

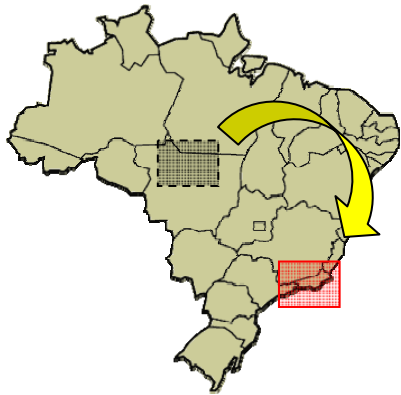


Continuous Interaction with TDK: Improving the User Experience in Terralib

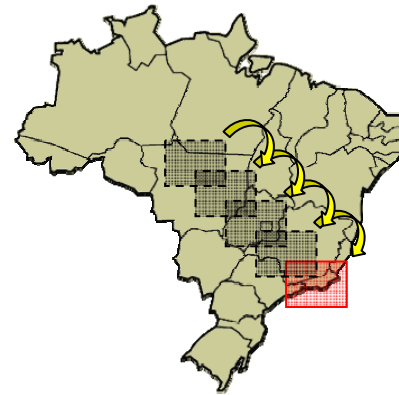
Tecgraf - Computer Graphics Technology Group,
Pontifical Catholic University of Rio de Janeiro (PUC-RJ)

Authors: Marcelo Gomes Metello
Mário de Sá Vera
Melissa Lemos
Leone Pereira Masiero
Marcelo Tilio Monteiro de Carvalho

