**Examples for using GISAlgebra**

This document describes how to work with GISAlgebra in Secondo.

GISAlgebra is used for terrain analysis, like slope or contour calculation.

# Operator reference

The GISAlgebra consists of five operators’ aspect, contourlines, hillshade, ruggedness and slope. All operators are written for Raster2- and TileAlgebra data.

## aspect

The aspect operator calculates the direction in which a hillside is shown. As parameter the z-factor (conversion factor between different scale units) must be assigned.

**syntax:**

\_ aspect [\_]

**signatures:**

sT x double -> sT for T in {int, real}

stream(tT) x double -> stream(tT) for T in {int, real}

**examples:**

query [const sint value ((1.0 1.0 1.0)(3 3)(0 0 (90 91 92 114 115 116 132 133 134)))] aspect[1.0]

query treal feed aspect[1.0] consume

## contourlines

The contourlines operator calculates the contour for a given interval.

**syntax:**

\_ contourlines [\_]

**signatures:**

sT x integer -> stream(lines) for T in {int, real}

stream(tT) x integer -> stream(lines) for T in {int, real}

**examples:**

query [const sint value ((1.0 1.0 1.0)(3 3)(0 0 (90 91 92 114 115 116 132 133 134)))] contourlines[2] consume

query treal feed contourlines[2] consume

## hillshade

The hillshade operator calculates a hypothetical illumination of a surface. As parameters the azimuth and the altitude of the illumination source and the z-factor (conversion factor between different scale units) must be assigned.

**syntax:**

\_ hillshade [\_,\_,\_]

**signatures:**

sT x double x double x double -> sT for T in {int, real}

stream(tT) x double x double x double -> stream(tT) for T in {int, real}

**examples:**

query [const sint value ((1.0 1.0 1.0)(3 3)(0 0 (90 91 92 114 115 116 132 133 134)))] hillshade[1.0, 100.0, 20.0]

query treal feed hillshade[1.0, 100.0, 20.0] consume

## ruggedness

The ruggedness operator calculates the amount of elevation difference between adjacent cells.

**syntax:**

\_ ruggedness

**signatures:**

sT -> sT for T in {int, real}

stream(tT) -> stream(tT) for T in {int, real}

**examples:**

query [const sint value ((1.0 1.0 1.0)(3 3)(0 0 (90 91 92 114 115 116 132 133 134)))] ruggedness

query treal feed ruggedness consume

## slope

The slope operator calculates the maximum rate of change from a cell to its neighbors. As parameter the z-factor (conversion factor between different scale units) must be assigned.

**syntax:**

\_ slope [\_]

**signatures:**

sT x double -> sT for T in {int, real}

stream(tT) x double -> stream(tT) for T in {int, real}

**examples:**

query [const sint value ((1.0 1.0 1.0)(3 3)(0 0 (90 91 92 114 115 116 132 133 134)))] slope[1.0]

query treal feed slope[1.0] consume

Example for slope usage

In this chapter the route for a bicycle trip will be planned.

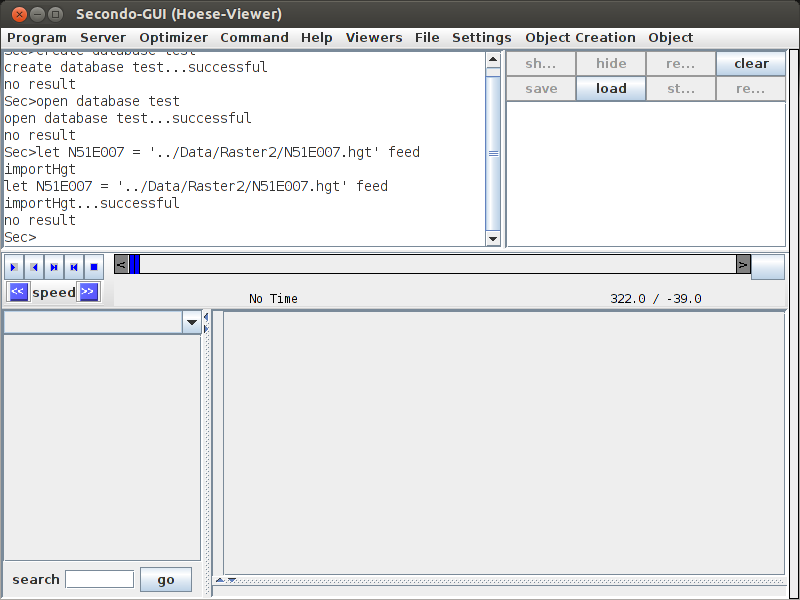
First Secondo must be started; a database must be created and opened

