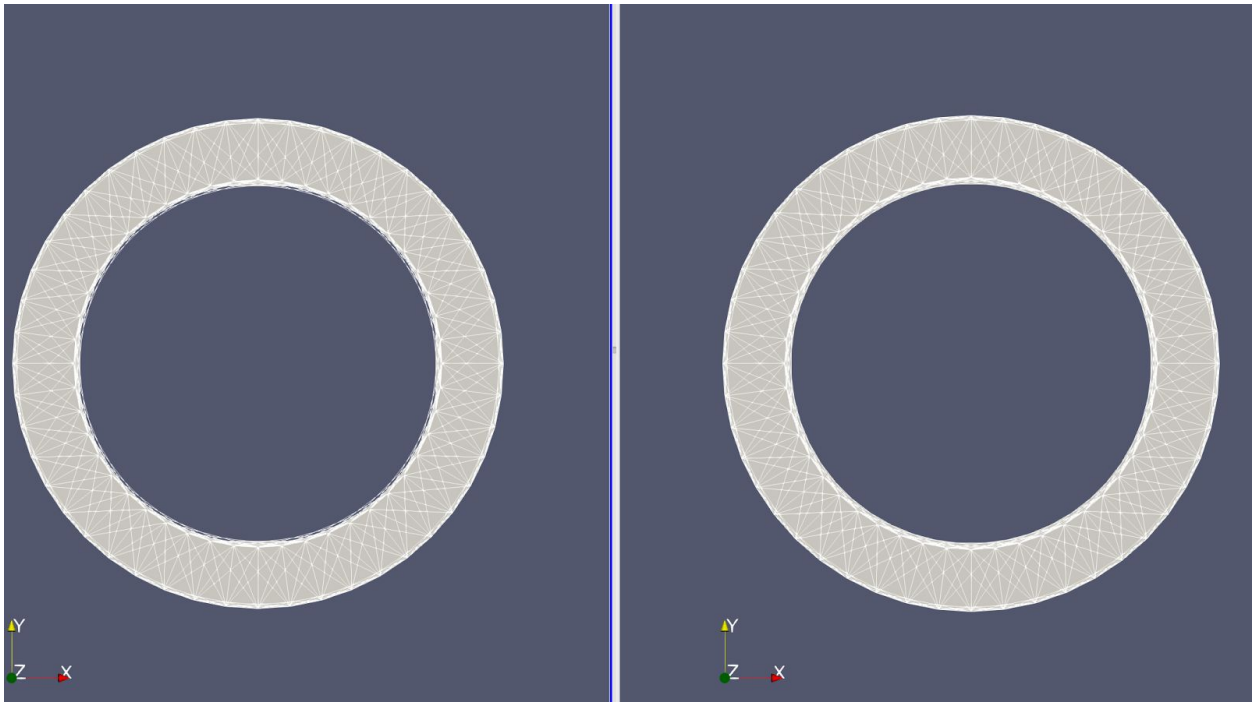
I initially set the radius to 0.8, but no visual difference was found, in fact they pretty much look the same



Then I decided to increase the size of the radius to 0.1, but still no difference