Discrete Interaction vs Continuous Interaction



Discrete Interaction



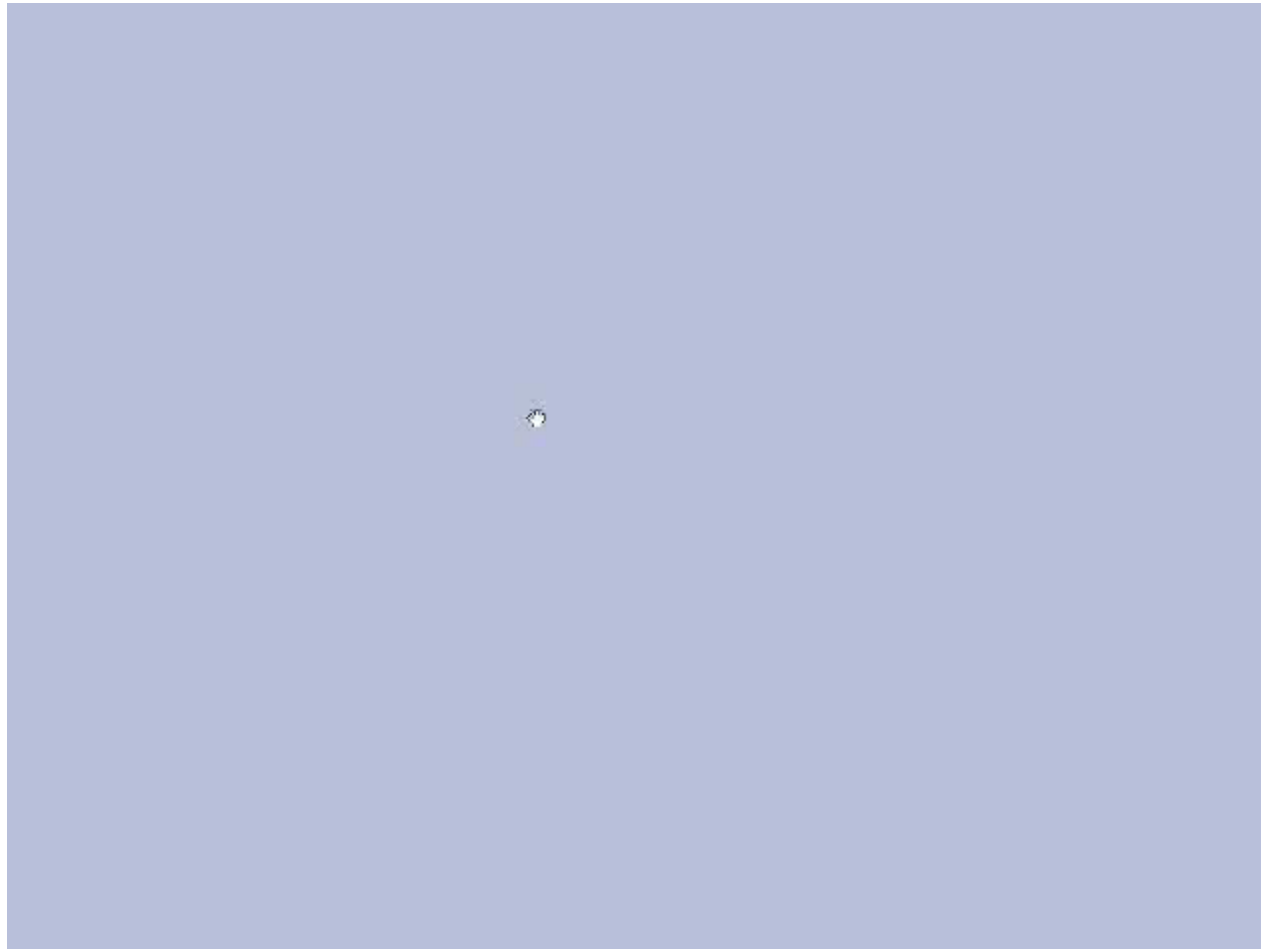
Continuous Interaction

Continuous
Interaction

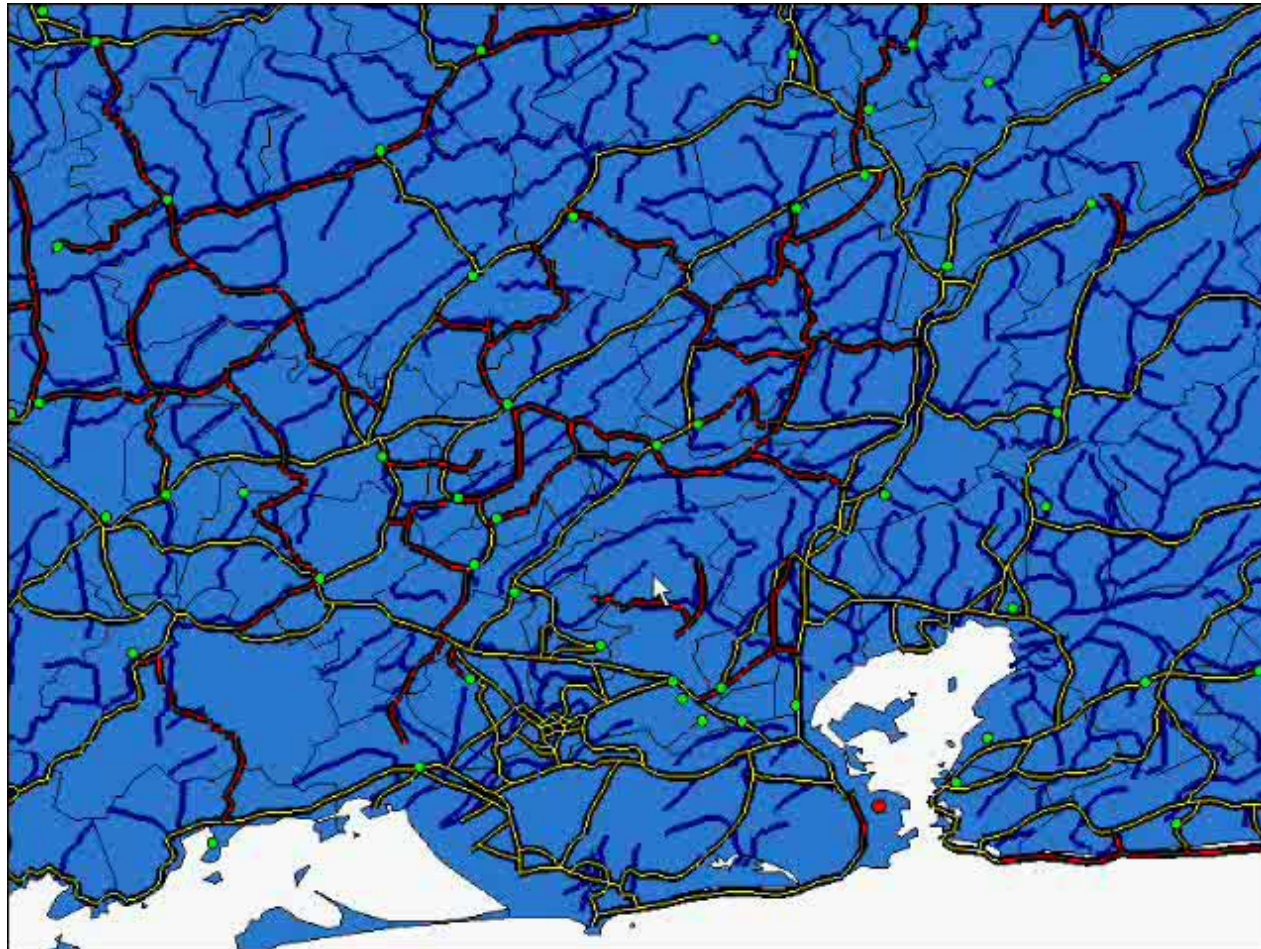
=

No discontinuous
changes in the
visualization window

Discrete Interaction



Continuous Interaction





What are the requirements
for this kind of interaction?

Fast map displaying!
(at least 6 frames per second)

Why pre-rendering does not work

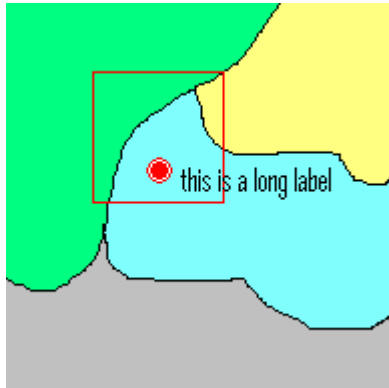
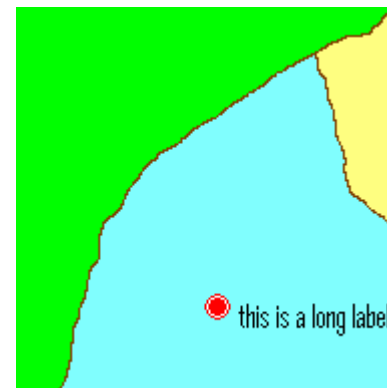


Image stretching
→



Not good!

