DWD
Deutscher Wetterdienst

GEOINFORMATIONSDIENST DER BUNDESWEHR

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Swiss Confederation

Federal Department of Home Affairs FDHA
Federal Office of Meteorology and Climatology MeteoSwiss

MeteoSwiss

Danish Meteorological Institute

Environment Canada Environnement Canada
Meteorological Service of Canada
Service météorologique du Canada

EuMetSys
European Meteorological Systems
Ernst Basler + Partner GmbH

ACCSO
ACCELERATED SOLUTIONS

NinJo
Meteorological Workstation
Station de travail météorologique

Canvas-Grid

A new approach to NWP data visualization in NinJo

16th Workshop on Meteorological Operational Systems (MOS)

March, 2017

Waldemar Busiakiewicz, Oliver Eggert, Jan Schröter, Sören Kalesse

Outline

- Motivation for Canvas-Grid
- Idea
- Concepts
- Grib-lookup
- Sampling
- Calculation-FWK

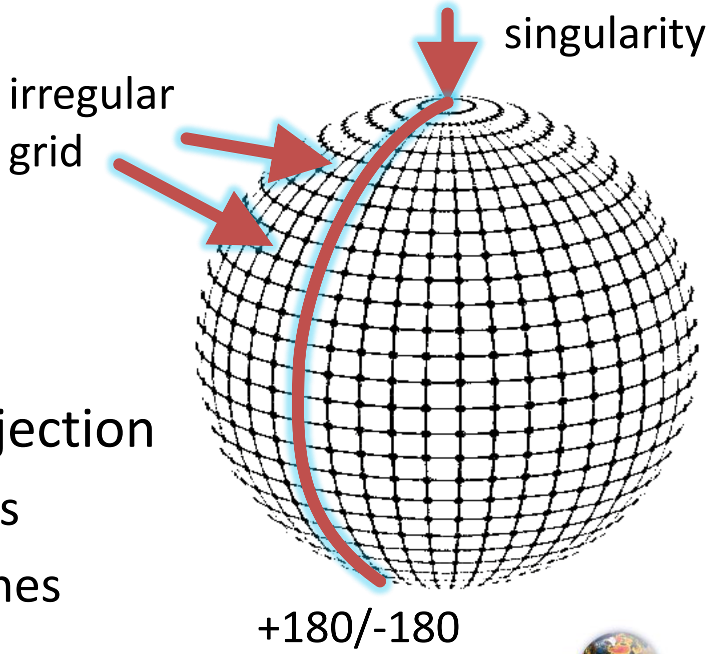
Motivation for Canvas-Grid

Visualization based directly on the globe is complicated

- Projection of GRIB-field differs from scene projection
  - How to check which part of the GRIB-field is currently visible?
  - The GRIB-field is irregular when projected onto the globe

- Special cases on dateline and poles
  - Wrap around dateline
  - Ambiguous pole representation

- Contour generation not on target projection
  - can cause self-intersections in polygons
  - can cause intersection of splined iso-lines

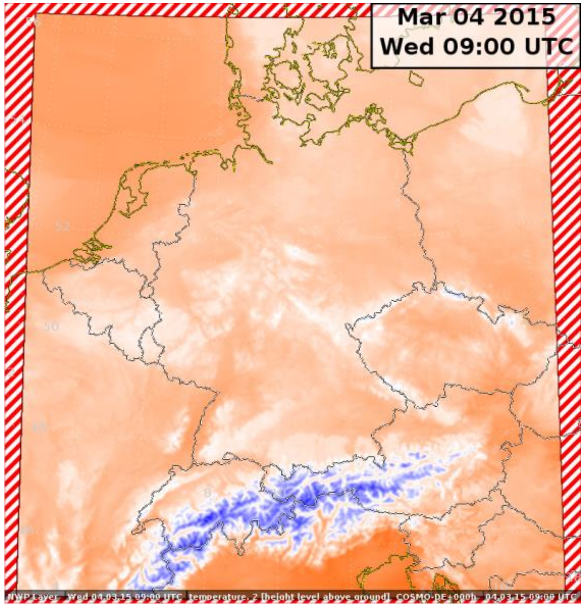


Motivation for Canvas-Grid

Work in display coordinate system is much simpler

- A simple 2D Cartesian coordinate system
- (almost) no special cases
- Simple iso-area and iso-line generation
- Simple accuracy estimation based on pixels

