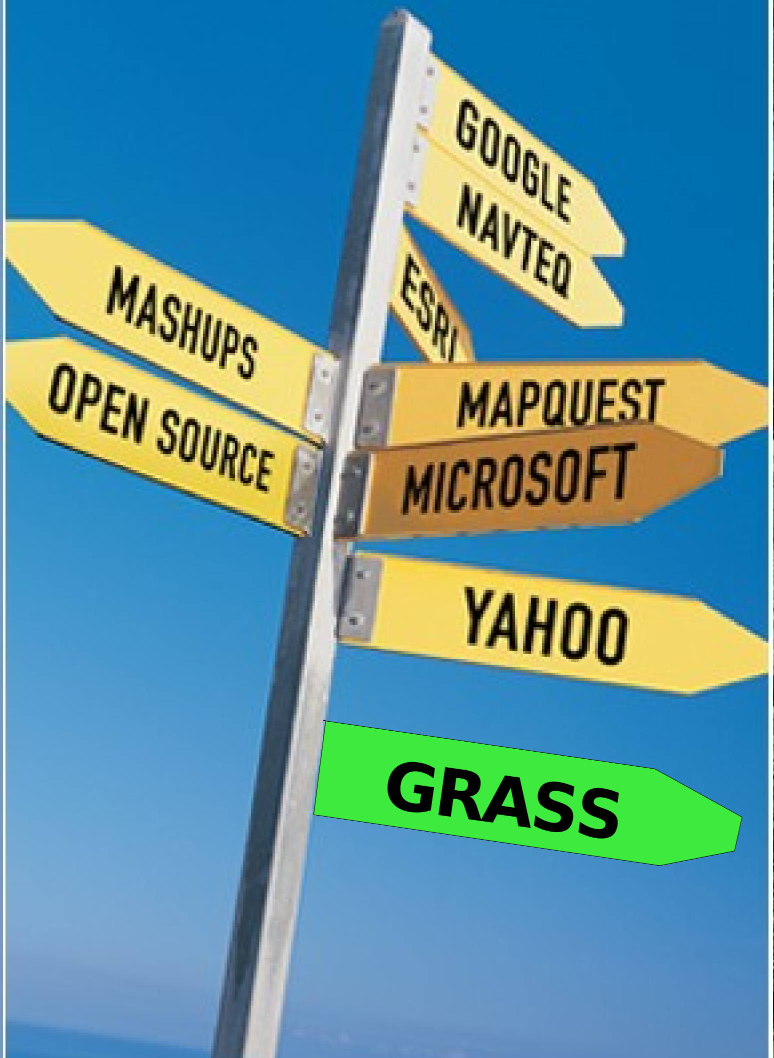
O'REILLY

where2.0™ CONFERENCE

The Future of Mapping and Local Search



Wilderness Navigational Planning Using GRASS GIS Analysis and Public Geographic Data

Dylan E. Beaudette
University of California at Davis
Dept. Land, Air, Water Resources

Talk Objectives

Background on GRASS and GIS data types

- Historical significance, and open source development
- Supported data types with examples

Outline of GRASS component model and data storage

- Metaphors used to define GRASS data sets

Sample application of analytical tools

- Use of freely available DRG and DEM products
- Route planning in a wilderness area
- 2D exploration of data with QGIS
- 3D exploration with NVIZ
- Press-quality output with Generic Mapping Tools

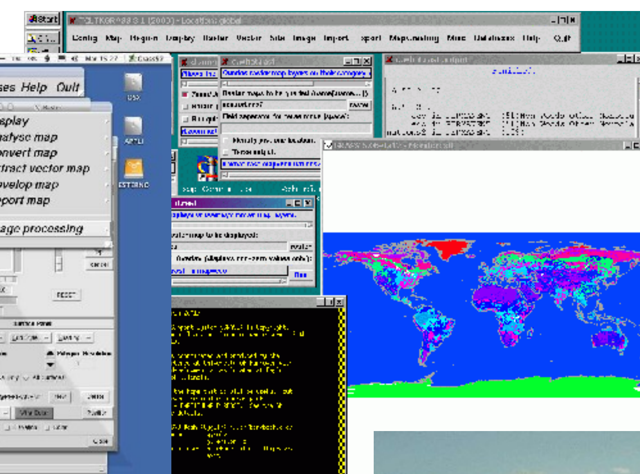
GRASS GIS

Brief Introduction

- Developed since 1984, **always Open Source**, since 1999 under GNU GPL
- Written in C programming language, **portable code** (multi-OS, 32/64bit)
- **International development team**, since 2001 coordinated at ITC-irst
- GRASS master Web site:

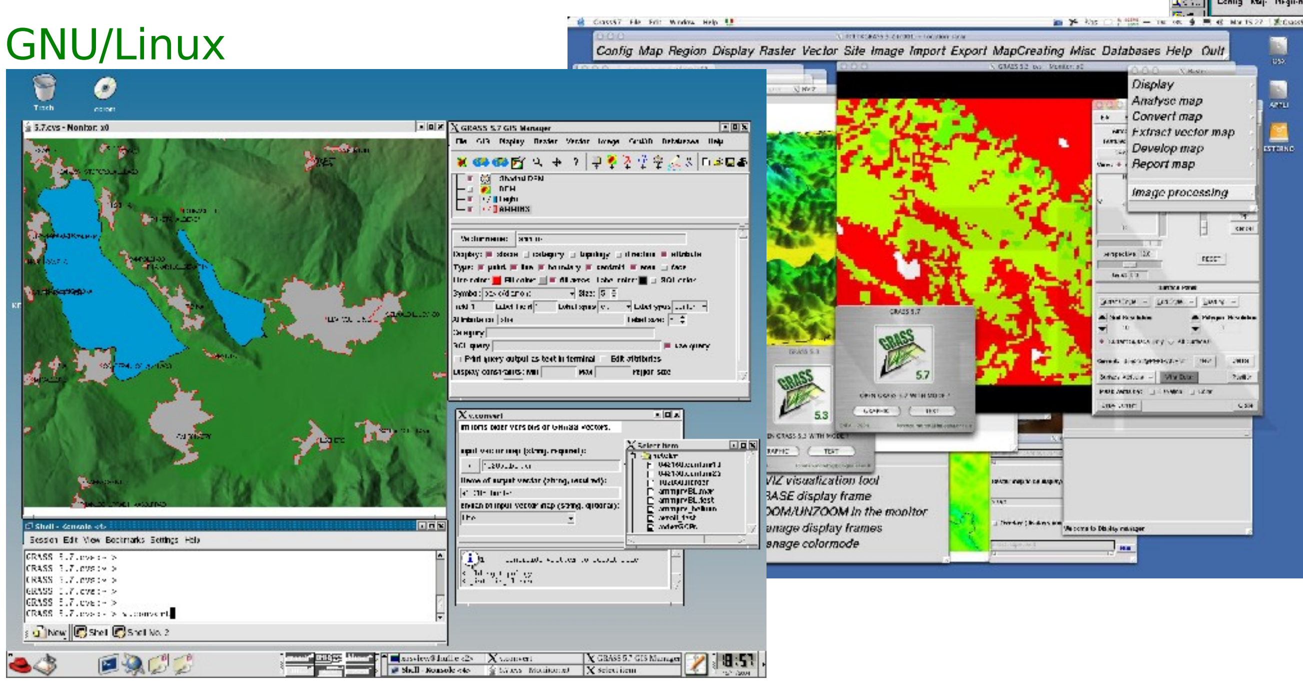
http://grass.itc.it

MS-Windows



MacOSX

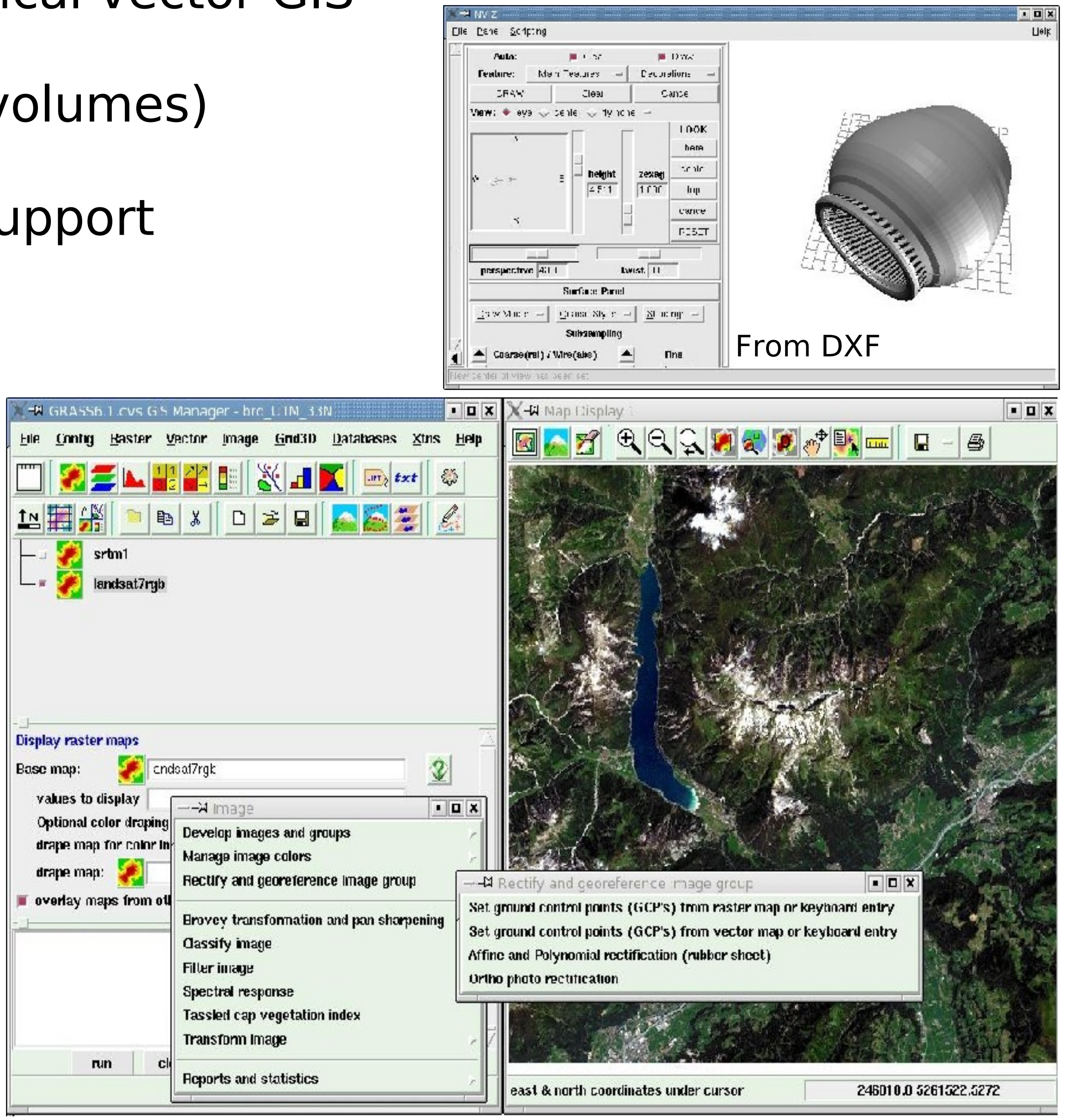
GNU/Linux



iPAQ

What's GRASS GIS?

- Raster and 2D/3D topological vector GIS
- Voxel support (raster 3D volumes)
- Vector network analysis support
- Image processing system
- Visualization system
- DBMS integrated (SQL) with dbf, PostgreSQL, MySQL and sqlite drivers
- In GRASS 6.1 translations of the user interface to 16 languages ongoing
- Interoperability: supports all relevant raster and vector formats