With pre-rendering we do not get smooth zooming. Therefore, for our goals, map displaying does mean map rendering.



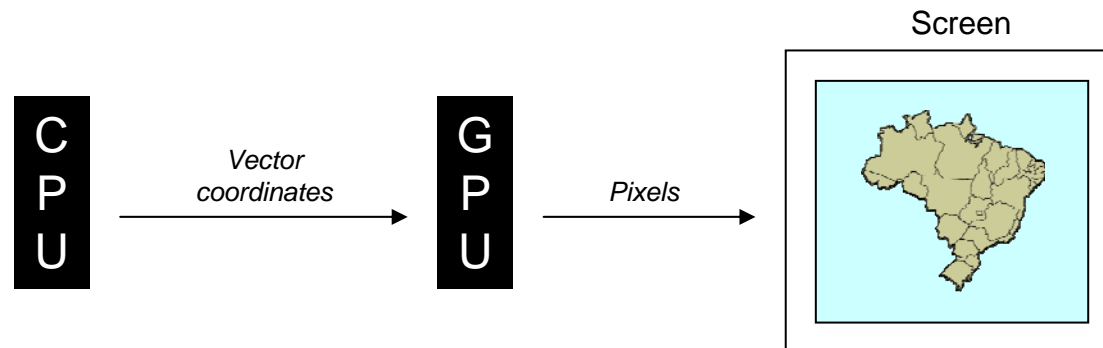
Correct image



How fast can we render?

- 6 fps \sim 0.17s / frame
- At that rate, today's average graphics cards can render up to a few million coordinates
- Fortunately, a few million coordinates is a reasonable number for GIS visualization
- Must use the GPU's rendering power

- Stands for *Graphics Processing Unit*
- It is a processor specialized in graphics rendering operations
- Found on graphics cards
- You may not know it but yes, you have one in your machine





Things that cannot be done at rendering time:

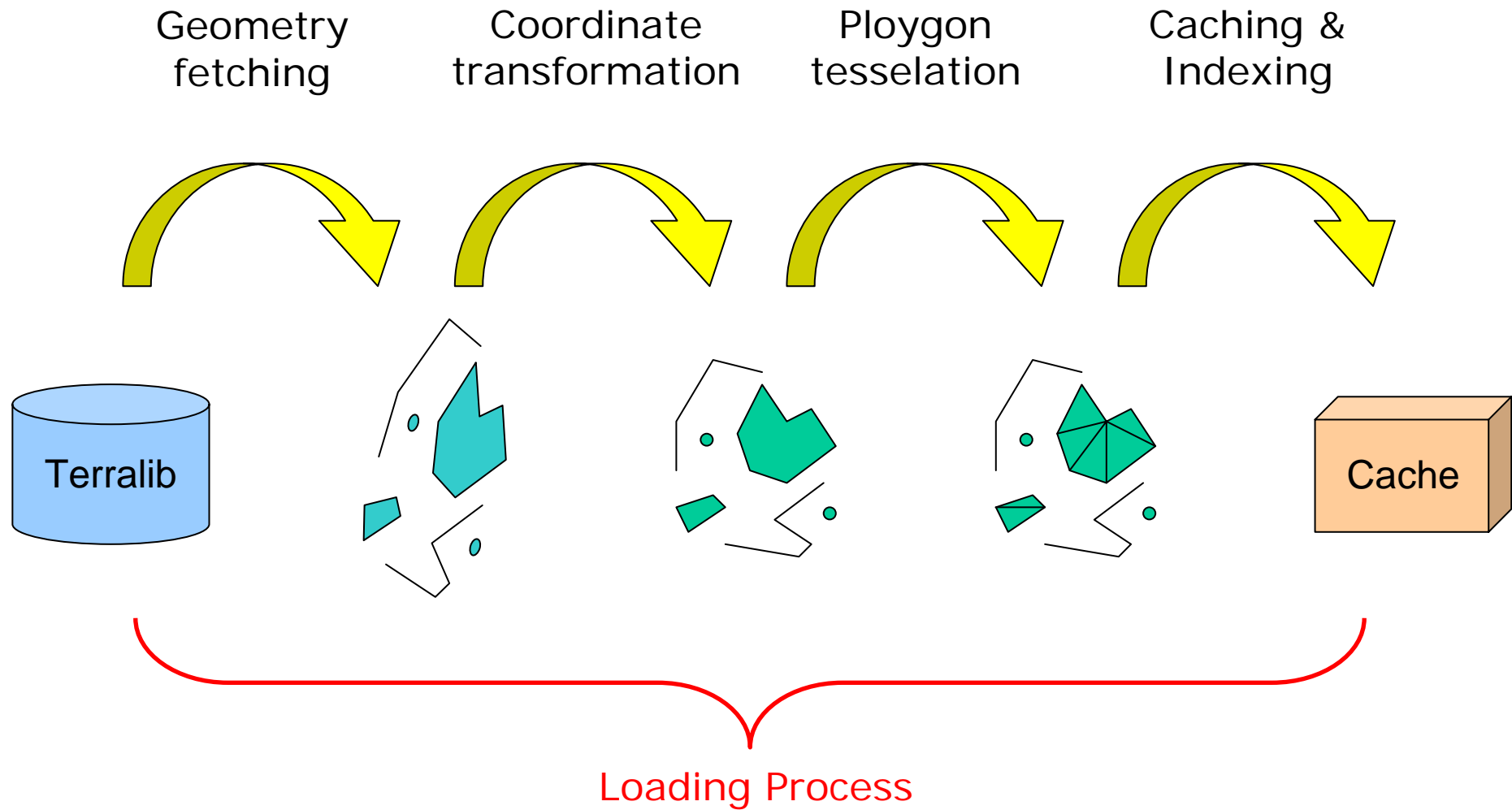
- Database accesses
- File accesses (just a few)
- Reference system coordinate transformations
- Polygon rendering (only triangles)

What is the solution?