create database test

open database test

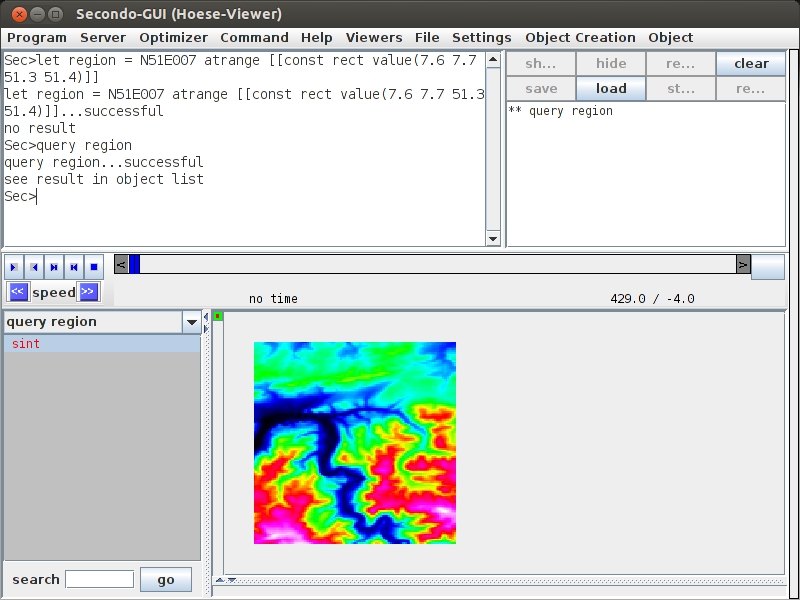
Next step is to import raster data in hgt-format. For example import file N51E007.hgt

let N51E007 = '../Data/Raster2/N51E007.hgt' feed importHgt



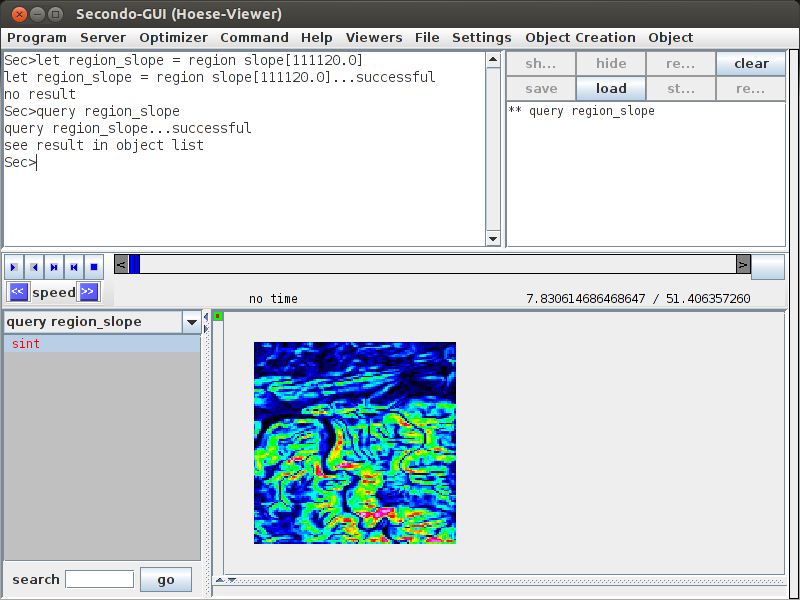
The loaded data must be cut to the area of interest

let region = N51E007 atrange [[const rect value(7.6 7.7 51.3 51.4)]]



The slope must be calculated for the area of interest. Because x- and y-coordinates are given in degree and the height is given in meter the z-factor must be set to 111200.0.