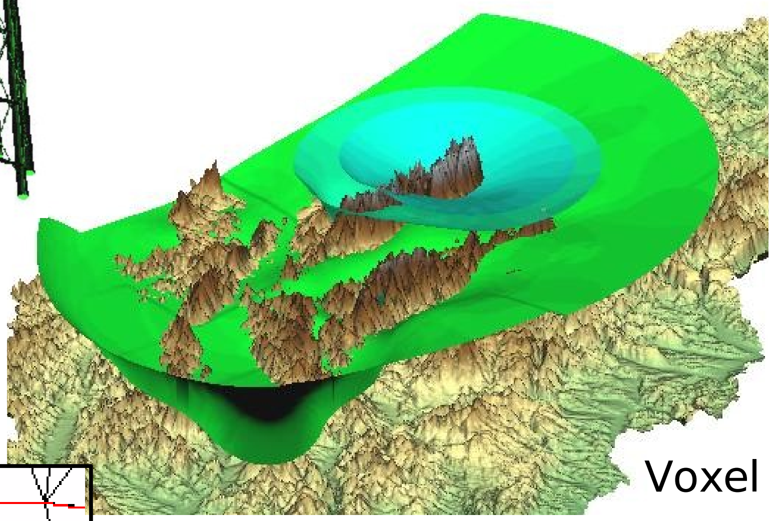
Spatial Data Types

Supported Spatial Data Types

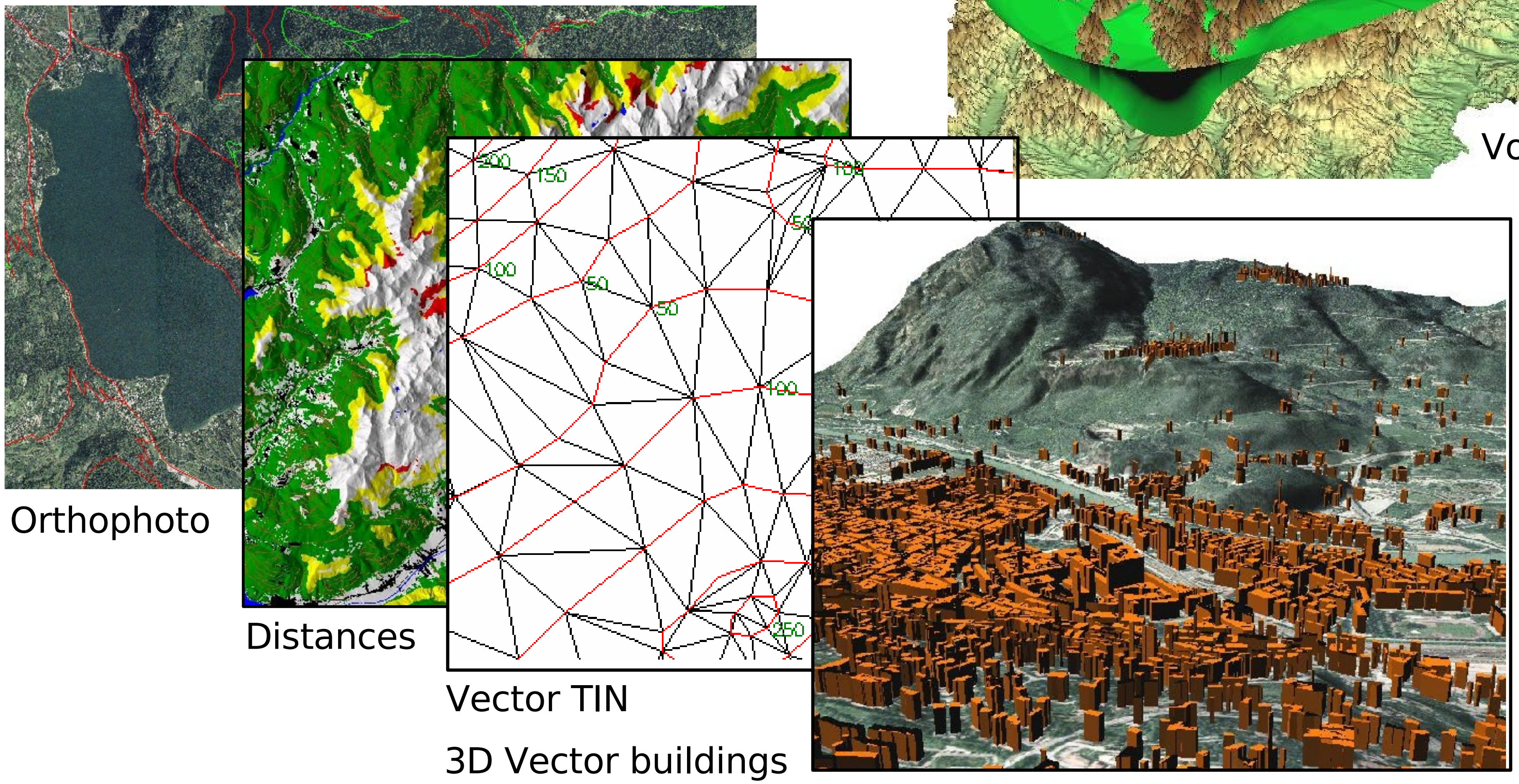
- 2D Raster data incl. image processing
- 3D Voxel data for volumetric data

- 2D/3D Vector data with topology
- Multidimensional points data

http://grass.itc.it



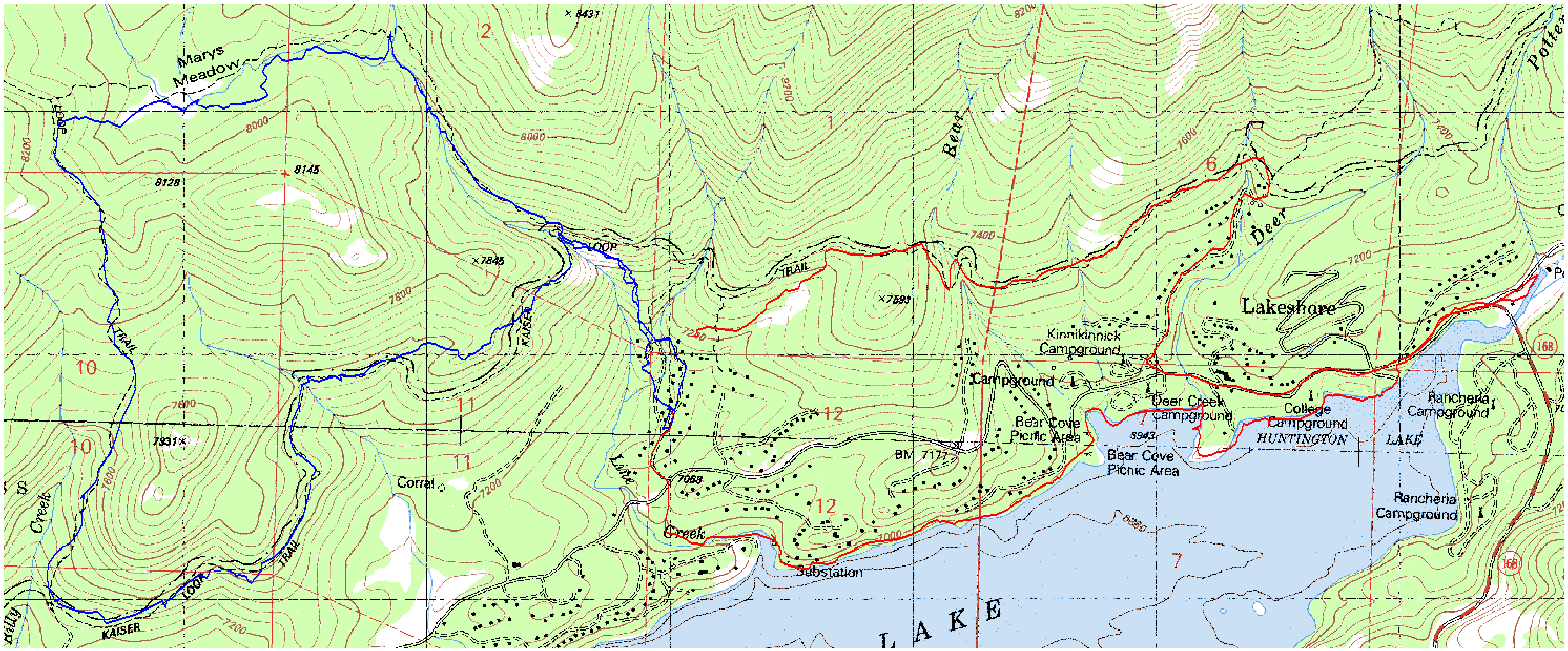
Voxel



Orthophoto
Distances
Vector TIN
3D Vector buildings

Sample Maps

USGS Topographic Map with GPS track data



See example paper maps…

Raster data model

Raster geometry

- cell matrix with coordinates
- resolution: cell width / height   (can be in kilometers, meters, degree etc.)

1.0	2.0	2.0	3.0	3.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0
0.0	1.0	1.0	2.0	2.0	3.0	3.0	4.0	4.0	4.0	5.0	5.0
0.0	0.0	0.0	1.0	1.0	2.0	2.0	3.0	3.0	3.0	4.0	4.0
1.0	0.0	0.0	0.0	0.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0
2.0	1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	2.0	2.0
3.0	2.0	2.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0
4.0	3.0	3.0	2.0	2.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0
5.0	4.0	4.0	3.0	3.0	2.0	2.0	1.0	1.0	1.0	0.0	0.0
6.0	5.0	5.0	4.0	4.0	3.0	3.0	2.0	2.0	2.0	1.0	1.0

y resolution

x resolution

Vector data model

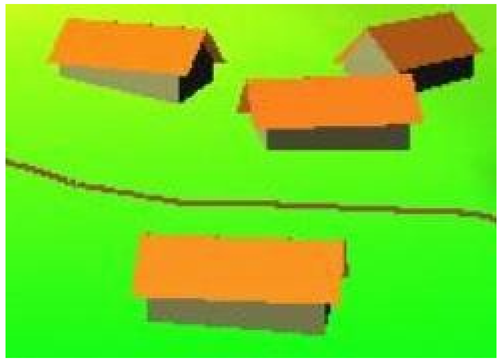
Vector geometry types

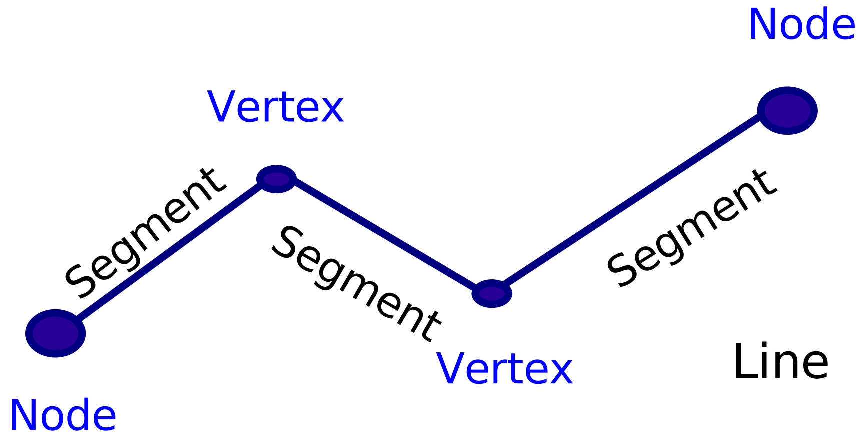
- Point
- Centroid
- Line
- Boundary
- Area (boundary + centroid)
- face (3D area)
- [kernel (3D centroid)]
- [volumes (faces + kernel)]

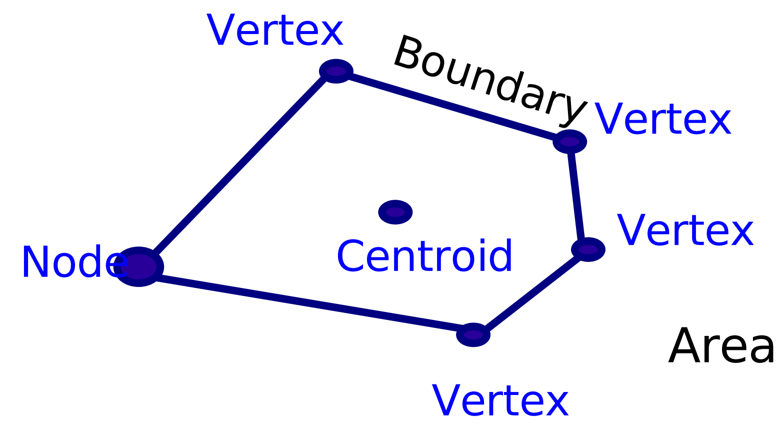
not in all GIS!