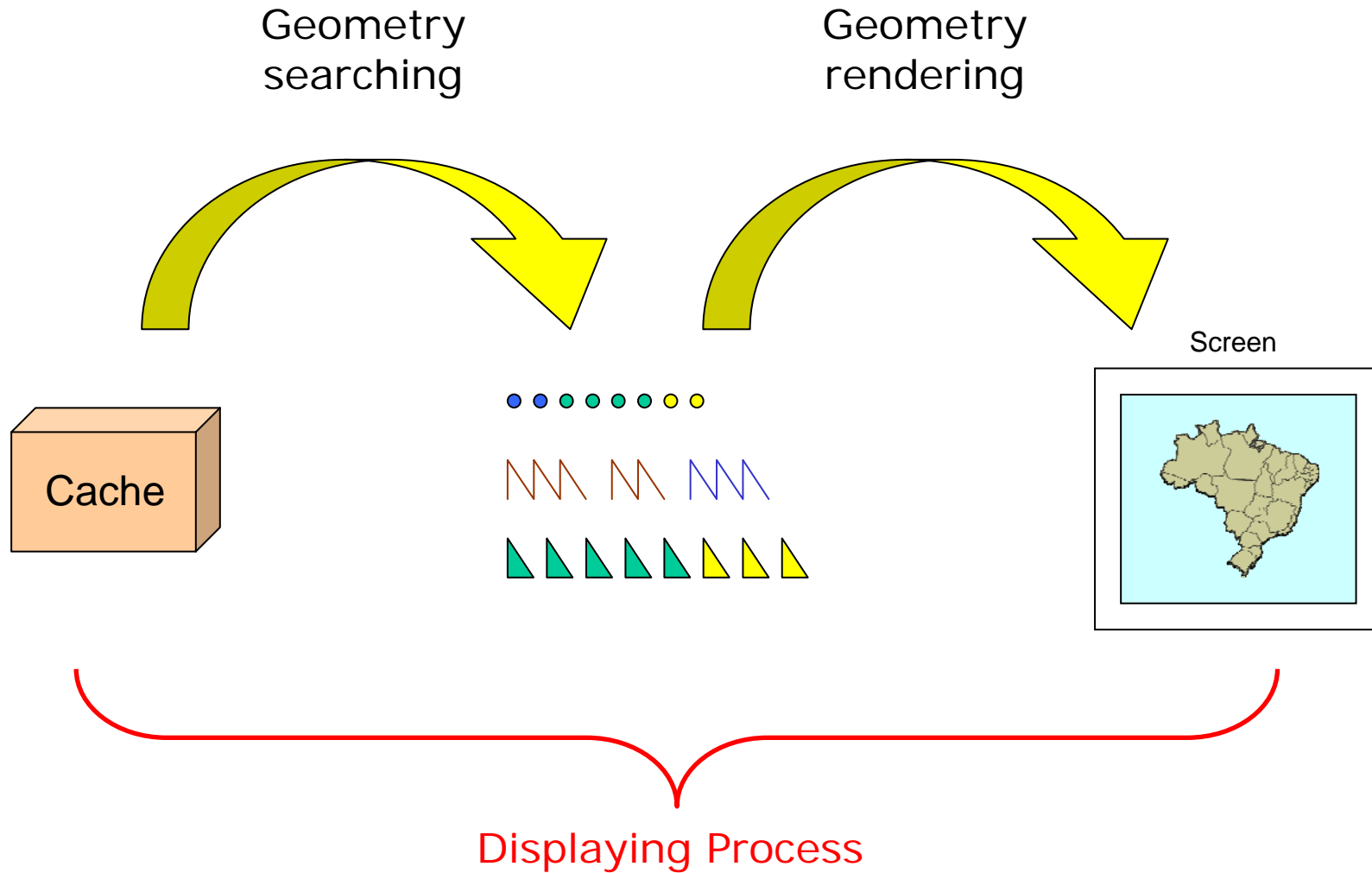


- Render from memory through a caching mechanism
- Preprocess the data (coordinate transformation and polygon tessellation)
- Render only the portion of the map that has been loaded so far