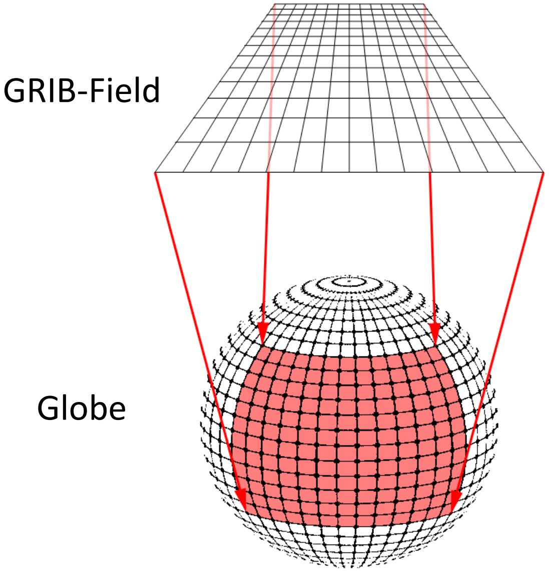
- Added benefit: new visualizations possible
  - e.g. per-pixel-coloring

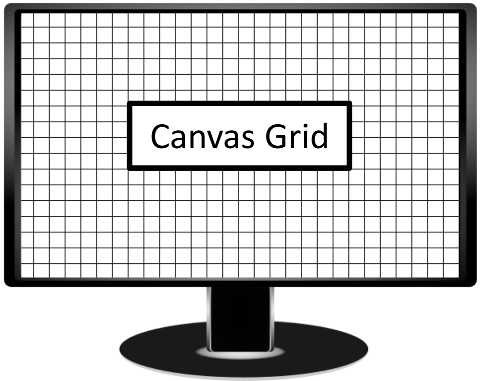


Pixel coloring example, combined with no-data visualization

Idea – Canvas-Grid

- Create an equidistant grid reflecting the display (“canvas-grid”)
- Lookup values from GRIB-field for each canvas-grid point
  - Canvas-grid points correspond to pixels
  - lower resolution is possible as well
    - Well suitable for iso-lines/-areas visualizations
- Use canvas-grid as a basis for visualization
  - Multiple visualizations on the same canvas-grid possible (lines, area, pixel, …)

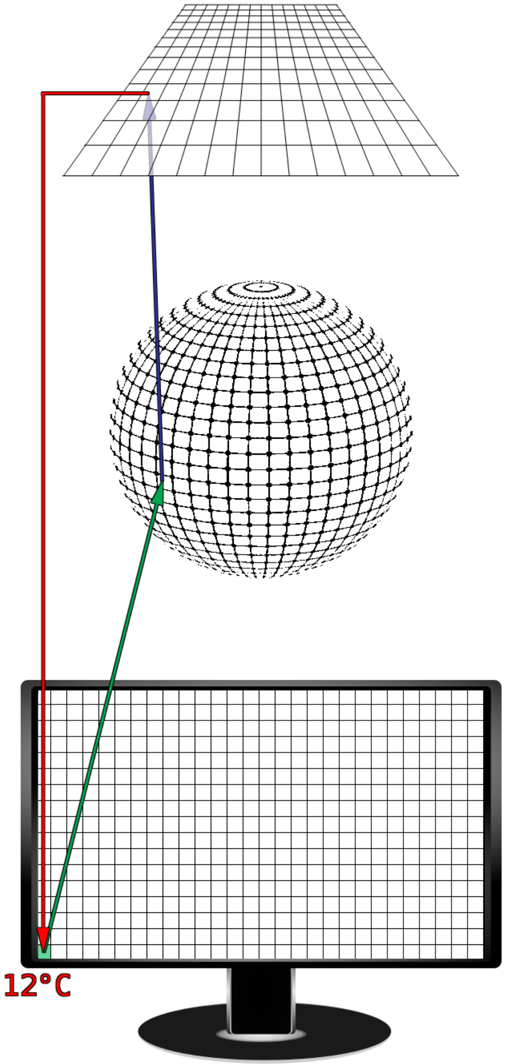




Concepts

Generation of the canvas-grid

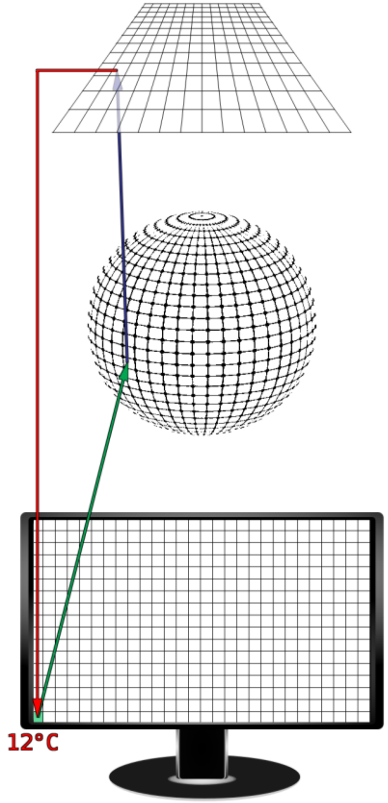
1. Compute long/lats for each canvas-pixel
   - Performed using target map projection
2. Compute GRIB-indices for each long/lat
   - Done by GRIB-containers projection
   - Result: GRIB-indices for each pixel
   - GRIB-indices are floating-points, as the long/lats are positioned in between GRIB-points
3. Lookup values from GRIB-field for each pixel
   - Use the precomputed GRIB-indices
   - **This is the new part!**