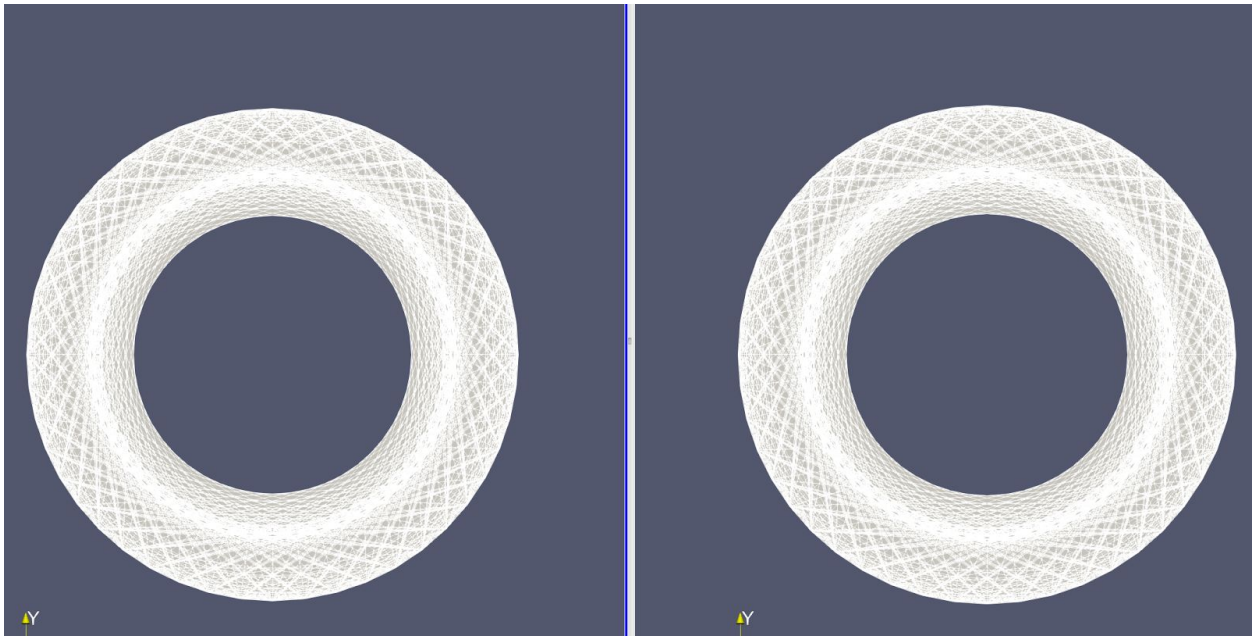
With radius 0.2



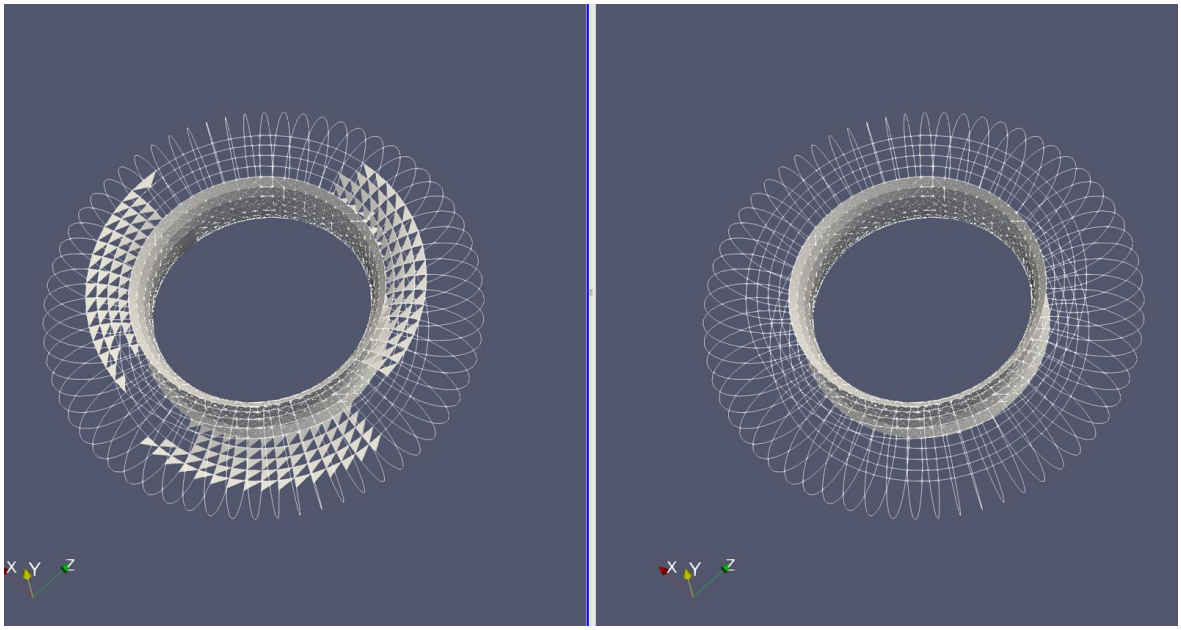
Since with dataset I wasn't really able to tell major difference visually then I decided to use look at the number of cells in the VR and cech complex. I discovered that the VR contains more cells than the cech. In fact here we are looking at the