Geometry is **true** 3D: x, y, z

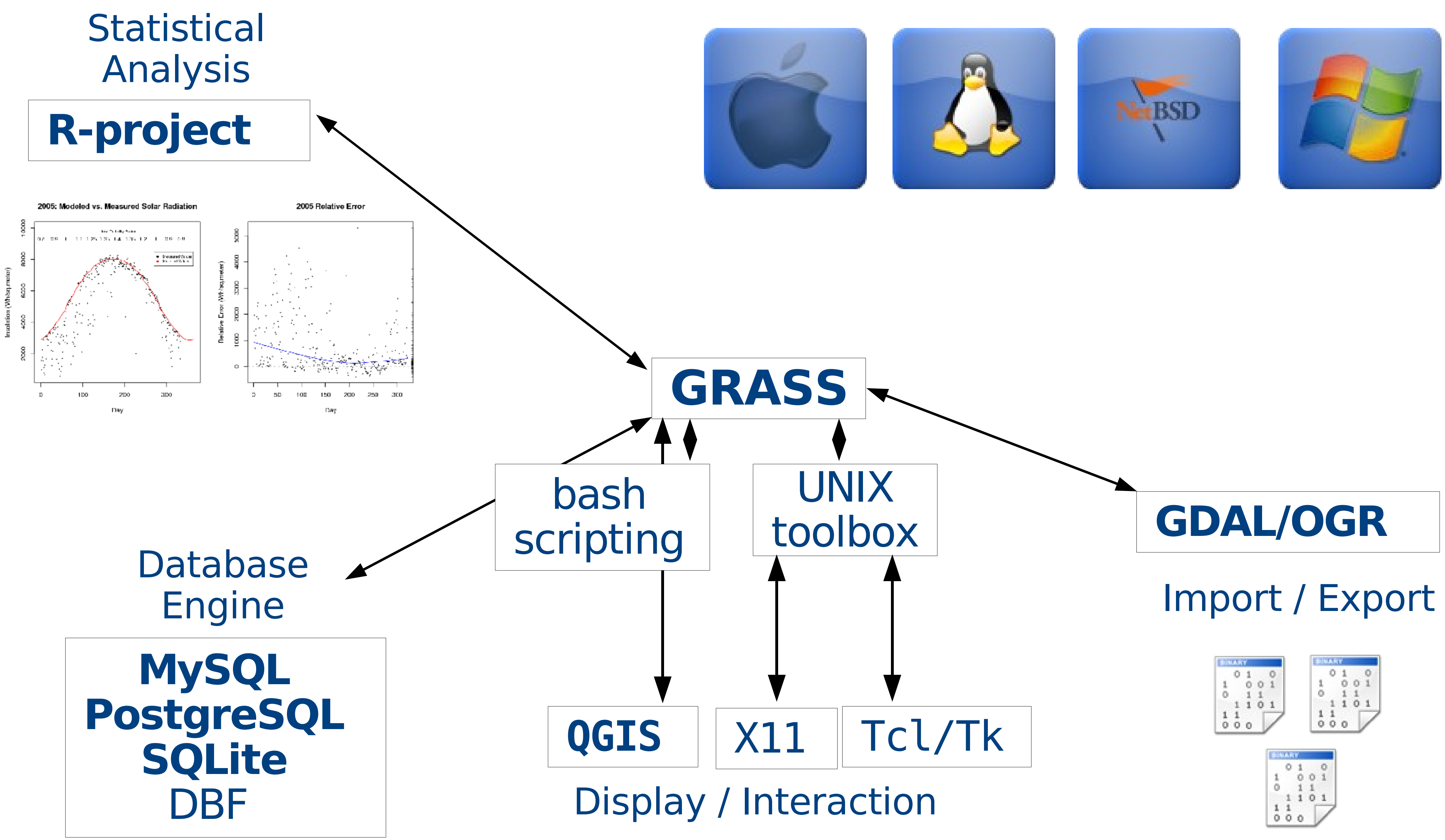
Faces



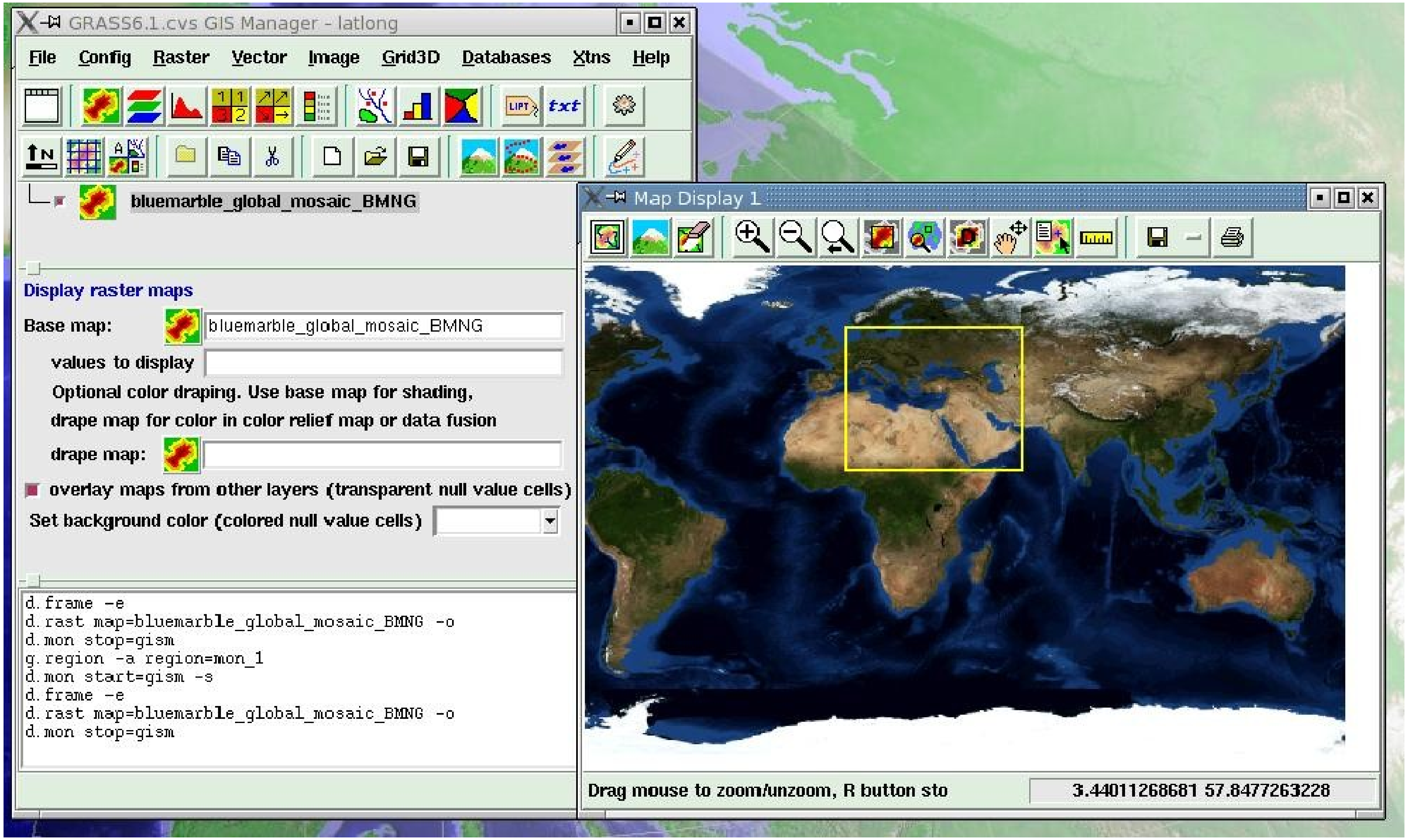




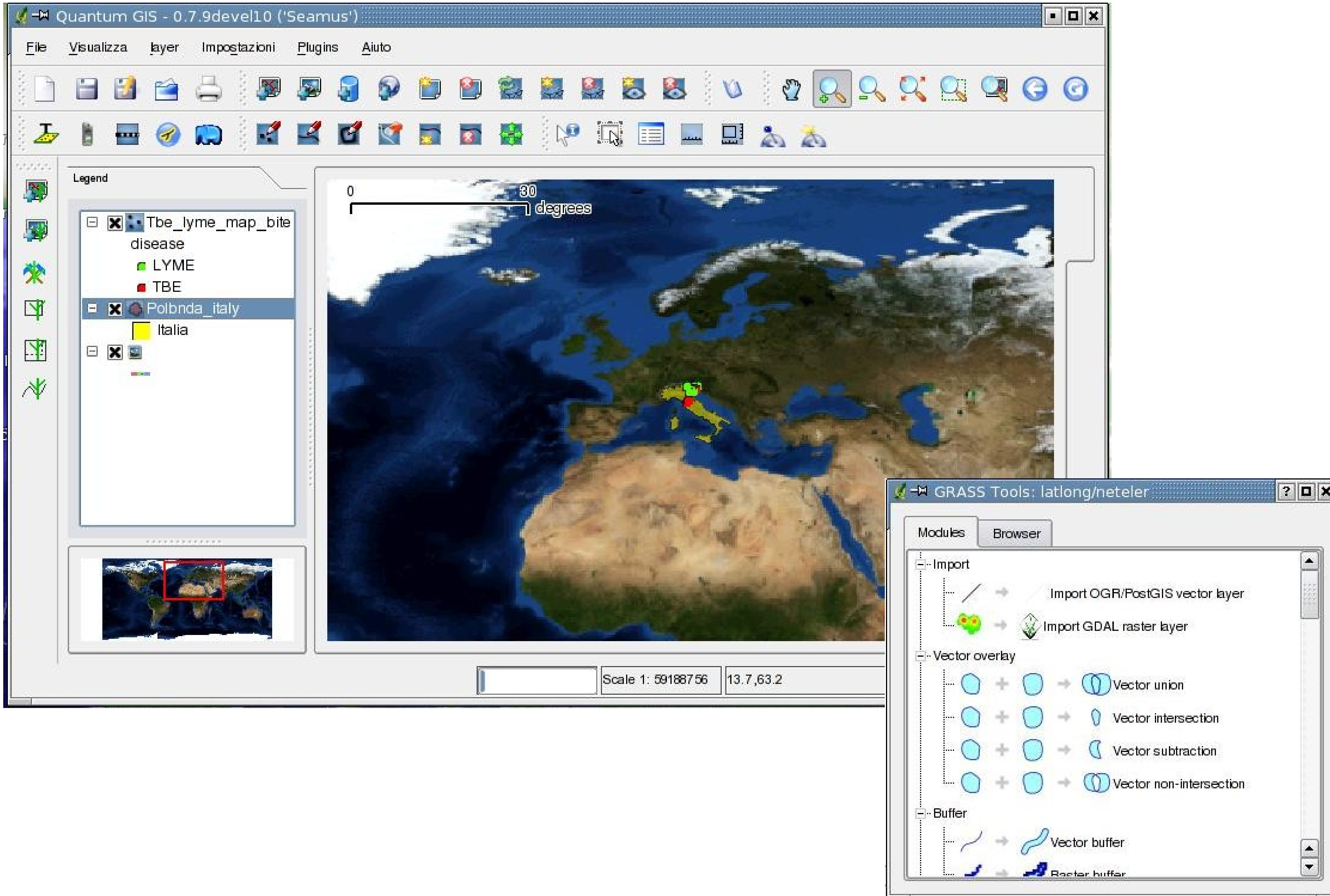
GRASS Basics: Component Model



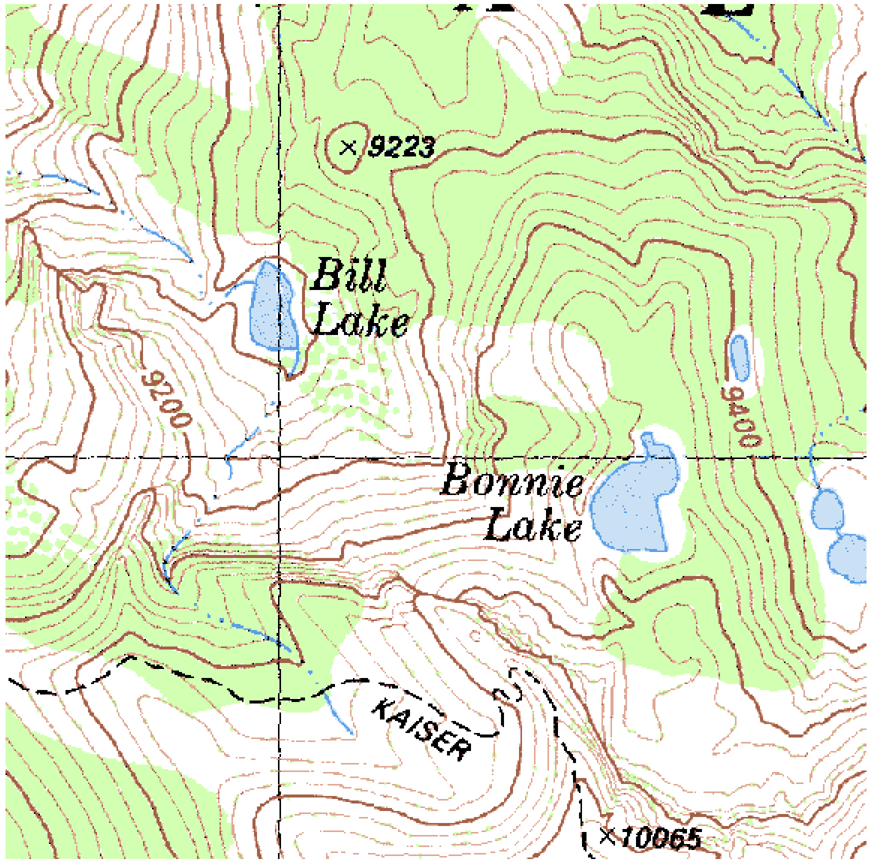
GRASS Basics: Screen Shots



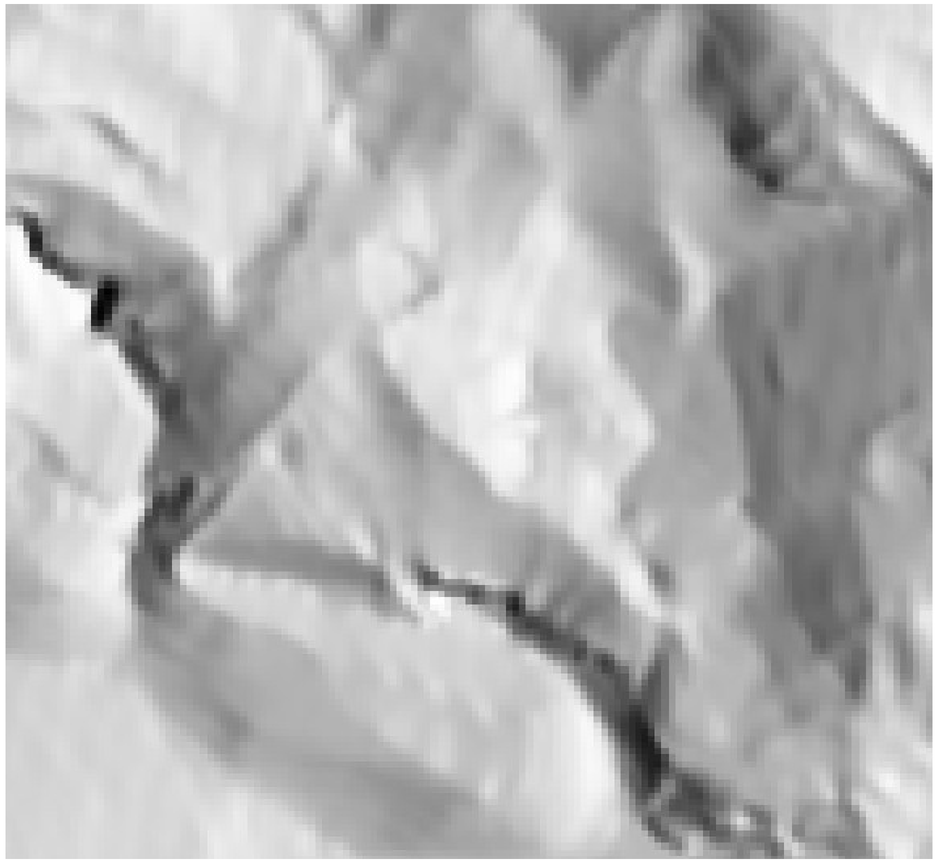
GRASS Basics: Screen Shots



GRASS Basics: Sample Application



Digital **R**aster **G**raphic
USGS Topo-map



Digital **E**levation **M**odel
USGS elevation data

Objectives:

Use freely available DRG and DEM products to assist with navigation in wilderness area

Route planning based on cost surface and network analysis

GRASS Basics: Importing Data

`$grass61 grass/huntington/PERMANENT`

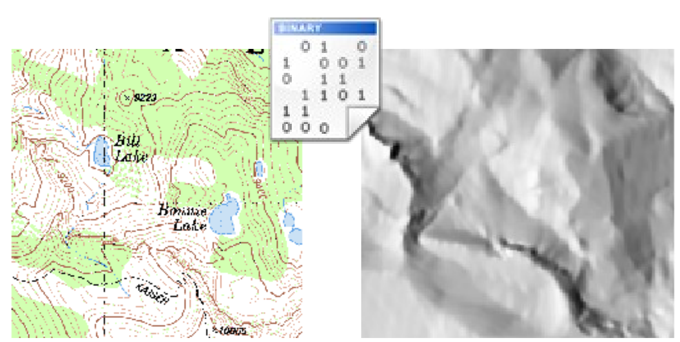
database	location	mapset
/home/dylan/grass/	huntington	PERMANENT

WMS
network raster source
`r.in.wms`

GPS
waypoints, track, routes
`v.in.garmin`
`v.in.gpsbabel`

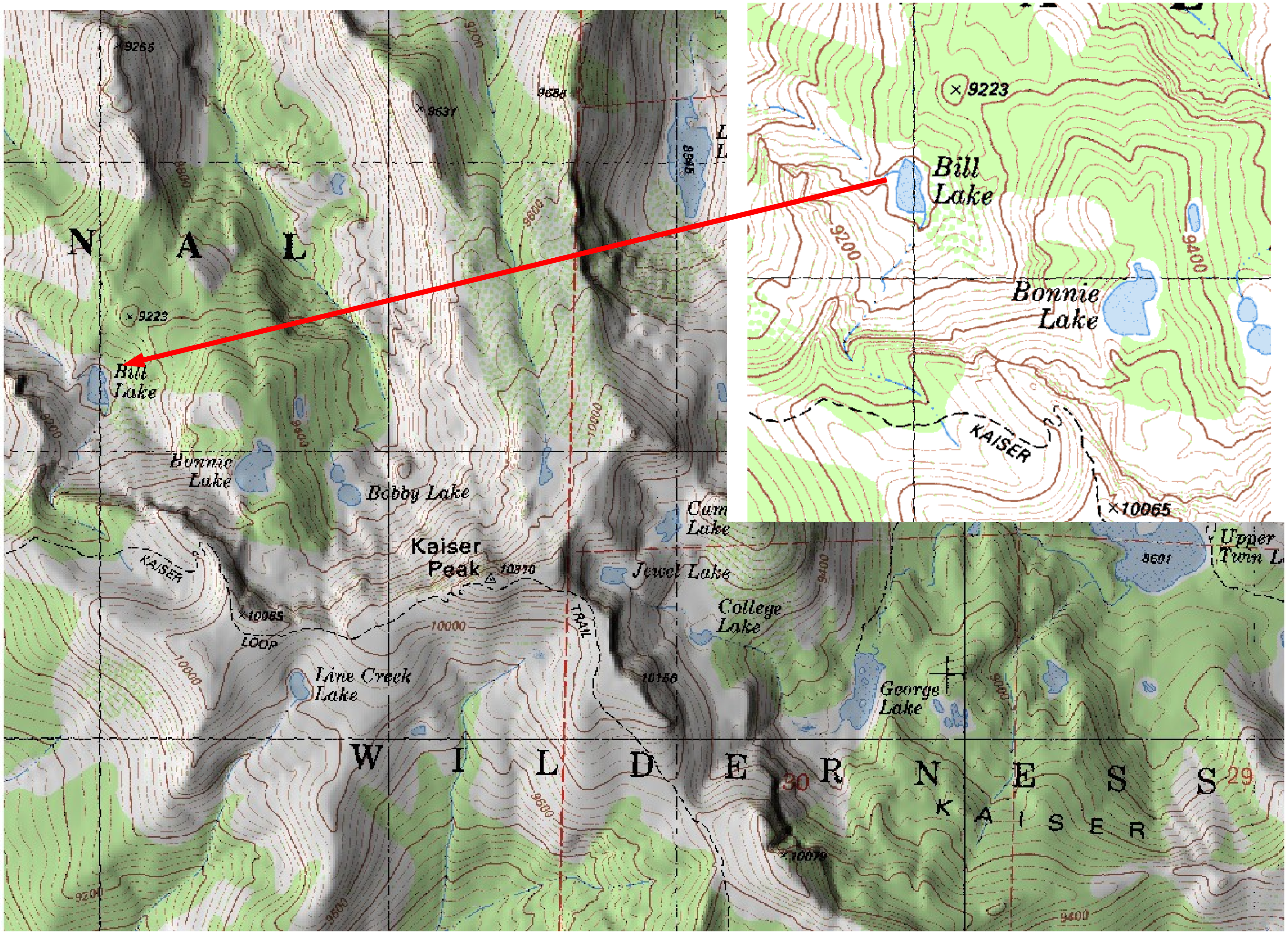
vector
points, lines, polygons
`v.in.ogr`

raster
image, terrain, ...
`r.in.gdal`



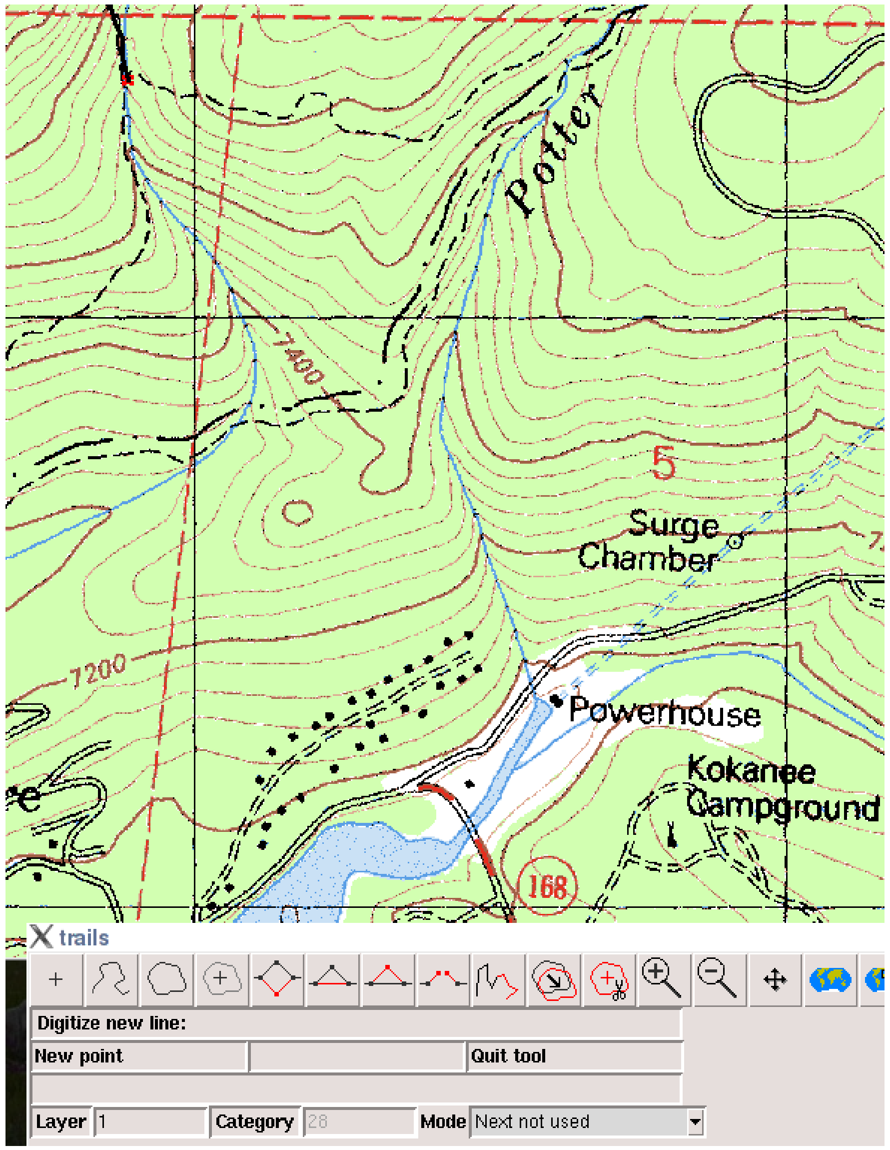
ascii
xyz lists, CSV files
`v.in.ascii`

GRASS Basics: Planning a Wilderness Adventure



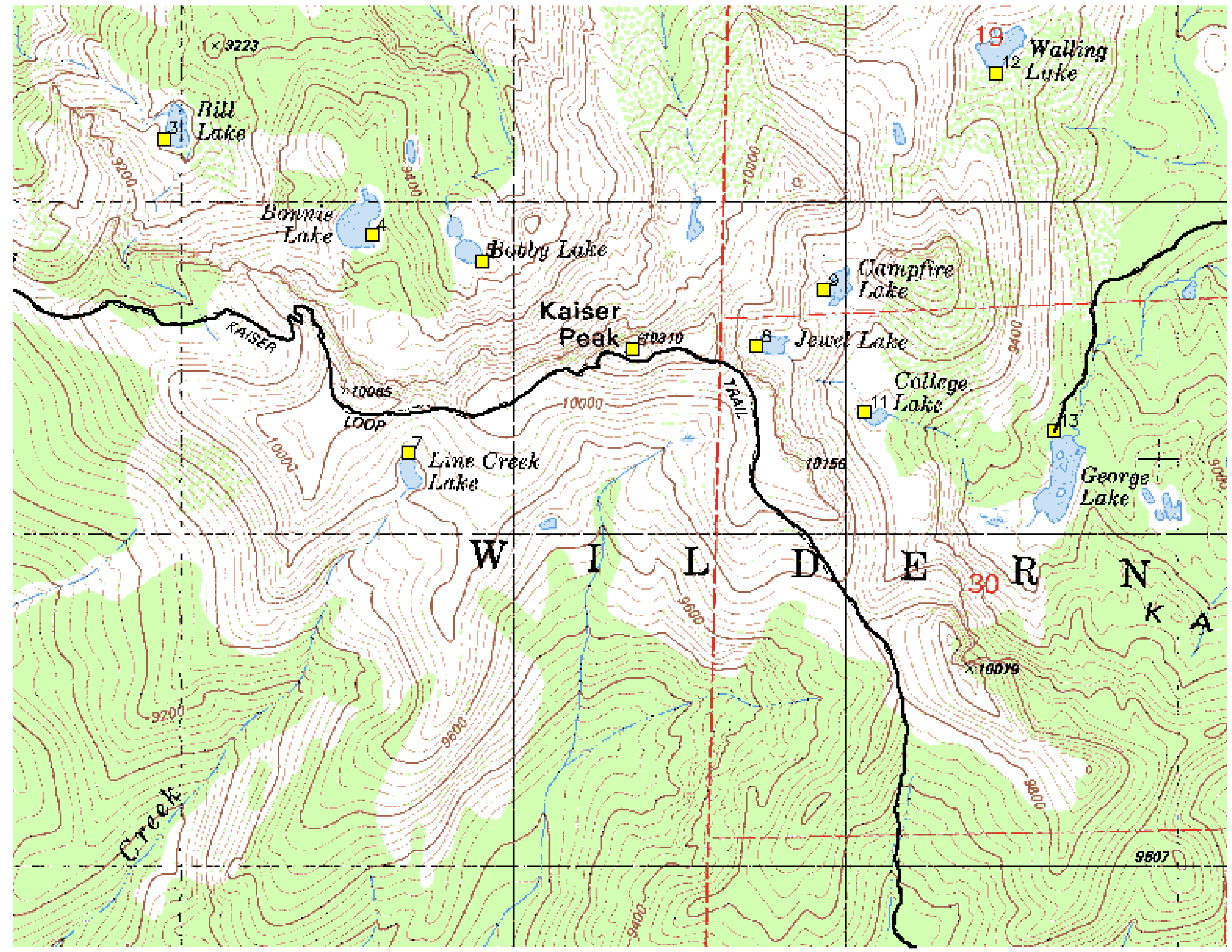
We would like to visit numerous alpine lakes located some distance from the main trail.