Detailed look into step 3: GRIB-lookup

Goal: for each canvas-pixel, lookup the GRIB-value

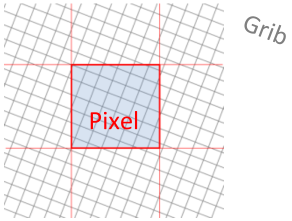
- **Input**:
  - GRIB-index (x,y) for a given pixel
    - The index is floating-point,
      i.e. points in between GRIB-points
- **Output**:
  - GRIB-value at (x,y)
    - Not necessarily the value at a given GRIB-point
    - Could be interpolated between GRIB-points
    - Could cover/span multiple GRIB-points



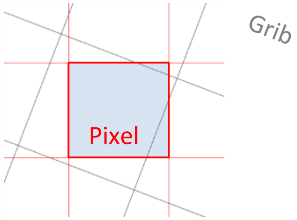
GRIB-lookup – Variants

Depends on canvas- vs. GRIB-resolution:

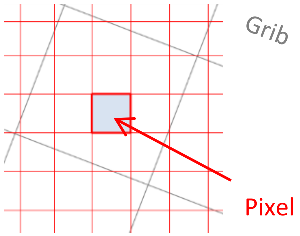
- canvas-resolution **<** GRIB-resolution
  - Aggregate all GRIB-points covered by a canvas-pixel



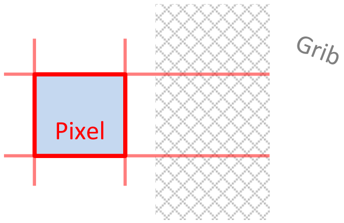
- canvas-resolution ≈ GRIB-resolution
  - Interpolate *bilinear* between GRIB-points



- canvas-resolution **>** GRIB-resolution
  - Interpolate *monotone-bicubic* between GRIB-points

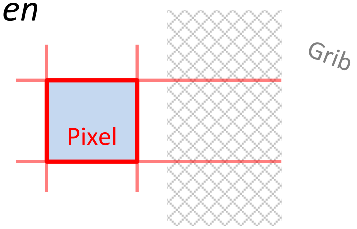


- Not enough surrounding GRIB-points
  - Nearest neighbor or NaN



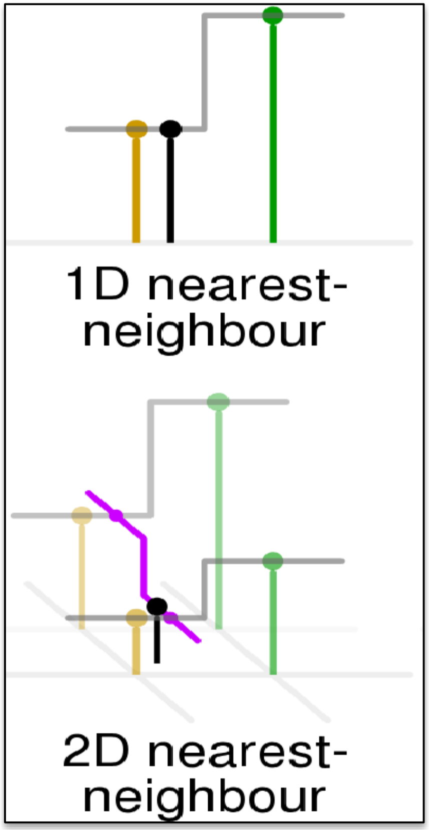
Note that, for each canvas pixel, a different option might have to be chosen

Used when

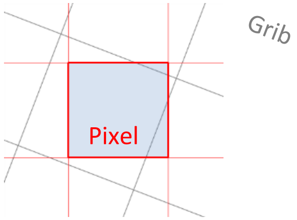


Nearest Neighbor

- The simplest form of interpolation
- Just use the nearest GRIB-point
- Used when not enough neighbors for applying other sampling methods