VR complex 107824 cells

Cech complex 98216 cells
To create this difference I set the radius to 0.5, and still visually no real difference was found



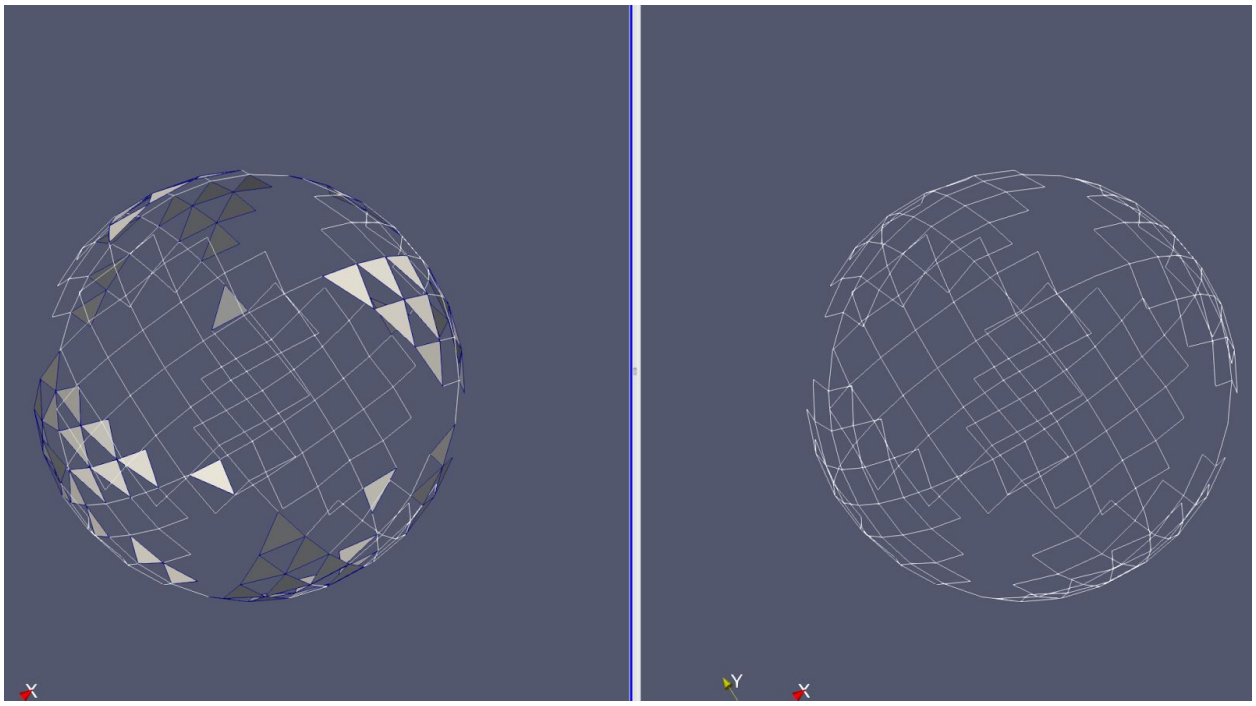
I decided to move my experimentation to a more interesting data set, the torus. The visual difference in this case is definitely present. By setting the radius to 0.15



(right cech,left vr)
As we can see we can already see some triangles on the top of our torus in the cech complex, but in the VR we are not including those cells yet.
In this case we have more cells in Cech complex .

By setting the radius to 0.22 the torus is complete and we can see not a lot of visual difference. In fact the torus is complete in both complex, but the number of cells is slightly higher in the cech complex.
VR complex 39400 cells
Cech complex 44020 cells

Another data set of interested is the sphere point data. This data set with the torus underline the best the difference between the two complex.



By setting the radius to 0.13 the Cech complex on the right is already constructing triangles where in the VR we just have edge. Even in this case the number of cell is greater in the Cech complex.

Result by setting the radius to 0.15