GRASS Basics: Digitizing Vector Features



hand digitize trails: approx 30 min
digitize points of interest: aprx 3 min

```
v.digit
```



finished trails (black lines)
and points of interest (yellow boxes)

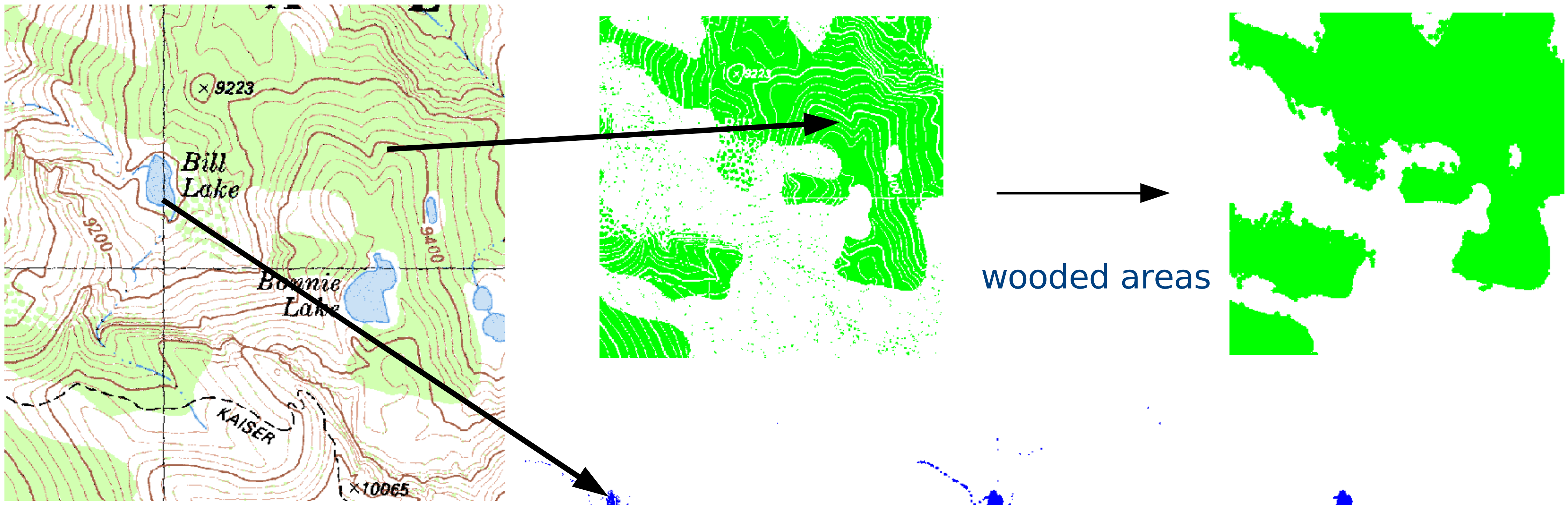
GRASS Basics: Feature Extraction from Topo-map

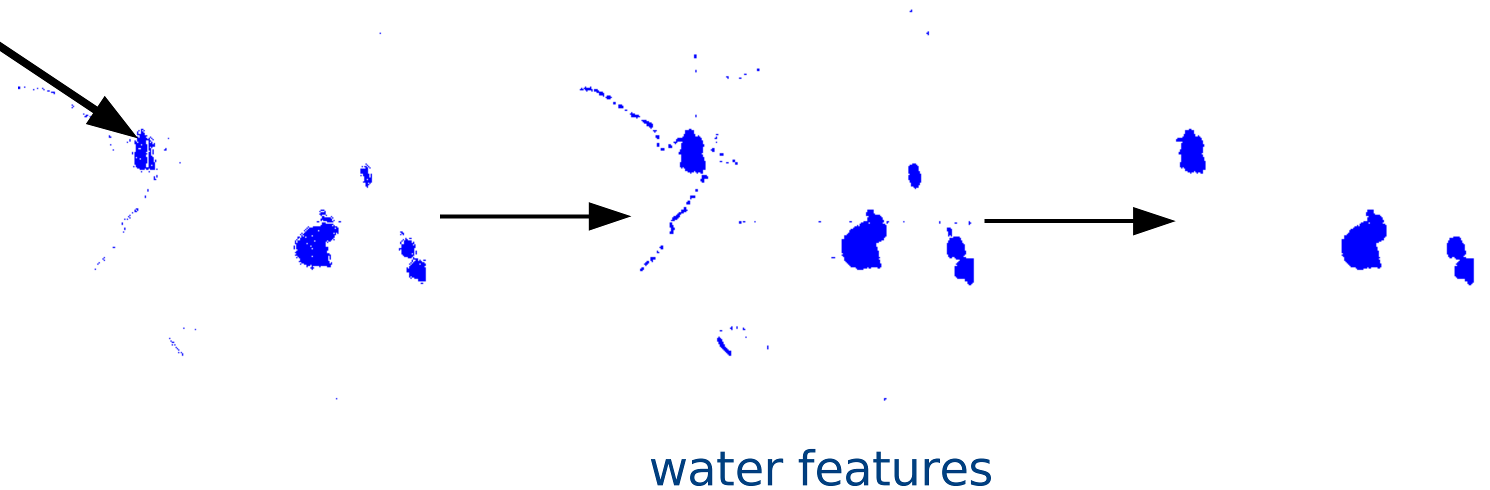
13	3	2	2	9	9	9	9	9	9	2	2	6	
2	2	2	2	9	9	9	9	9	9	2	5	2	2
5	2	2	2	9	9	9	9	9	3	2	5	6	2
2	5	2	9	9	9	9	9	9	3	2	2	2	2
2	5	2	3	9	3	9	9	9	9	2	5	6	6
2	5	2	3	9	9	9	9	9	9	2	2	2	6
2	5	2	3	9	9	3	9	9	9	3	2	5	
2	13	2	9	9	9	9	9	9	9	3	2	2	5
2	2	2	2	3	9	9	9	9	9	9	2	2	5
2	5	2	2	2	2	3	3	3	3	3	6	5	6
2	2	13	5	2	2	2	2	2	2	2	6	6	6
2	2	2	2	2	5	5	2	6	6	3	9	2	6
2	2	2	2	2	2	2	2	5	2	9	6	2	2
13	2	2	2	2	2	2	2	2	3	5	2	6	6

wooded areas

water features

GRASS Basics: Feature Extraction from Topo-map

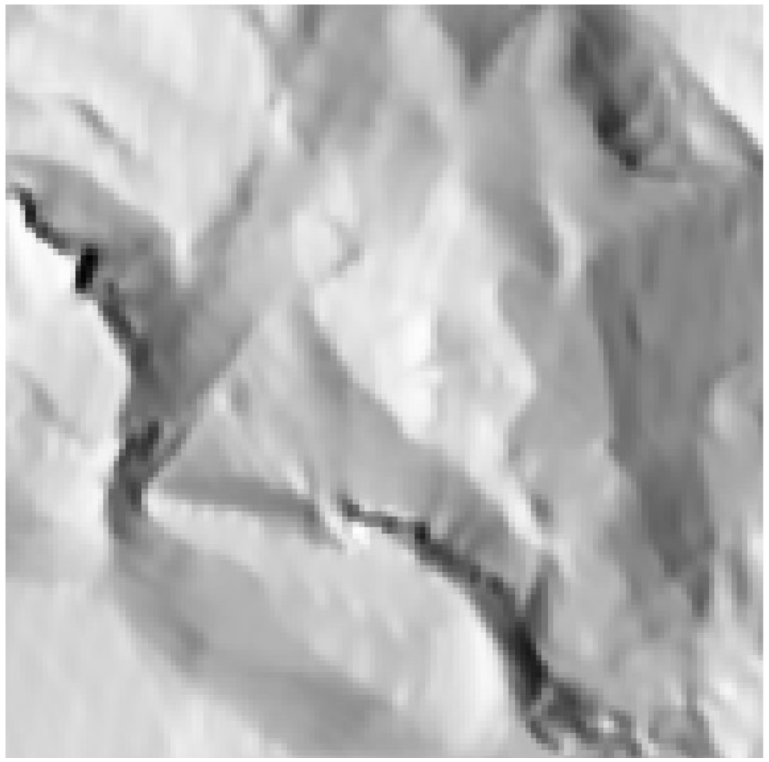




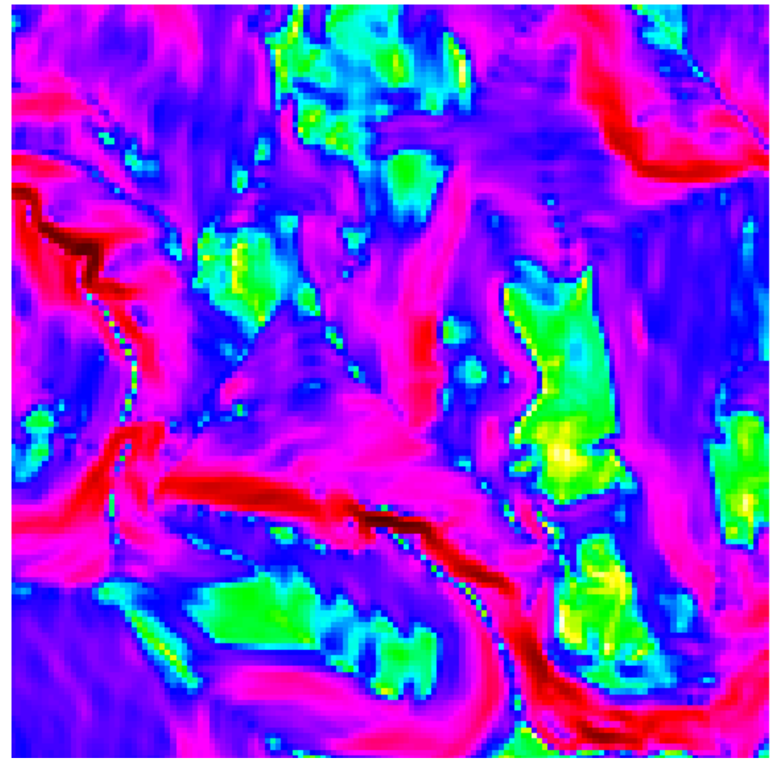

```
#reclass drg to find water features:
r.reclass in=drg out=l <<< EOF
9 = 1 lakes
* = NULL
EOF

#grow water features, to make them contiguous (remove some noise)
r.grow in=l out=lg radius=3

#make the grown water features blue:
echo "1 blue " | r.colors map=lg color=rules

#find lakes by isolating areas greater than 0.3ha
r.reclass.area in=lg out=lakes greater=0.3
```

`r.reclass` `r.grow` `r.reclass.area`

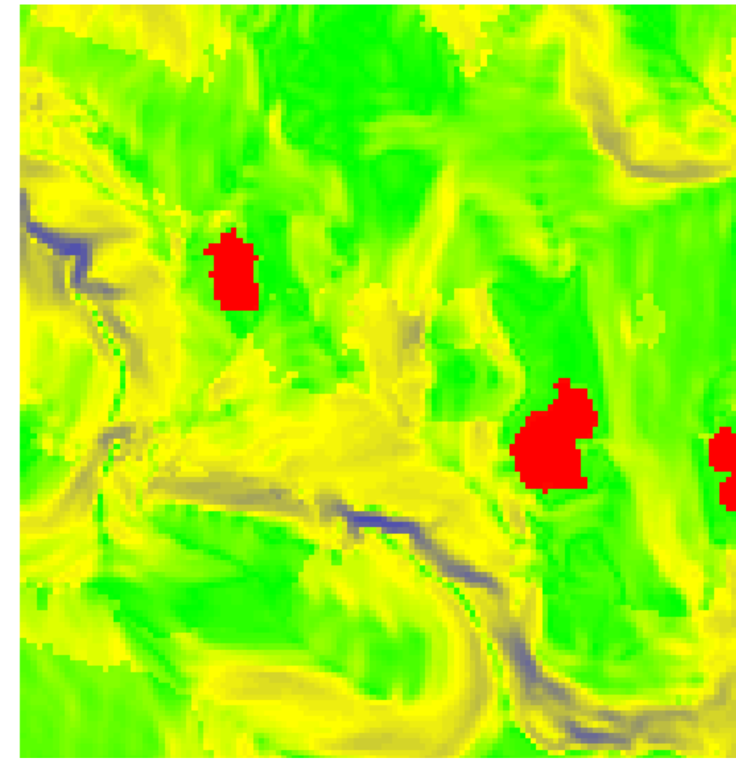
GRASS Basics: Generate Travel “Friction” Map



`r.shaded.relief`



`r.slope.aspect`



`r.mapcalc`

```
#update our slope map, to include traversing water features, and prefering wooded areas
#add a "cost" of 1000 to lake areas
r.mapcalc "new_slope = if(isnull(lakes) == 0, 1000.0+slope, slope)"

#subtract a small amount of cost for wodded areas:
r.mapcalc "new_slope = if(isnull(trees_final) == 0, abs(new_slope - 10.0), new_slope)"
```

GRASS Basics: Feature Extraction from Topo-map

13	3	2	2	9	9	9	9	9	9	2	2	6	
2	2	2	2	9	9	9	9	9	9	2	5	2	2
5	2	2	2	9	9	9	9	9	3	2	5	6	2
2	5	2	9	9	9	9	9	9	3	2	2	2	2
2	5	2	3	9	3	9	9	9	9	2	5	6	6
2	5	2	3	9	9	9	9	9	9	2	2	2	6
2	5	2	3	9	9	3	9	9	9	3	2	5	
2	13	2	9	9	9	9	9	9	9	3	2	2	5
2	2	2	2	3	9	9	9	9	9	9	2	2	5
2	5	2	2	2	2	3	3	3	3	3	6	5	6
2	2	13	5	2	2	2	2	2	2	2	6	6	6
2	2	2	2	2	5	5	2	6	6	3	9	2	6
2	2	2	2	2	2	2	2	5	2	9	6	2	2
13	2	2	2	2	2	2	2	2	3	5	2	6	6