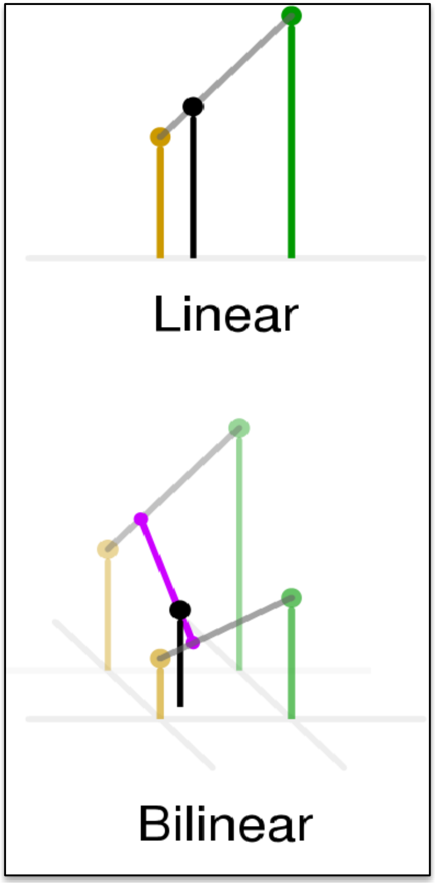
Used when

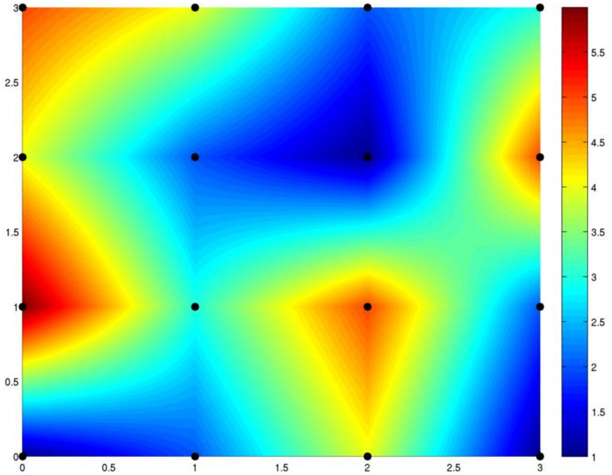


Bi-linear Interpolation

- Performs well when GRIB and display are of similar resolution

- Produces artifacts when both resolutions differ to much

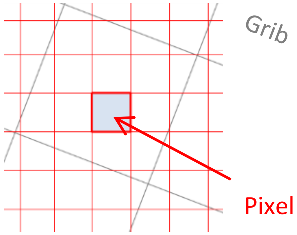




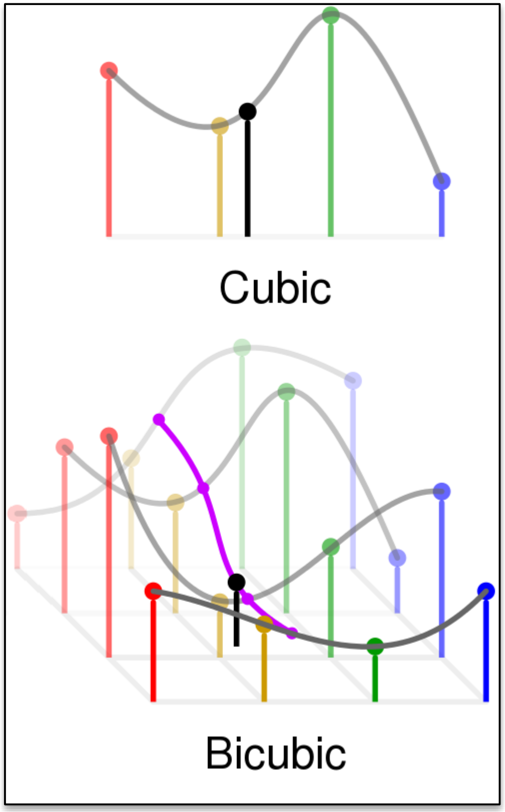
Result of bi-linear interpolation

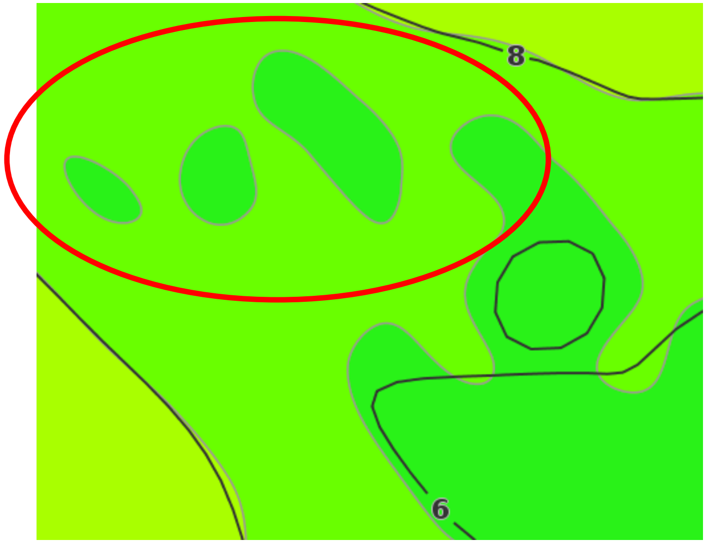
Monotone bi-cubic

Used when



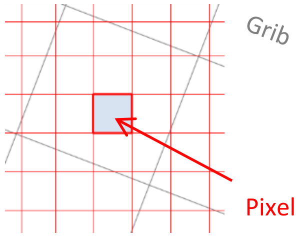