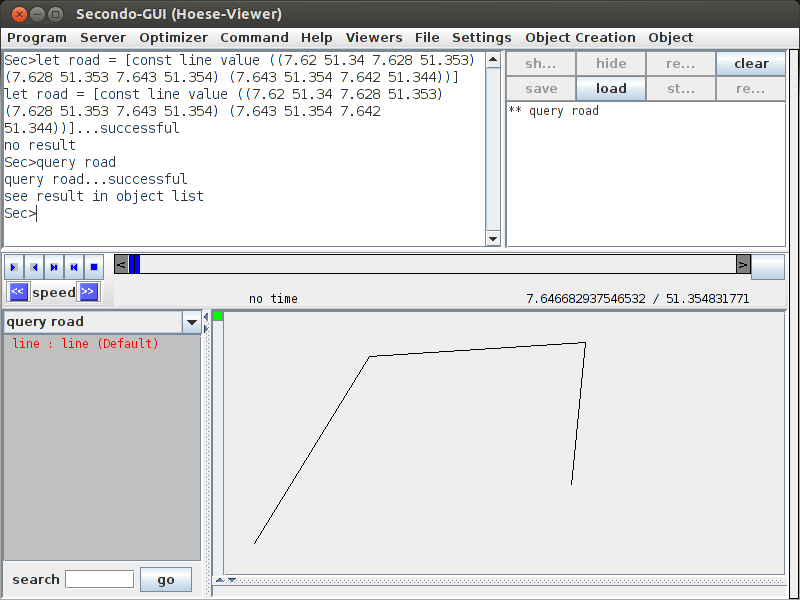
let region\_slope = region slope[111120.0]



All cells with yellow or red color have a high ascending slope, all cells with blue or green color are nearly flat.

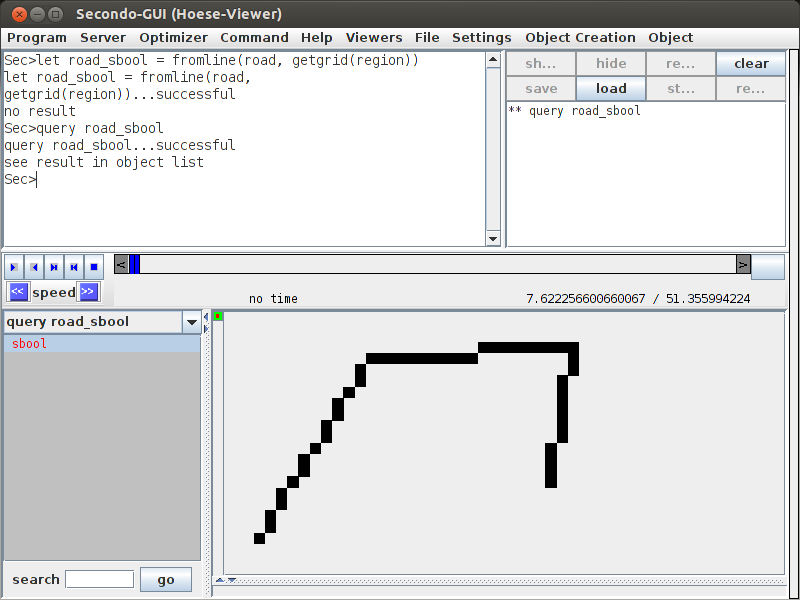
To calculate the slope for a specific road a line must be created

let road = [const line value ((7.62 51.34 7.628 51.353) (7.628 51.353 7.643 51.354) (7.643 51.354 7.642 51.344))]