GRASS Basics: Locating Least Cost Path

2.0	4.1	6.2	7.6	2.1	3.3	4.0	5.9	6.9	6.9	8.1	10.6	11.9	12.5
6.8	7.3	8.4	8.4	0.9	2.4	4.1	6.8	8.0	7.2	8.3	10.1	10.9	11.5
7.5	7.6	9.1	9.8	3.4	3.1	3.3	6.6	8.6	7.9	8.3	9.5	10.1	10.7
7.4	8.3	9.6	11.4	2.2	3.8	4.6	6.7	7.9	7.8	8.0	9.0	9.6	10.2
9.8	8.9	9.2	11.4	1.9	4.4	5.4	6.1	6.9	7.2	7.4	8.6	9.7	10.5
11.6	9.5	9.3	10.3	2.1	3.4	4.7	5.8	6.2	6.3	6.7	7.9	9.8	11.1
13.0	10.3	9.2	8.5	2.8	2.6	3.8	5.6	5.4	5.5	6.0	6.3	8.9	11.5
15.0	10.3	8.0	7.3	6.5	5.5	4.2	4.6	4.2	4.8	5.8	5.6	8.5	12.2
18.1	12.2	8.9	8.7	8.8	8.0	6.4	4.3	3.3	5.0	6.7	8.3	11.6	14.7
20.1	16.8	12.9	10.6	8.6	8.2	7.6	5.9	3.6	5.3	7.5	10.5	14.3	17.4
18.8	18.8	17.1	14.5	10.1	8.8	7.9	7.1	4.7	4.1	8.0	12.9	18.1	20.4
16.5	17.8	18.3	17.4	13.4	11.5	9.8	8.3	6.0	2.4	9.7	17.0	21.8	22.1
16.4	16.7	17.6	17.9	15.2	13.2	11.3	10.0	7.8	5.8	14.4	21.4	23.7	22.3
17.8	16.7	17.0	17.6	16.2	13.9	11.9	11.2	9.4	14.2	20.8	23.3	23.3	23.0

GRASS Basics: Locating Least Cost Path

2.0	4.1	6.2	7.6	1002.1	1003.3	1004.0	1005.9	1006.9	1006.9	1008.1	10.6	1.9	2.5
6.8	7.3	8.4	8.4	1000.9	1002.4	1004.1	1006.8	1008.0	1007.2	8.3	0.1	0.9	1.5
7.5	7.6	9.1	9.8	1003.4	1003.1	1003.3	1006.6	1008.6	1007.9	8.3	0.5	0.1	0.7
7.4	8.3	9.6	1011.4	1002.2	1003.8	1004.6	1006.7	1007.9	1007.8	8.0	1.0	0.4	0.2
9.8	8.9	9.2	11.4	1001.9	1004.4	1005.4	1006.1	1006.9	1007.2	7.4	1.4	0.3	0.5
11.6	9.5	9.3	10.3	1002.1	1003.4	1004.7	1005.8	1006.2	1006.3	1006.7	7.9	0.2	1.1
13.0	10.3	9.2	8.5	1002.8	1002.6	1003.8	1005.6	1005.4	1005.5	1006.0	6.3	1.1	1.5
15.0	10.3	8.0	7.3	1006.5	1005.5	1004.2	1004.6	1004.2	1004.8	1005.8	4.4	1.5	12.2
18.1	12.2	8.9	8.7	1008.8	1008.0	1006.4	1004.3	1003.3	1005.0	1006.7	1.7	1.6	4.7
20.1	16.8	12.9	10.6	1008.6	1008.2	1007.6	1005.9	1003.6	1005.3	997.5	0.5	4.3	7.4
18.8	18.8	17.1	14.5	10.1	1.2	2.1	2.9	5.3	5.9	2.0	2.9	8.1	10.4
16.5	17.8	18.3	17.4	13.4	1.5	0.2	1.7	4.0	7.6	0.3	7.0	11.8	12.1
16.4	16.7	17.6	17.9	15.2	13.2	1.3	0.0	2.2	4.2	4.4	11.4	13.7	12.3
17.8	16.7	17.0	17.6	16.2	13.9	1.9	1.2	0.6	4.2	10.8	13.3	13.3	13.0

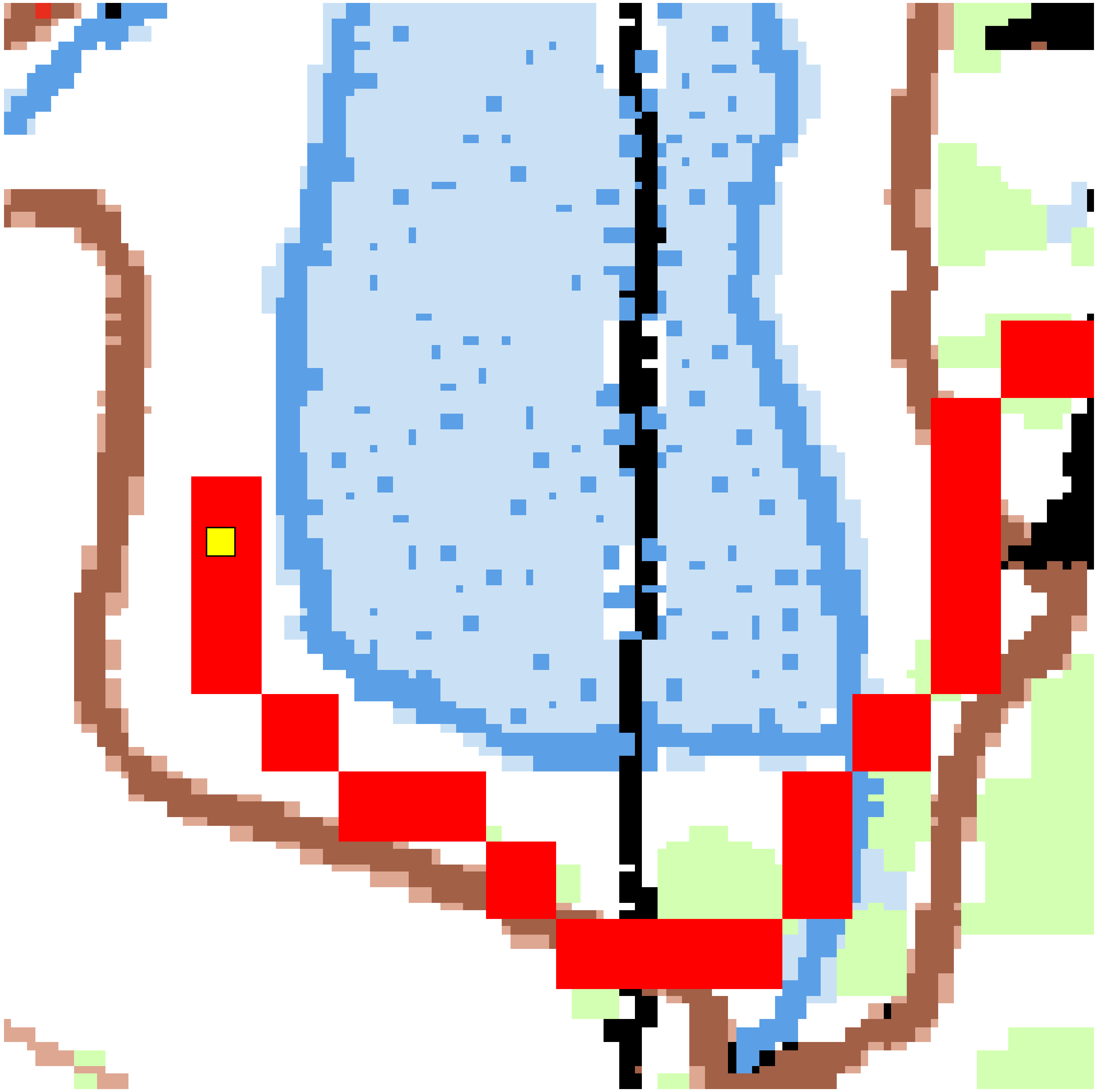
GRASS Basics: Locating Least Cost Path

53.4	55.7	58.4	565.0	1569.0	2800.7	3048.8	2824.6	1758.2	781.8	573.5	68.5	64.6	65.8
47.7	48.4	50.9	557.9	1561.2	2796.7	N	2829.3	1582.6	575.0	67.3	63.1	63.5	64.1
40.2	39.6	44.0	552.7	1558.2	2801.3	N	2828.1	1584.0	575.8	67.7	63.3	63.0	63.3
32.5	30.3	531.6	741.7	1715.0	2776.1	N	2827.1	1583.6	575.8	67.9	63.4	62.8	62.8
25.4	20.8	20.2	530.5	1536.0	2763.3	N	2827.8	1582.1	575.0	67.7	63.3	62.5	62.7
20.4	12.6	9.4	518.3	1523.2	2750.9	N	2825.7	1759.9	784.5	573.5	66.2	62.2	62.8
18.6	8.9	0.0	507.1	1511.3	2158.1	2983.8	3033.8	2207.0	1577.1	571.3	65.2	61.5	62.4
20.2	11.7	7.9	514.9	737.4	1713.1	2779.2	2801.6	2799.9	1571.8	566.5	61.4	60.3	67.1
26.0	18.3	15.9	19.4	528.2	1536.6	1559.8	1559.1	1561.0	1568.1	562.6	58.4	58.8	62.0
36.4	29.0	25.6	27.1	536.0	551.1	552.9	554.0	557.6	562.9	555.4	57.4	59.8	65.2
50.3	43.5	38.2	37.7	40.8	46.4	48.1	49.6	53.1	57.4	55.6	56.8	62.3	70.2
66.3	59.4	54.1	51.4	52.1	47.8	47.4	48.4	50.4	55.3	54.5	58.2	67.2	76.3
82.6	76.0	71.7	67.5	59.6	55.1	48.2	47.6	48.7	51.3	55.5	62.8	72.6	83.2
98.1	93.1	89.3	82.6	71.8	56.8	48.9	48.2	48.0	50.4	57.9	68.0	79.2	91.4