- Bi-cubic is much smoother than bilinear
- can produce artifacts
  → not used
- Use monotone bi-cubic instead





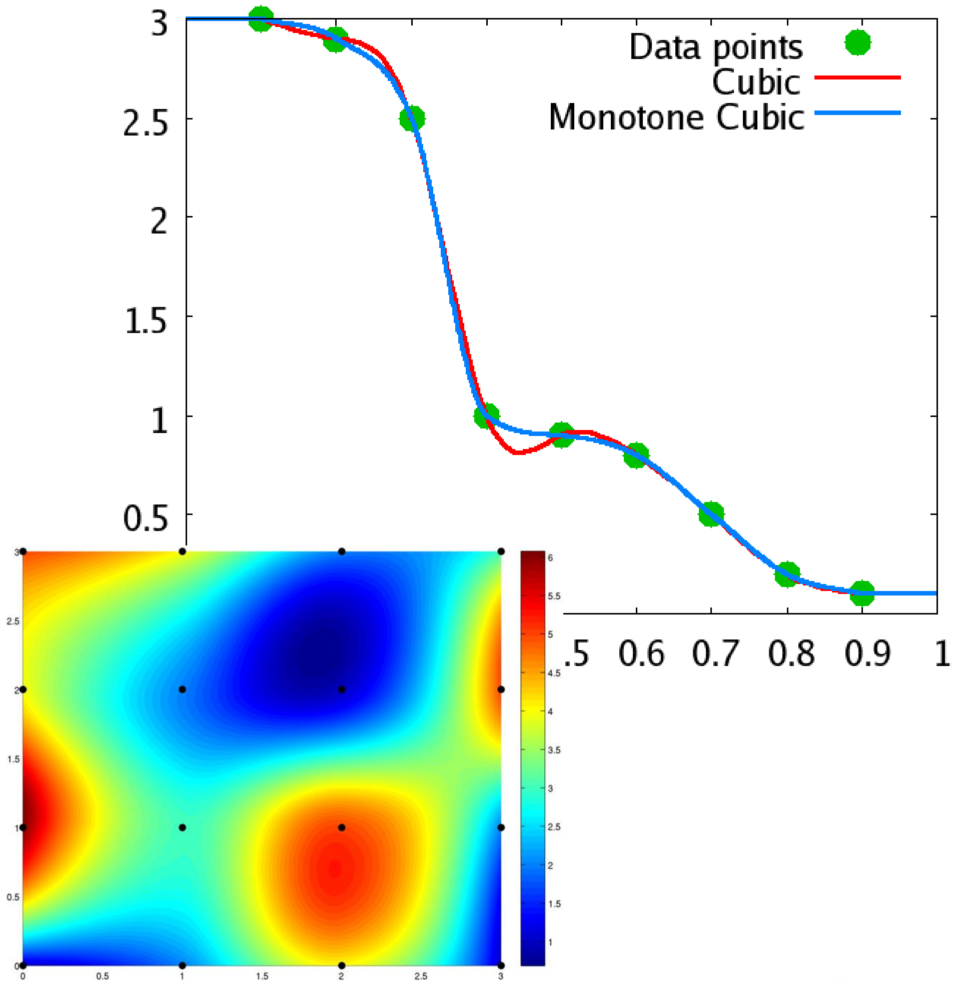
Overshoot artifacts of cubic interpolation

Used when



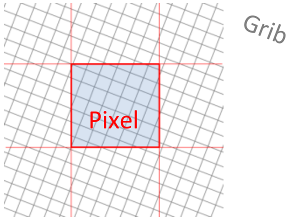
Monotone bi-cubic

- Same as bi-cubic
  - But uses harmonic-means instead of slopes/differentials
- Smooth transitions
  - But not as smooth as bicubic
- No overshoots
- Well suitable for high-resolution display



