# Presentation

## Title

Data structures for implicitly encoded n-Gmap-based pyramids.

## Who am I?

Information about me, a photo and e-mail.

## Overview n-Gmaps/Pyramids

What is it?

What are the involutions? How do they work? Do I make difference between implicitly and explicitly involutions?

Find an image/draw of n-Gmap to insert in the presentation. Assegnare colore sfondo immagine lo stesso della diapositiva.

## Problem

One of the most important problems is the memory occupied to save information about n-Gmap.

Give an example of the problem related to the memory occupied if we worked on the original image not converted in Gmap (also an image is useful to understand better).

The straightforward way to implement involutions is an array storing for each dart its involution-counterparts. This, however, turns to be (1) prohibitive for huge maps resulting from large volumes and (2) not necessary given that a large portion of the involutions can be encoded implicitly.

## The goal

The goal is to propose and implement a data structure for storage and processing huge (50 to 400 Gigadarts) combinatorial pyramids on supercomputers provided by Vienna Scientific Cluster.

Should I insert a paragraph where I explain why I want to use the bidict? In particular the problems I want to resolve using bidict or something like that.

## Future works

1. Implementation of a data structure can provide information in *O(1)*
2. Analysis of the state of the art
3. Using the Morton code to encoding the dart (what is the advantage of the Morton code?)
   1. In 2 and 3-D
4. Implementation of the relation between darts and levels using the bidict data structure that is better than the usual dict we use in python because allow us to maintain the relationship in both directions.
5. Computing in parallel the future implementation

### Morton code

Vantaggi

* in primo luogo separare i bit meno significativi in c\_x e c\_y da N-1
* in secondo luogo combinarli in un codice Morton.

LUT 3D

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 |  |  | … | 3D |
| α0 |  |  |  |  |  |  |  |  |
| α1 |  |  |  |  |  |  |  |  |
| α2 |  |  |  |  |  |  |  |  |
| α3 |  |  |  |  |  |  |  |  |

LUT 2D

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 |  |  | … | 2D |
| α0 |  |  |  |  |  |  |  |  |
| α1 |  |  |  |  |  |  |  |  |
| α2 |  |  |  |  |  |  |  |  |

## Slide 1

Hello everyone, today I am going to talk about my master thesis’s work on *Data structures for implicitly encoded n-Gmap-based pyramids*.

## Slide 2

You are wondering who I am, so, I am Carmine Napolano, I am 23 and I am studying computer science at the University of Salerno. I am here in order to work on my master thesis at the PRIP lab. Moreover, you can find my e-mail contact on this slide.

## Slide 3

So, in order to better understand the topic, I have prepared a brief overview of the G-map, just the notions.

## Slide 4

Basically, an n-Gmap is a combinatorial data structure allowing to describe n-dimensional orientable/nonorientable quasi-manifold with or without boundary and is defined by a set of darts, that is the basic element, and a set of involutions, that relate the darts to each other.

On the left, we can see three examples of n-Gmap. In particular, a 1-Gmap where we can identify vertices and edges; a 2-Gmap where we can identify vertices, edges and faces; eventually we have a 3-Gmap where we can identify vertices, edges, faces and volumes.

## Slide 5

Ok, now I can show you which are the problems.

## Slide 6

I have put on this slide an example that professor Jiri has shown us the last time to better understand it.

If we consider a level-0 of a 3-Gmap that has dimensions like 1000x1000x1000 voxels, we have to memorize about 1.6 TB, only for the level-0, I want to mark this concept.

## Slide 7

So, the next step is not to store the information about the level-0 because the involutions are implicitly, but we’ll have to store the explicit involutions due to contraction and removal operations.

In this way, securely the amount of data that we have to memorize is less than in the previous case.

## Slide 8

The straightforward(easy) way to implement involutions is an array storing for each dart its involution-counterparts. Basically, the LUTs in the slide are graphic examples of that.

But, as I said previously, we do not need to memorize the information about level-0, and it is prohibitive for huge maps.

## Slide 9

Now let see the goal that I’ll have to reach.

## Slide 10

This is the implementation of a data structure to store and process a huge amount of darts, also with the help of the Vienna Scientific Cluster.

## Slide 11

At first, analysis of the state of the art and the work done of the PRIP lab. After this, I’ll analyse the available data structures and eventually I’ll implement a script with the CGAL library. For future work I think is important to study the Morton code and check if it is useful for encoding Gmap.

## Slide 12

This is my work schedule. I am thinking of proceeding as follow.

## Slide 13

Thank you for your attention and time! Any questions? Advice?

## Note

* CGAL is not easy to implement
  + I won’t convert the whole python script to C++ (that is impossible, I don’t have enough time), but I will use the C++ script as an interface of the python one.
* Analysis of the work at the start of the project (better clarify)
* Analysis of the intermediated results to understand if I am walking on the right way or I have to choose another way
* Not only implementation
* Morton code is good for 2D-3D and maybe for the 4D but is not applicable for the next higher dimensions.
  + Maybe I don’t need to study and spend time with the Morton code
* The challenge can be how I have to choose the edge to removal/contract for having at the next level still implicitly involutions.
  + Find a way to do that, also with a small example (-> image of small resolution)
* Have some tests to check if I can obtain the expected results.
* Ask Majid if he knows a way to make involutions implicitly also the other levels of the pyramids. So, in that way, I can reduce the volume of data to store in memory. The questions are:
  + What is the memory occupied? Is it convenient for the scope?
  + Or is it convenient to store only the upper level of the pyramid? But I don’t think this solution is valid because, if I have understood correctly, I will have to store the involutions that link darts through the levels.