System Architecture

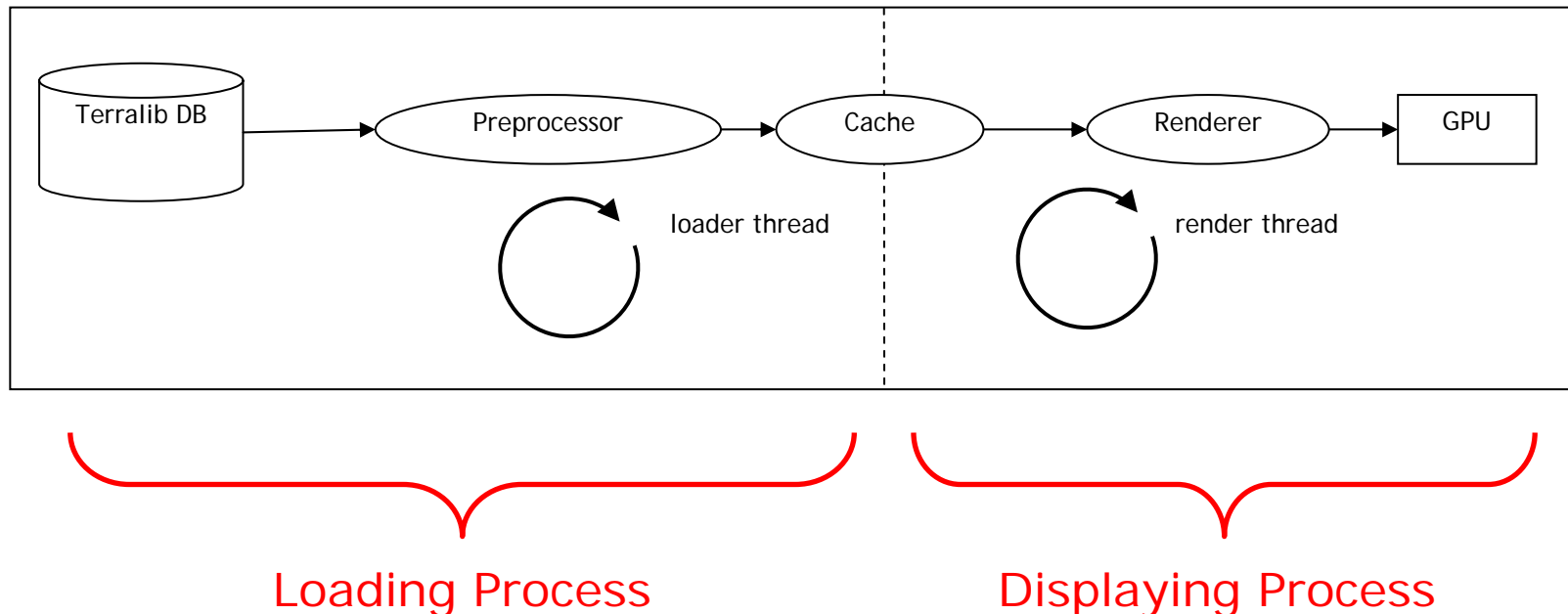


System Architecture



System Architecture

- The rendering process (fast) cannot wait for the loading process (slow)
- Solution: multithreading





- Caching requires a memory repository management
- Know what is in the repository and what is not
- Search in the repository
- Choose what to discard when all memory space is being used
- Do all these things fast!



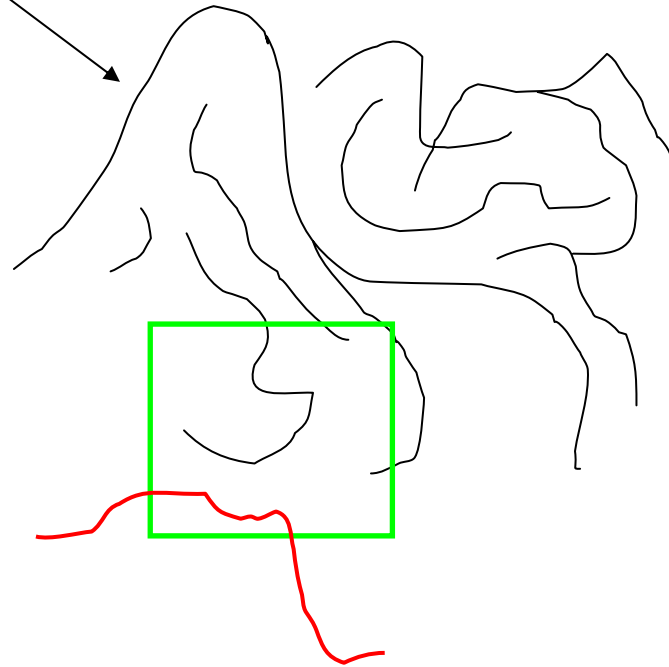
- **First option**: each cache entry keeps one geometry
- Very granular control
- Difficult memory management

- **Second option**: each cache entry keeps one geometry block
- Less granular control
- Memory management becomes more efficient