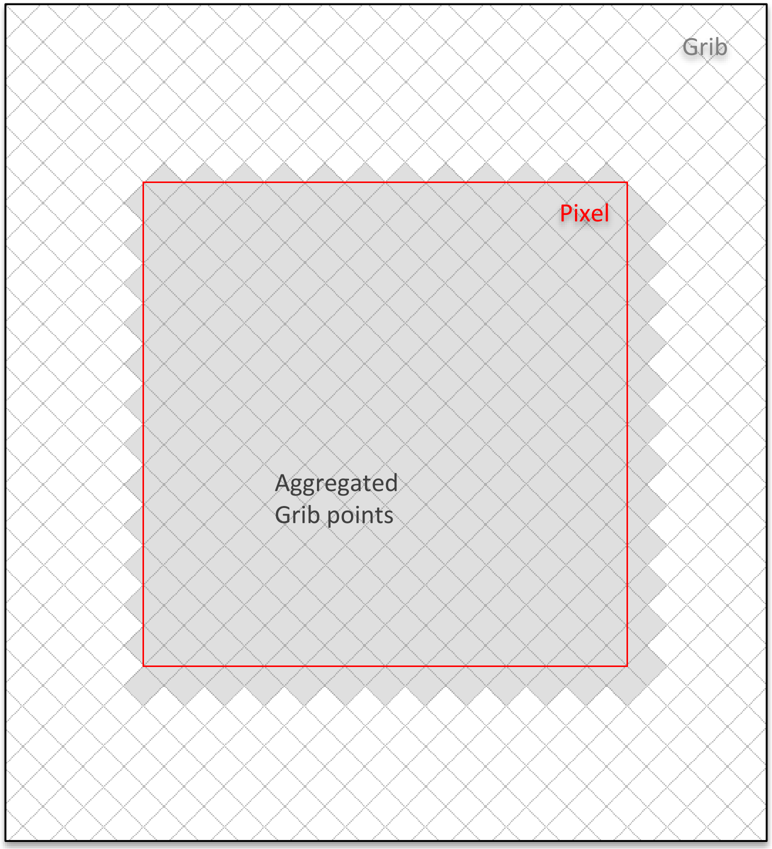
“Down-sampling”

Used when



Method of aggregation

- single canvas-pixel covers more than one GRIB-point
- Different methods for aggregating values
  - Minimum
  - Average
  - Maximum
- Implemented as a scanline algorithm



Use scan-line to aggregate covered GRIB-values

Canvas-Grid – Result

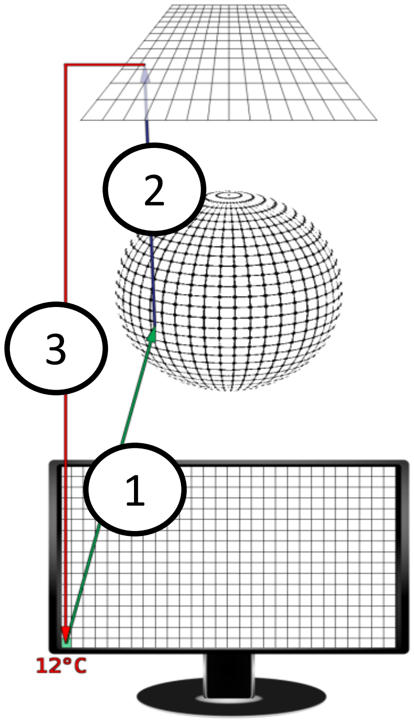
- The result of the canvas grid computation is called canvas-data
  - 2D array of float values in screen dimensions
- Canvas-data is input for Canvas-Grid visualizations
  - Not limited to GRIB-data
  - It is possible to transform other data (radar, sat, point, …) into Canvas-Data and thereby reuse visualization
- Downside: canvas-grid generation is expensive
  - *esp. lat/lon → grib-idx for unstructured gribs such as ICON*
- Performance optimizations required

Calculation-FWK – Introduction

- software framework for all sorts of computations in NinJo
  - allows nested computations
  - includes a global caching mechanism for results

- Computation of Canvas-data using Calculation Framework yields reusability and improved performance