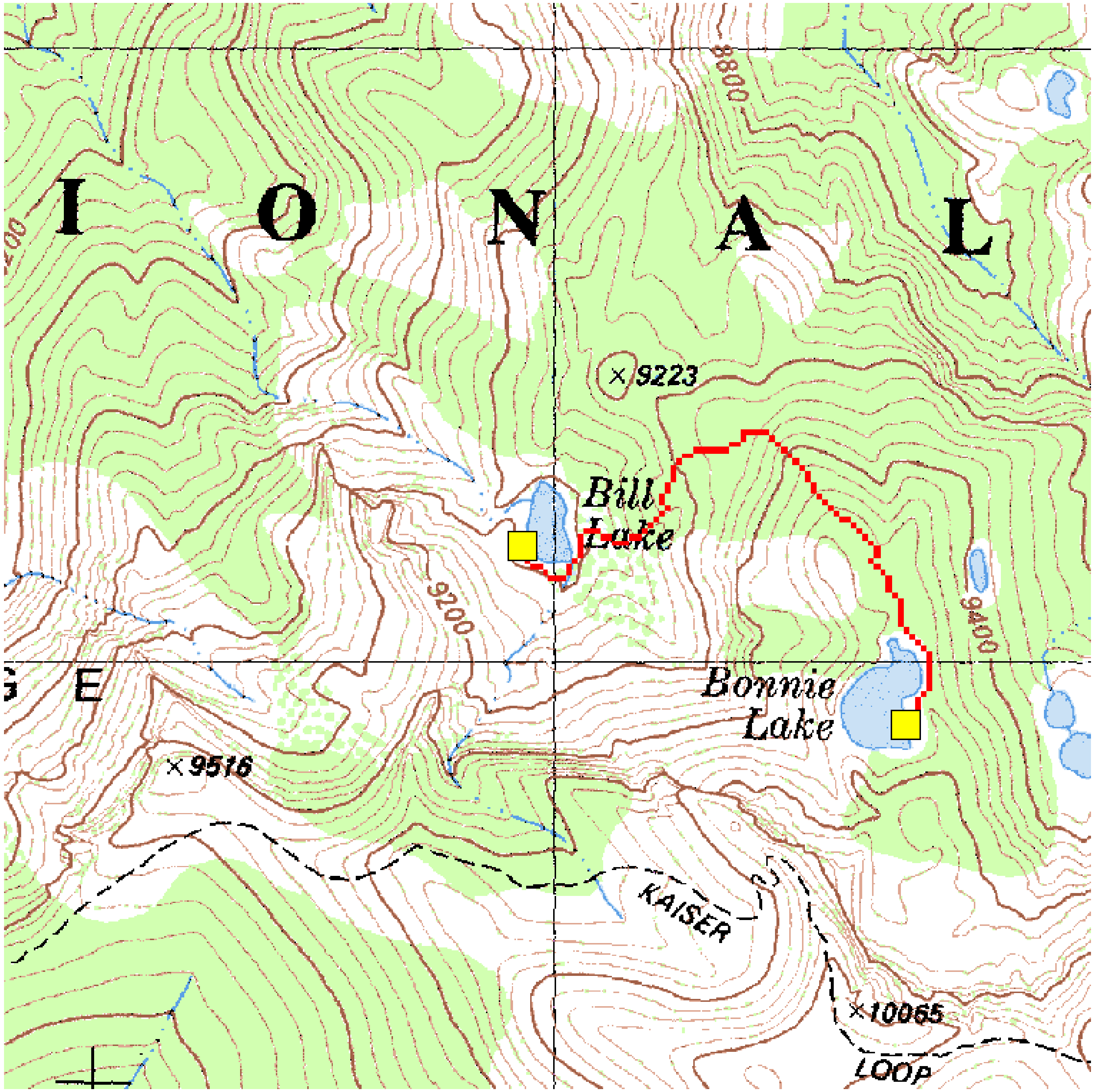
GRASS Basics: Locating Least Cost Path

53.4	55.7	58.4	565.0	1569.0	2800.7	3048.8	2824.6	1758.2	781.8	573.5	68.5	64.6	65.8
47.7	48.4	50.9	557.9	1561.2	2796.7	N	2829.3	1582.6	575.0	67.3	63.1	63.5	64.1
40.2	39.6	44.0	552.7	1558.2	2801.3	N	2828.1	1584.0	575.8	67.7	63.3	63.0	63.3
32.5	30.3	531.6	741.7	1715.0	2776.1	N	2827.1	1583.6	575.8	67.9	63.4	62.8	62.8
25.4	20.8	20.2	530.5	1536.0	2763.3	N	2827.8	1582.1	575.0	67.7	63.3	62.5	62.7
20.4	12.6	9.4	518.3	1523.2	2750.9	N	2825.7	1759.9	784.5	573.5	66.2	62.2	62.8
18.6	8.9	0.0	507.1	1511.3	2158.1	2983.8	3033.8	2207.0	1577.1	571.3	65.2	61.5	62.4
20.2	11.7	7.9	514.9	737.4	1713.1	2779.2	2801.6	2799.9	1571.8	566.5	61.4	60.3	67.1
26.0	18.3	15.9	19.4	528.2	1536.6	1559.8	1559.1	1561.0	1568.1	562.6	58.4	58.8	62.0
36.4	29.0	25.6	27.1	536.0	551.1	552.9	554.0	557.6	562.9	555.4	57.4	59.8	65.2
50.3	43.5	38.2	37.7	40.8	46.4	48.1	49.6	53.1	57.4	55.6	56.8	62.3	70.2
66.3	59.4	54.1	51.4	52.1	47.8	47.4	48.4	50.4	55.3	54.5	58.2	67.2	76.3
82.6	76.0	71.7	67.5	59.6	55.1	48.2	47.6	48.7	51.3	55.5	62.8	72.6	83.2
98.1	93.1	89.3	82.6	71.8	56.8	48.9	48.2	48.0	50.4	57.9	68.0	79.2	91.4

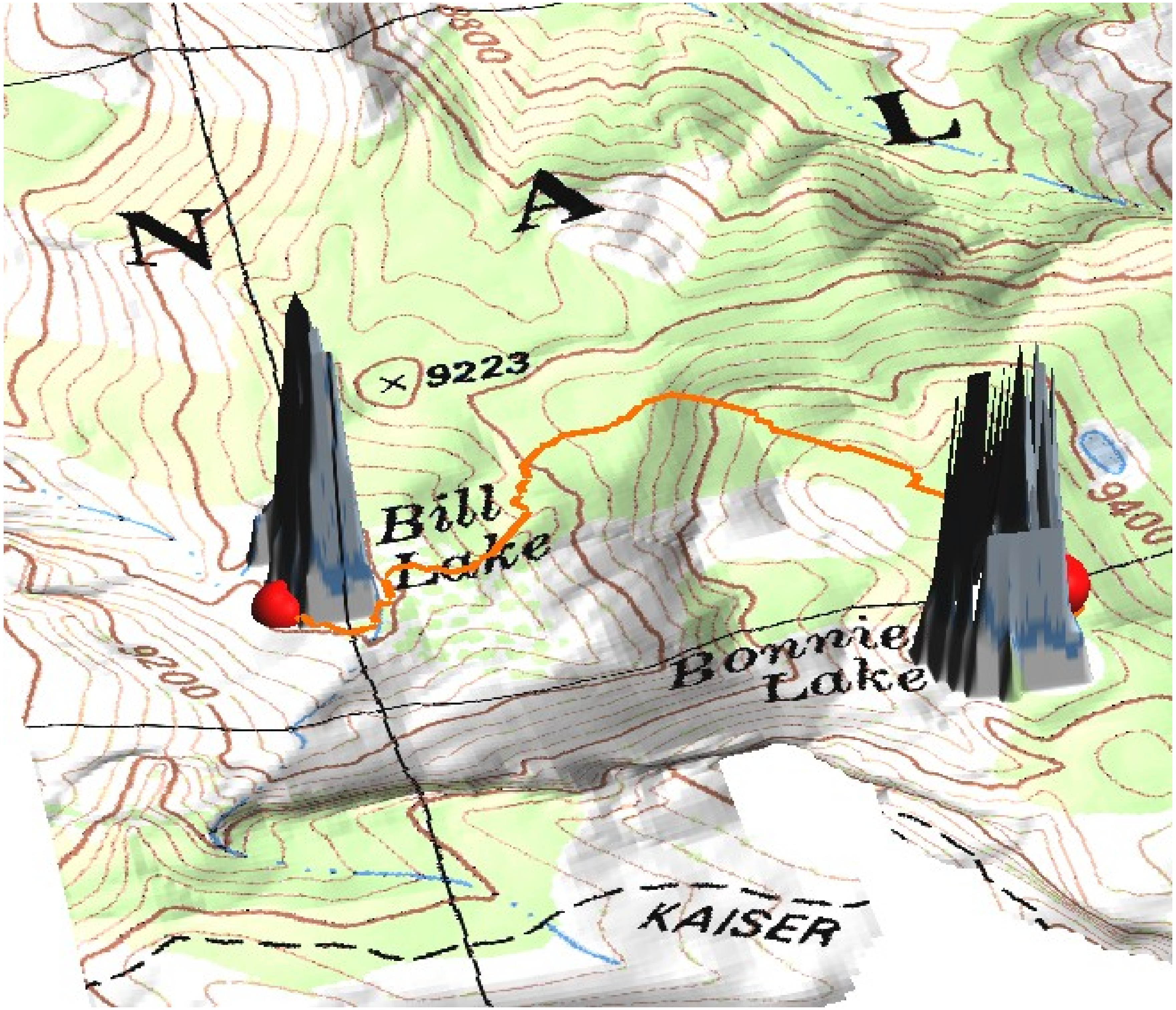
GRASS Basics: Locating Least Cost Path

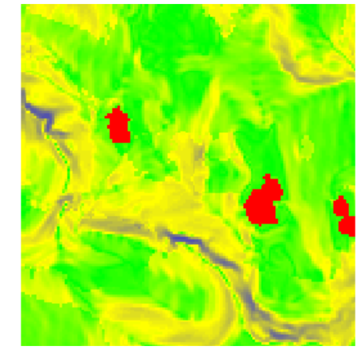


GRASS Basics: Locating Least Cost Path

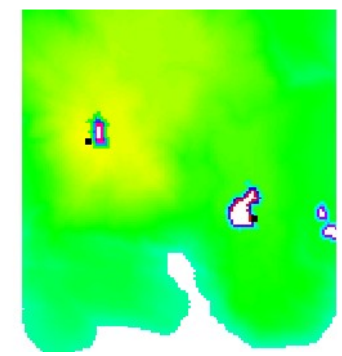


GRASS Basics: Locating Least Cost Path

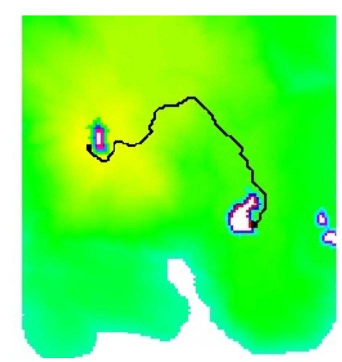




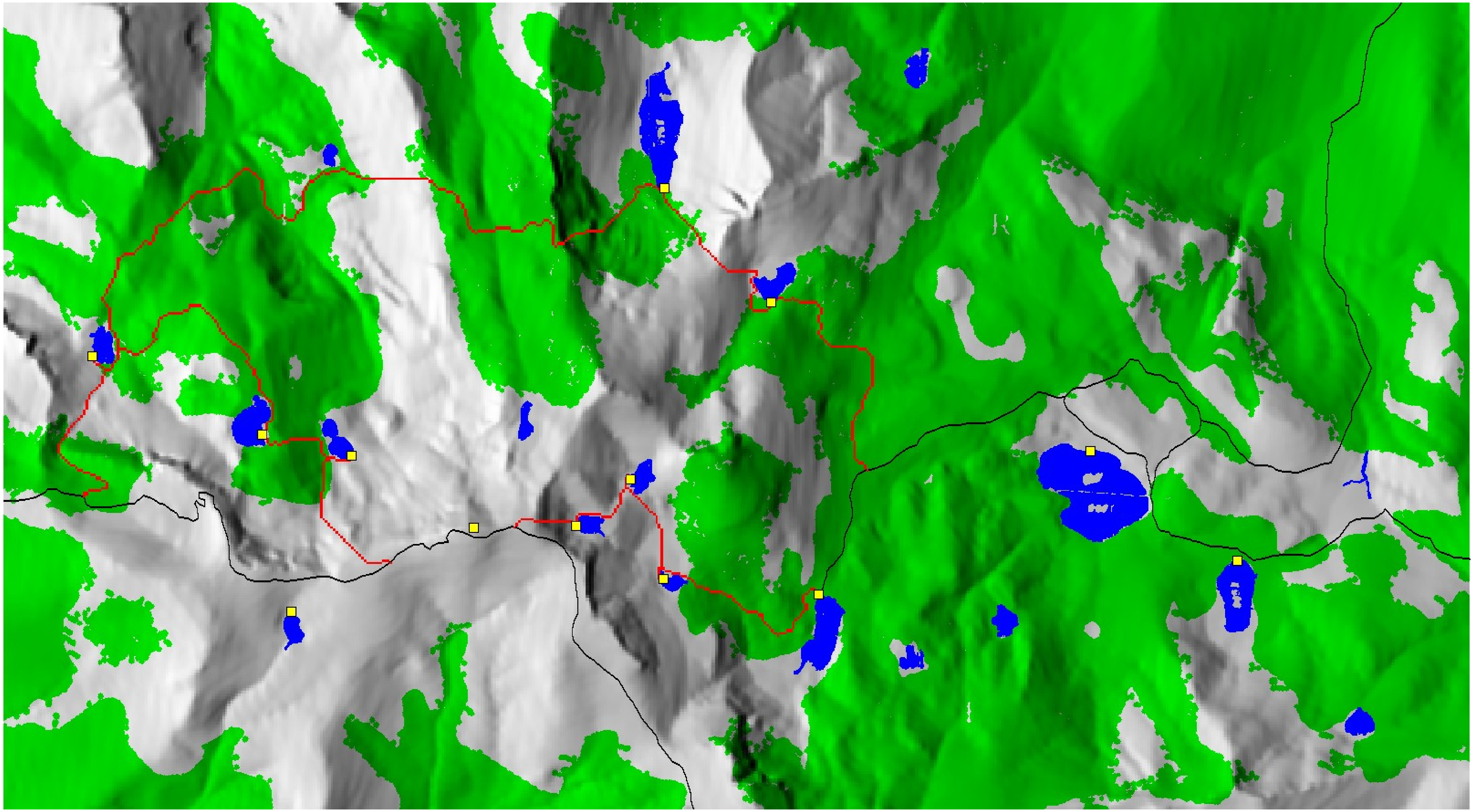
r.cost



r.drain



GRASS Basics: Least Cost Path Between POI



Red lines are the finished least cost paths between points of interest and main trail