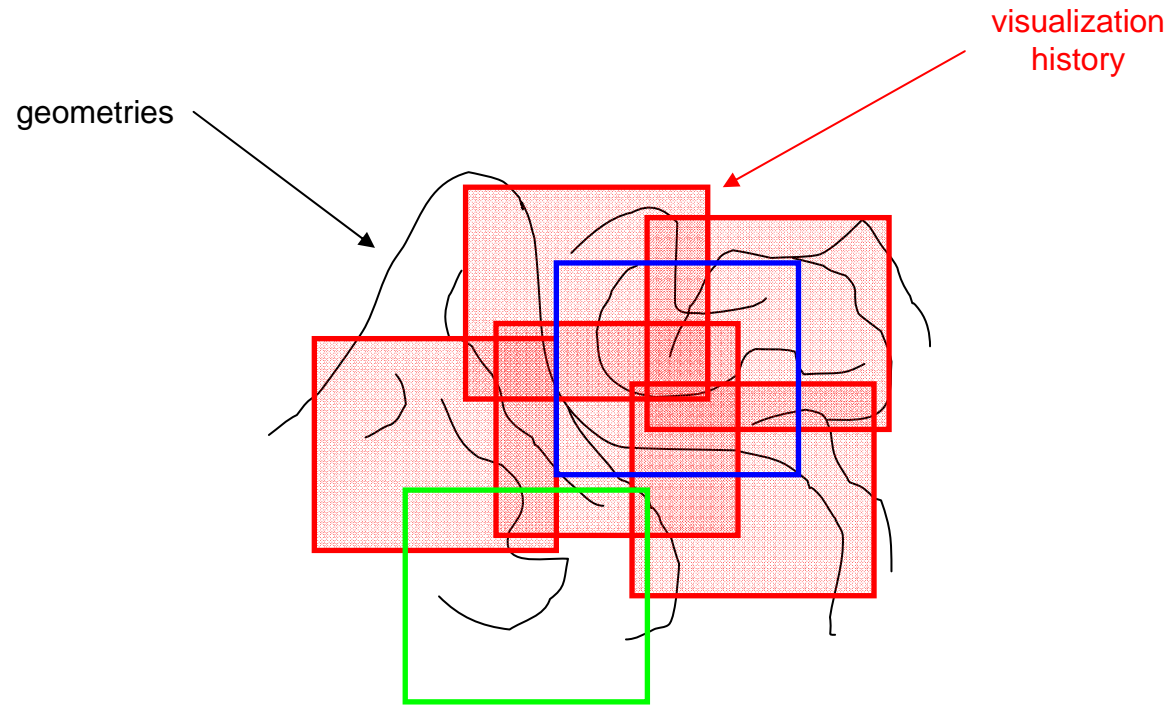
Memory Management



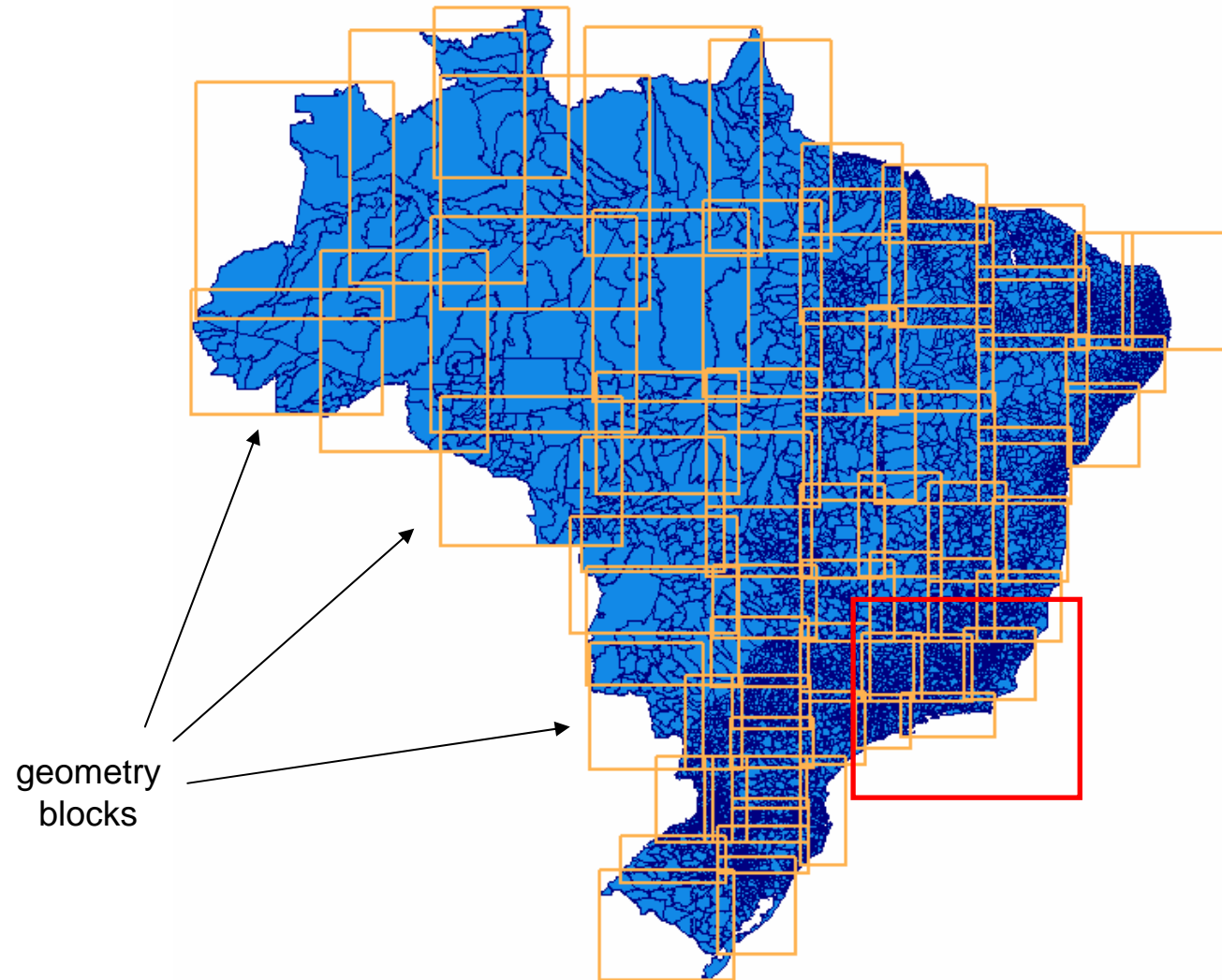
geometries



Memory Management



Memory Management



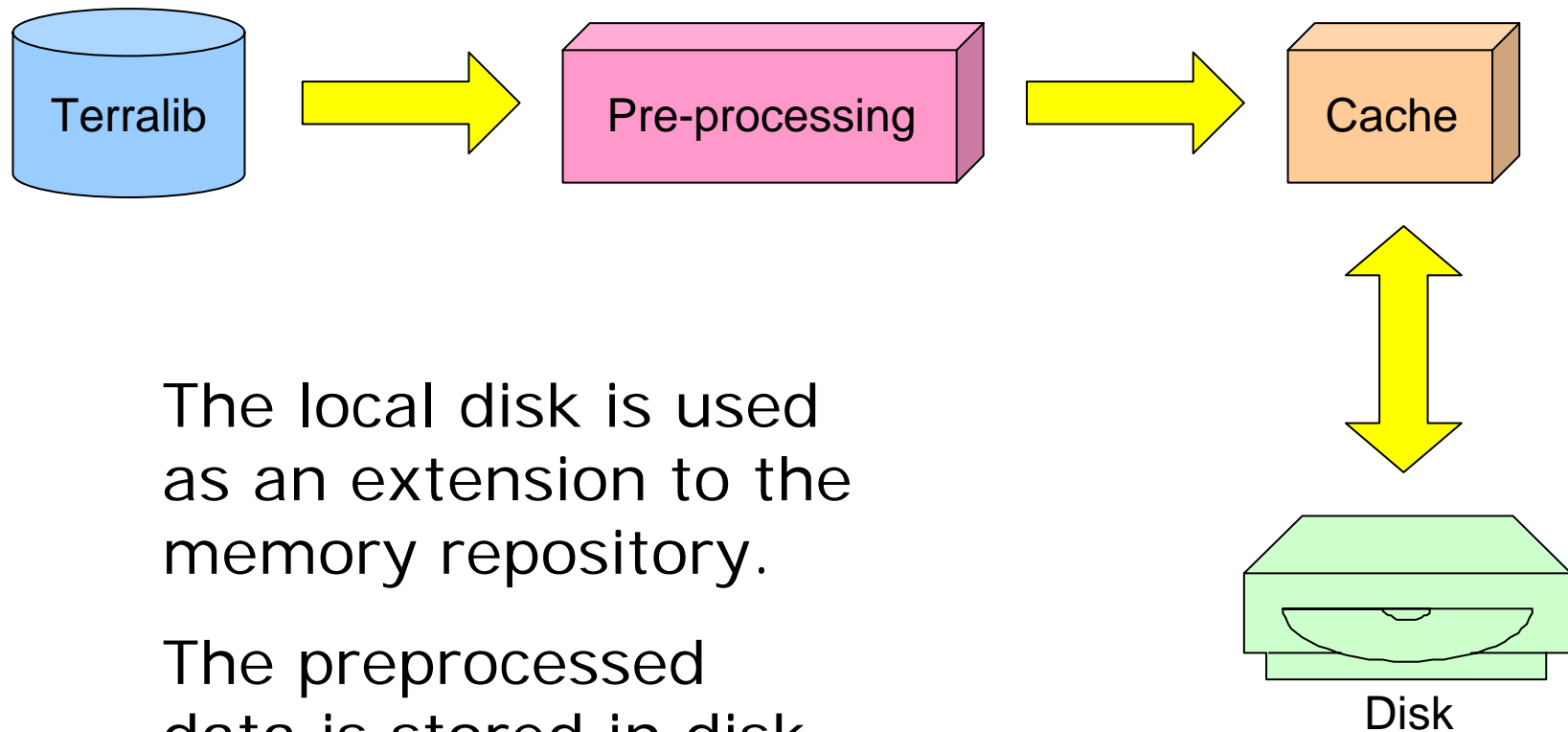


- Preprocessing
- Memory caching
- Loading and displaying processes done by different threads
- Memory management by blocks



- Disk cache
- Style grouping
- Complex style preprocessing

Disk cache



The local disk is used as an extension to the memory repository.