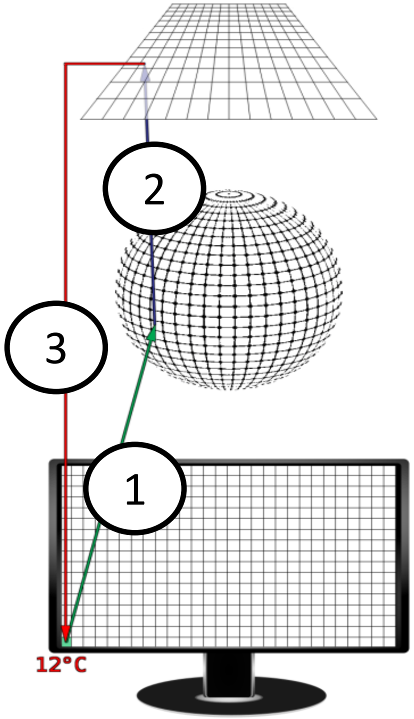
- Implemented as **three nested** calculations that mimic the “computation flow” (see slide 6):
  1. Compute lon/lats for each pixel
  2. Compute GRIB-indices for each lon/lat
  3. Lookup values from GRIB-field for each pixel



Canvas Calculation – Reusability

1. **Compute lat/lon for each pixel**
   - re-use as long as map projection is unchanged
     - all NinJo layers in one scene can share this data
2. **Compute GRIB-indices for each lat/lon**
   - re-use for all visualizations of the same model i.e. model geometry unchanged
3. **Lookup values from GRIB-field for each pixel**
   - not very re-usable
   - can only be shared for different visualizations of the same data

- **best case:** "simply" stepping through time
- **worst-case:** zoom/pan



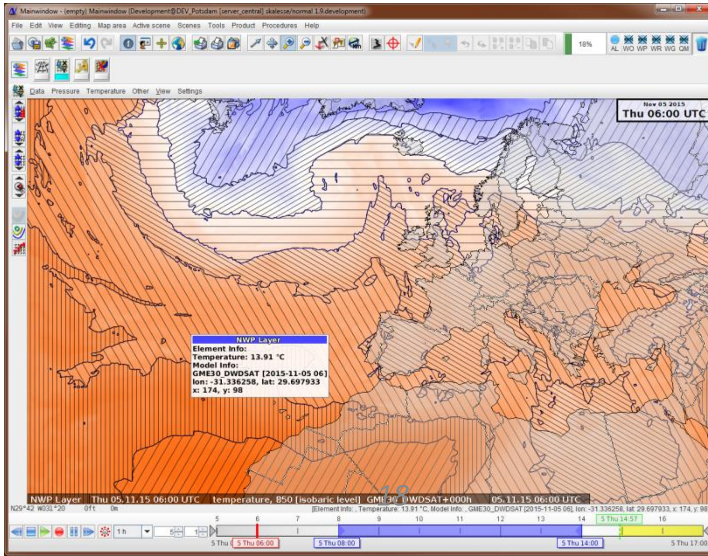
Canvas Calculation – performance

```
Convert    1770954 coords from scene to long/lat took:       137ms
Convert    1770954 coords from long/lat to grid took:         22ms
Lookup of  1770954 points from grid-field took:               92ms
    Nearest-neighbor: looked up     2824 pixels
    Downsample:       looked up        0 pixels
    Bilinear:         looked up  1768130 pixels
    Monotone-bicubic: looked up        0 pixels
```

- Numbers taken during development, they are not “final”
- First two lines (137ms + 22ms) can be saved when stepping through time
- accounts for approx. 60% of canvas grid calculation time

Canvas-Grid – Summary

- new met. data visualization for NinJo
  - based on a virtual Cartesian grid in the screen coordinate system
  - applicable for all sorts of data that can be transformed onto that virtual grid (Radar, SAT, …)
- idea:
  - estimate/look up values for each pixel of the canvas-grid
  - then visualize data on canvas-grid rather than GRIB or lat/lon coordinate system.
- iso-line/iso-area generation straight forward
- depending on the data projection, transformation might be expensive but can be optimized


Mainwindow - (empty) Mainwindow [Development@DEV_Potsdam [server_central] skalesse/normal 1.9.development]
File Edit View Editing Map area Active scene Scenes Tools Product Procedures Help
18%
AL WO WP WR WG QM
Data Pressure Temperature Other View Settings
Nov 05 2015
Thu 06:00 UTC
N29°42 W031°20 0ft 0m
1 h
5 6 7 8 9 10 11 12 13 14 5 Thu 14:57 16
5 Thu 5 Thu 06:00 5 Thu 08:00 5 Thu 14:00 5 Thu 17:00