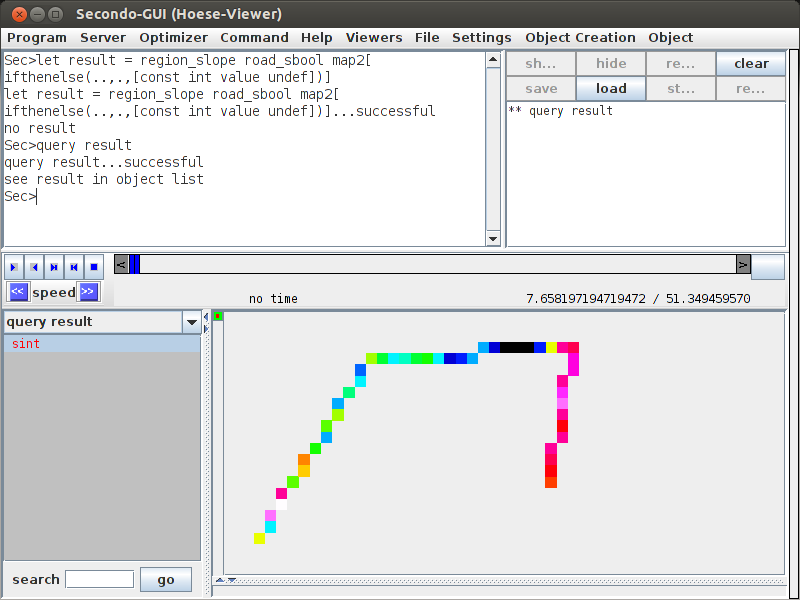
This line must be converted to a boolean Raster2-object with the same coordinates and cell size as the area of interest

let road\_sbool = fromline(road, getgrid(zuschnitt))



As final step the road and the calculated slope for the area of interest must be combined. The result is a line with colored cells.

let result = region\_slope road\_sbool map2[ ifthenelse(..,.,[const int value undef])]



The example is also possible for TileAlgebra data. In that case the commands look slightly different and the raster data must first be converted to tile data.

let N51E007Tiles = tiles(N51E007) namedtransformstream[No] consume

let region\_slope = region feed slope[111120.0] consume

Example for contourlines usage

In this chapter it is shown how a topographical map can be created.

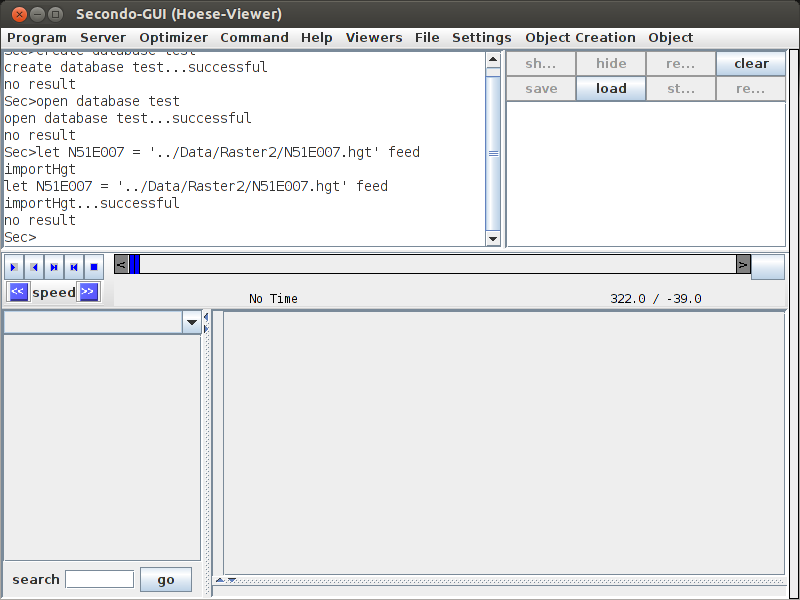
First Secondo must be started; a database must be created and opened