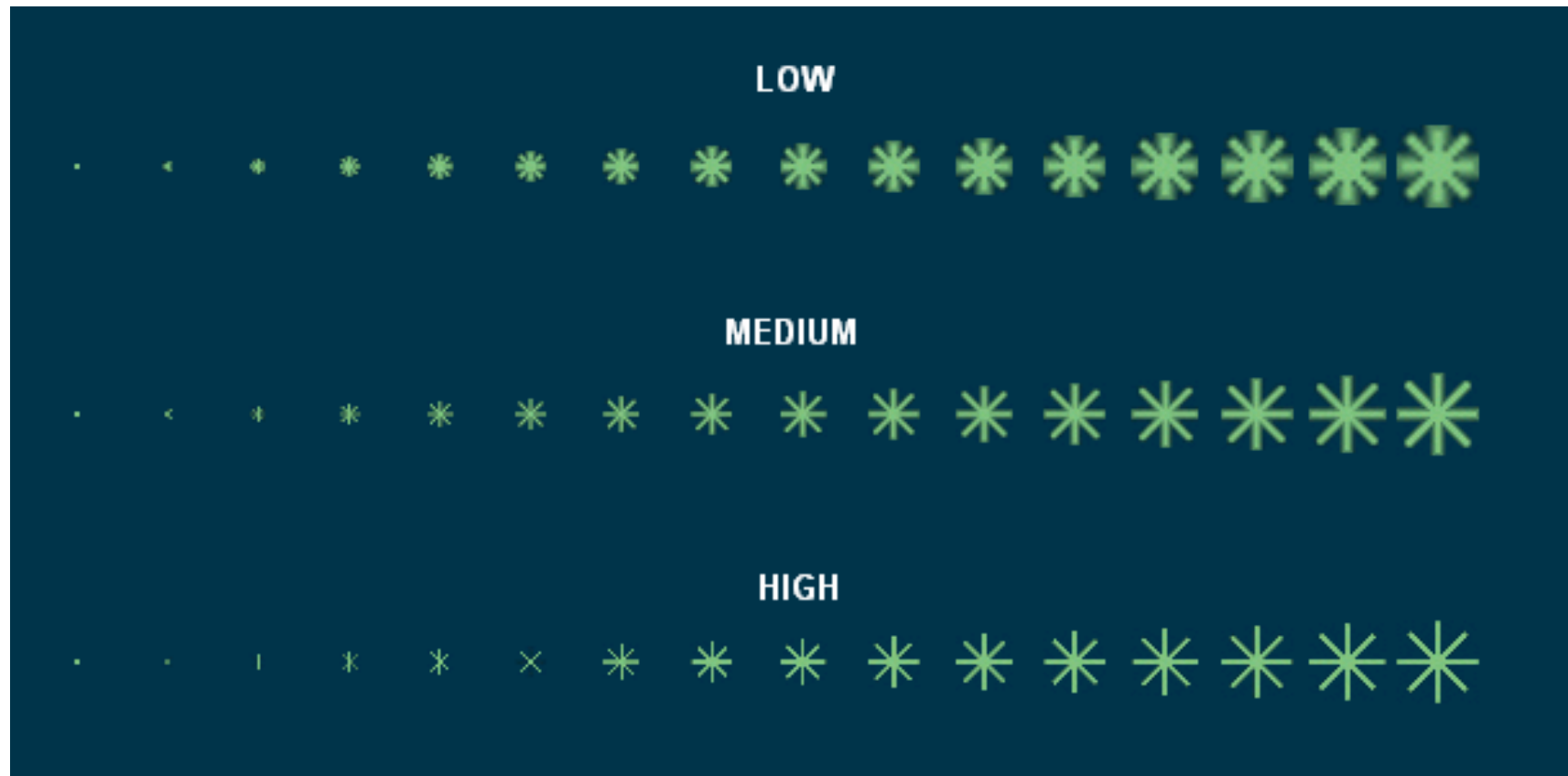
The preprocessed data is stored in disk.



- The more we can send geometries to the graphics card without changing its state, better the performance we get
- The state of the graphics card includes style information such as color, line width, filling patterns, etc
- The optimization consists in grouping together the geometries with the same style in the memory repository

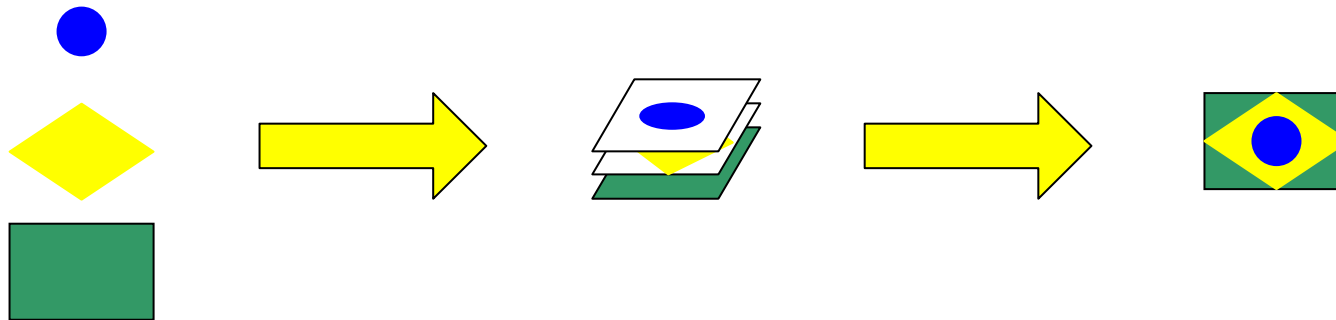
Complex style preprocessing

Using textures instead of drawing symbols
(may need multiple resolutions)





Merging multiple style layers into one single texture





- Success in building an continuous navigation application
- Outperformed Google Earth



- The code should be made available as open source in the near future as part of the TDK project
- TDK is an API developed by Tecgraf with the objective of helping with the implementation of applications on top of Terralib



- Development of a navigation and interaction model (e.g. geometry editing)
- Research on cache management heuristics (prefetch and discard policies, from disk and memory)
- Cache and fast rendering on WMS servers (sounds good!)
- Go one step further and use the GPU power for rendering 3D maps