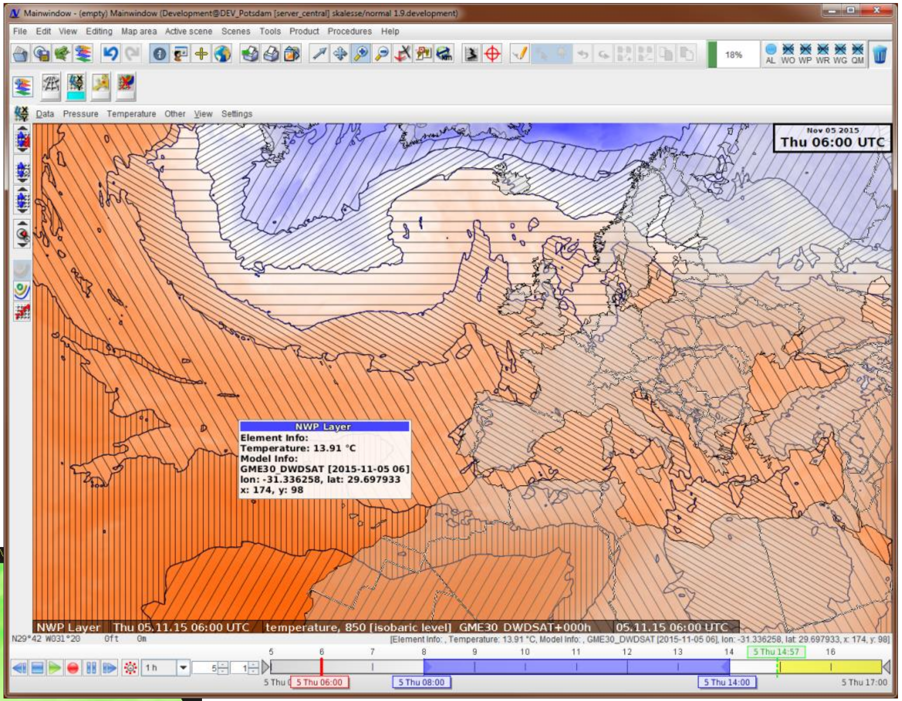
NWP Layer
Element Info:
Temperature: 13.91 °C
Model Info:
GME30_DWDSAT [2015-11-05 06]
lon: -31.336258, lat: 29.697933
x: 174, y: 98

NWP Layer Thu 05.11.15 06:00 UTC temperature, 850 [isobaric level] GME30_DWDSAT+000h 05.11.15 06:00 UTC

[Element Info: , Temperature: 13.91 °C, Model Info: , GME30_DWDSAT [2015-11-05 06], lon: -31.336258, lat: 29.697933, x: 174, y: 98

Canvas-Grid visualization

- **Thank You for the attention!**
- Credits
  - Waldemar Busiakiewicz
  - Oliver Eggert
  - Jan Schröter



Mainwindow - (empty) Mainwindow (Development@DEV_Potsdam [server_central] skalesse/normal 1.9.development)

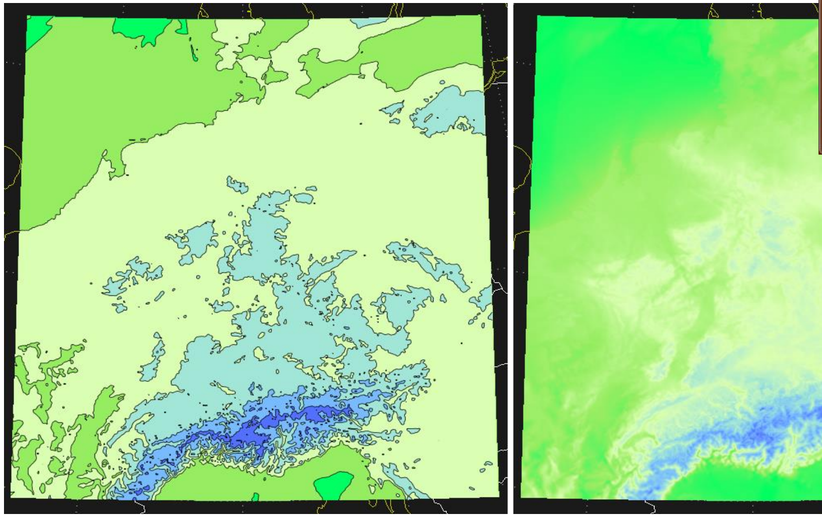
NWP Layer
Element Info:
Temperature: 13.91 °C
Model Info:
GME30_DWDSAT [2015-11-05 06]
lon: -31.336258, lat: 29.697933
x: 174, y: 98

NWP Layer Thu 05.11.15 06:00 UTC temperature, 850 [isobaric level] GME30 DWDSAT+000h 05.11.15 06:00 UTC

N29°42 W031°20 0ft 0m

[Element Info: , Temperature: 13.91 °C, Model Info: , GME30_DWDSAT [2015-11-05 06], lon: -31.336258, lat: 29.697933, x: 174, y: 98]

Pixel-coloring and hatch-filled iso-areas



Two types of visualizations on the canvas-grid for a German local model