create database test

open database test

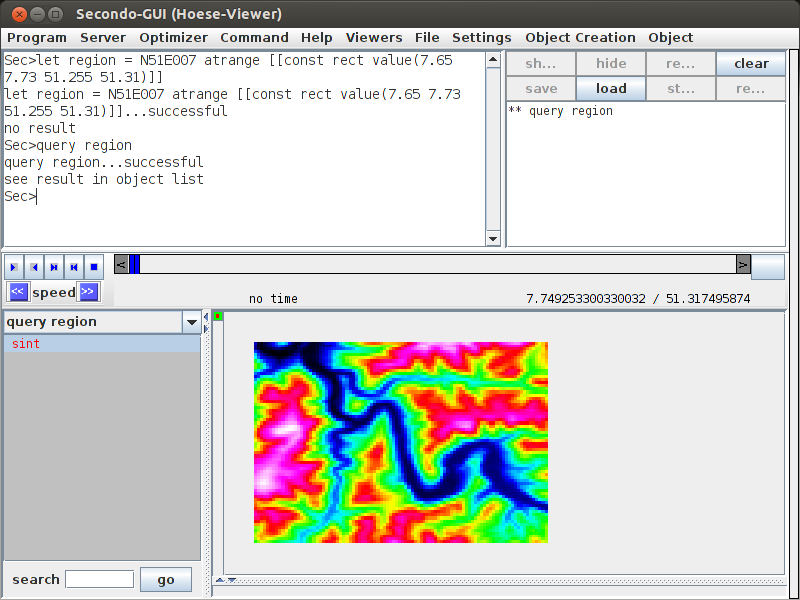
Next step is to import raster data in hgt-format. For example import file N51E007.hgt

let N51E007 = '../Data/Raster2/N51E007.hgt' feed importHgt



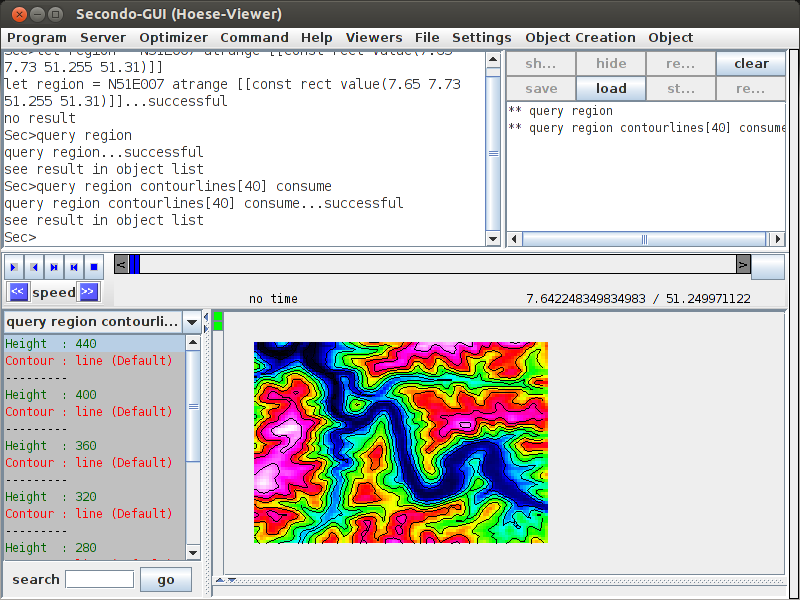
The loaded data must be cut to the area of interest

let region = N51E007 atrange [[const rect value(7.65 7.73 51.255 51.31)]]



The contour lines should be displayed every 40 meter

query region contourlines[40] consume



This example is also possible for TileAlgebra data. In that case the commands look slightly different and the raster data must first be converted to tile data.

let N51E007Tiles = tiles(N51E007) namedtransformstream[No] consume

query region feed contourlines[40] consume