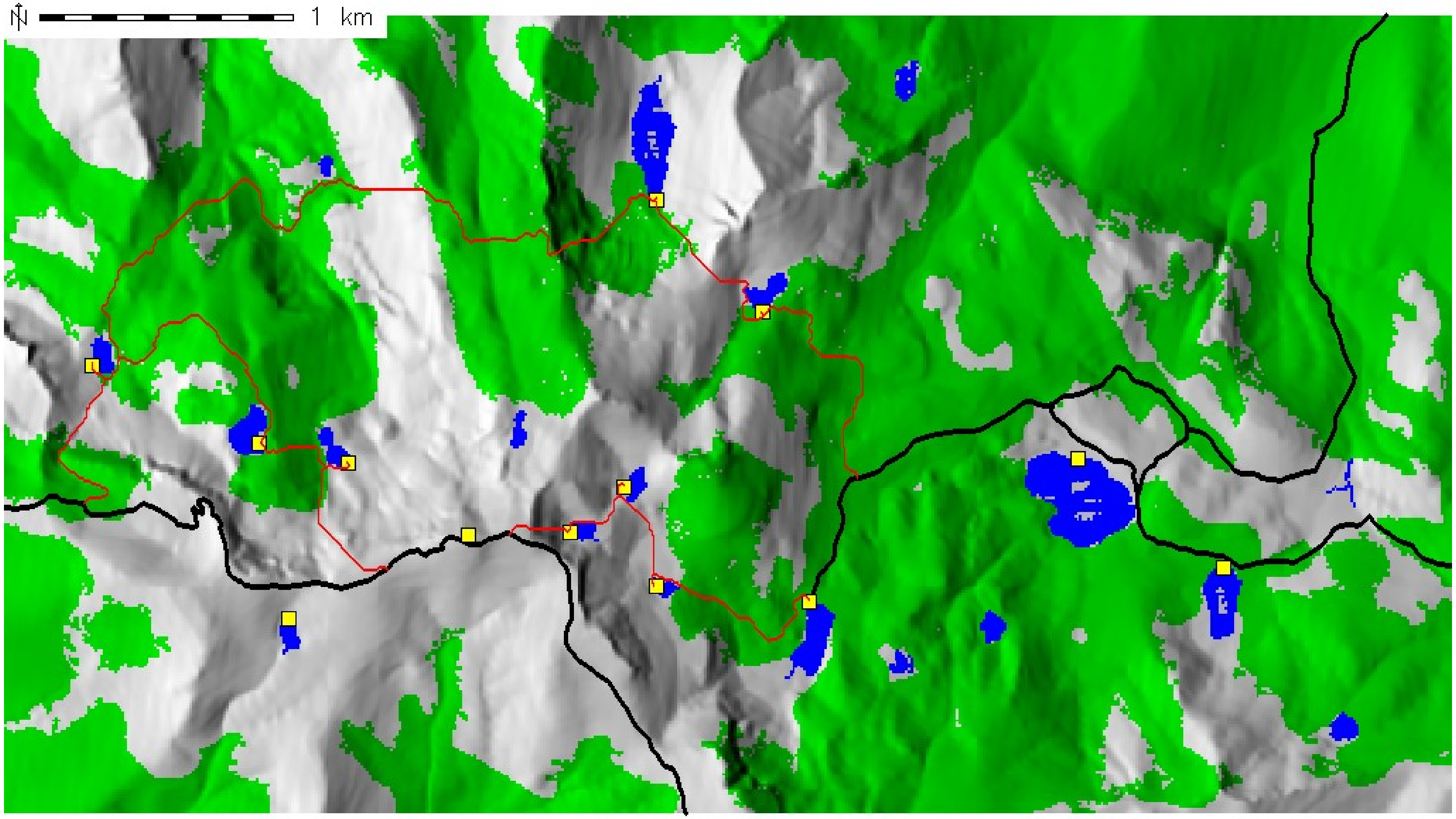
loop through POI

`r.cost` `r.drain` →

patch together and vectorize

`r.patch` → `r.to.vect`

GRASS Basics: Combine Trail Features



combine my trails with existing trails

search for breaks in connectivity

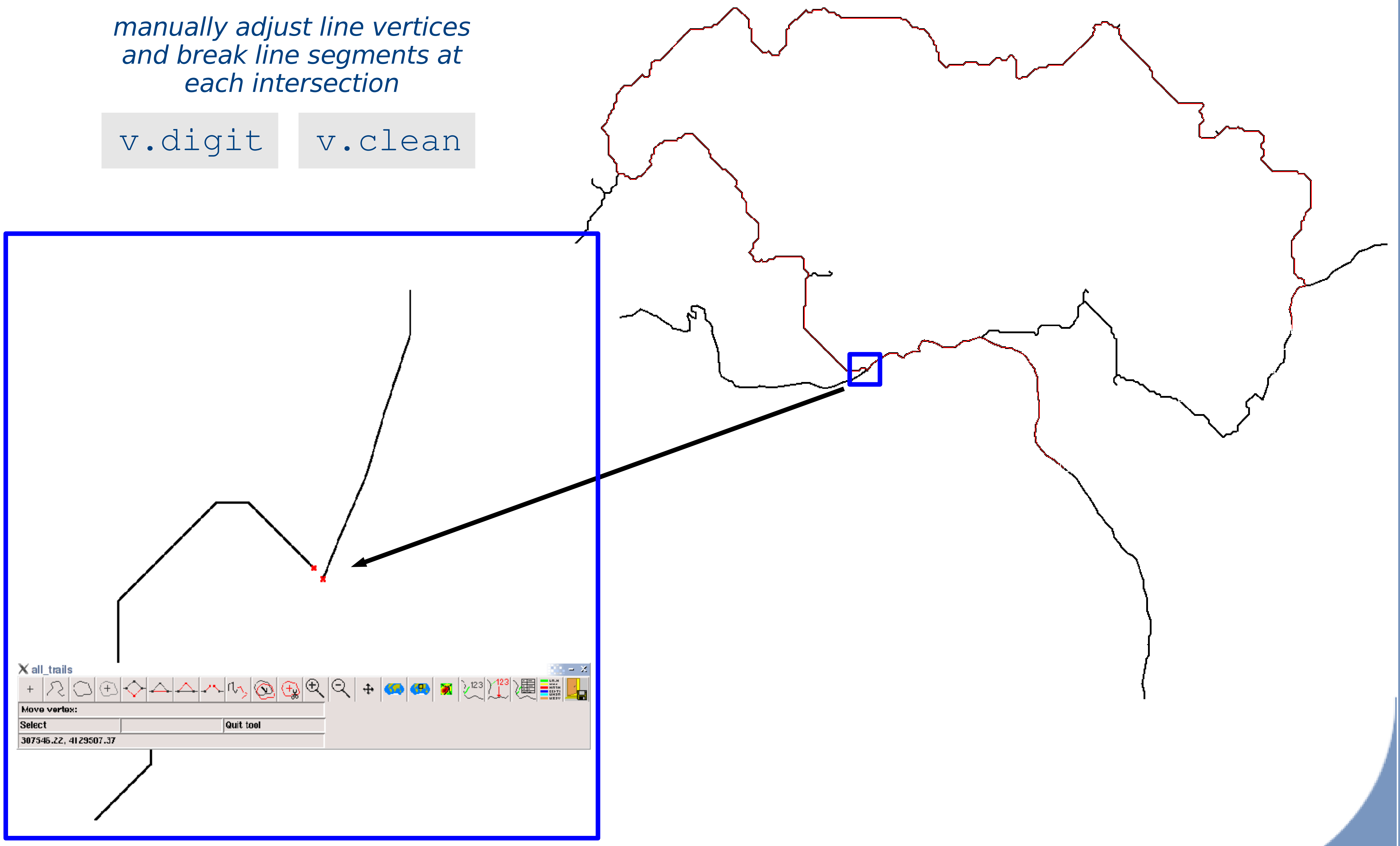
manually adjust line vertices and break line segments at each intersection

`v.patch` → `d.path` → `v.digit` `v.clean`

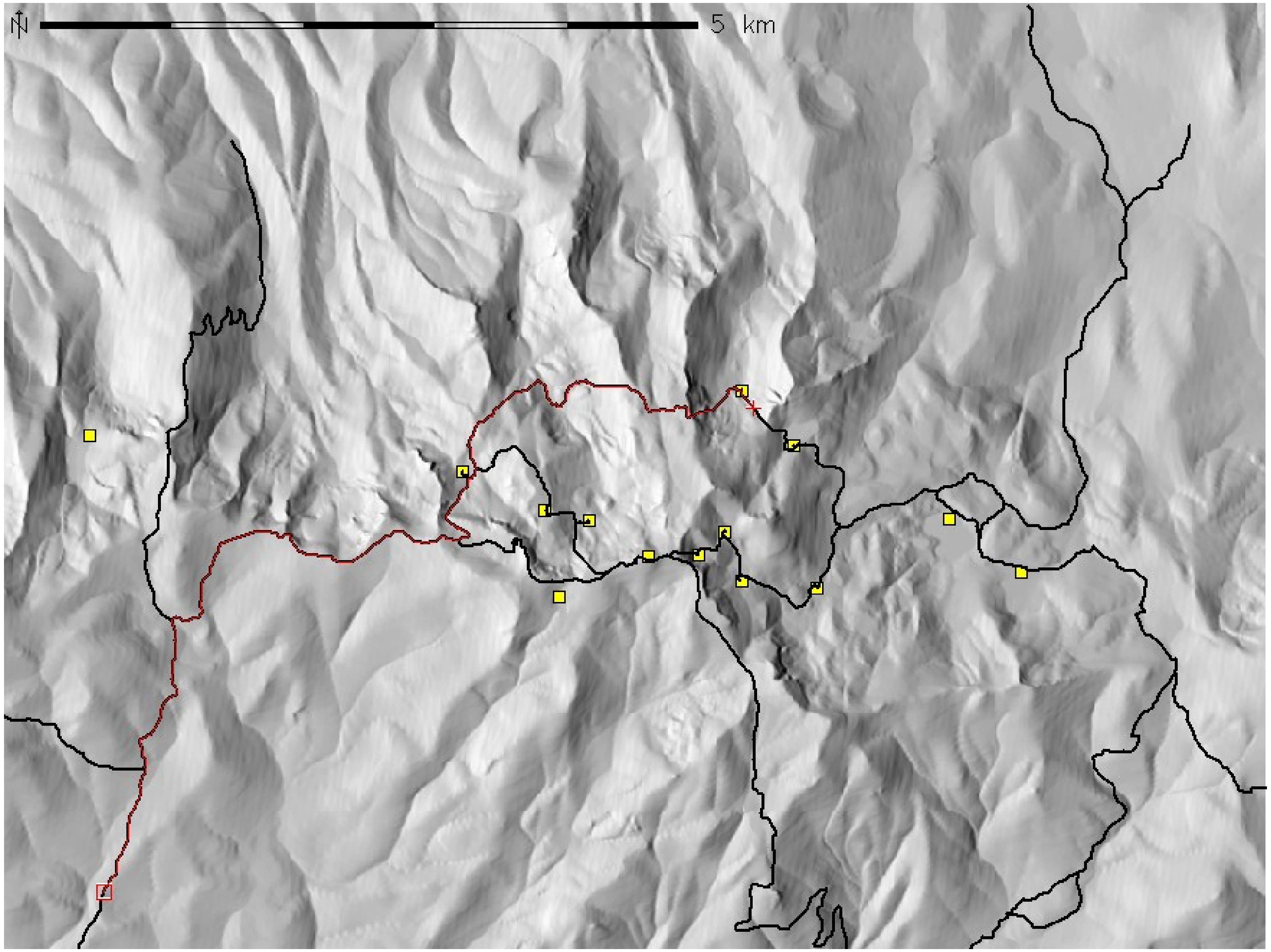
GRASS Basics: Fixing Line Topology for Network Analysis

manually adjust line vertices and break line segments at each intersection

`v.digit` `v.clean`



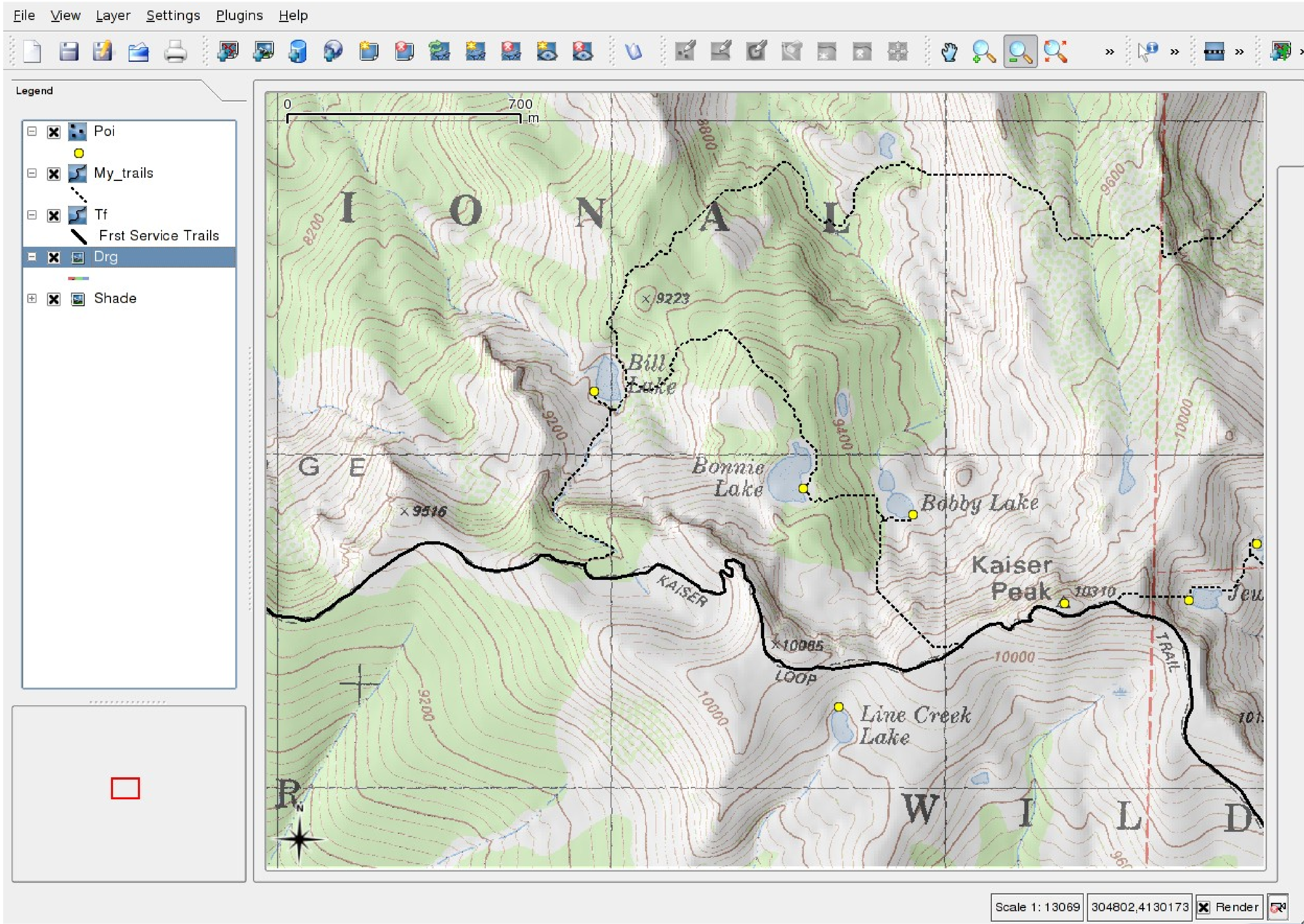
GRASS Basics: Network Analysis



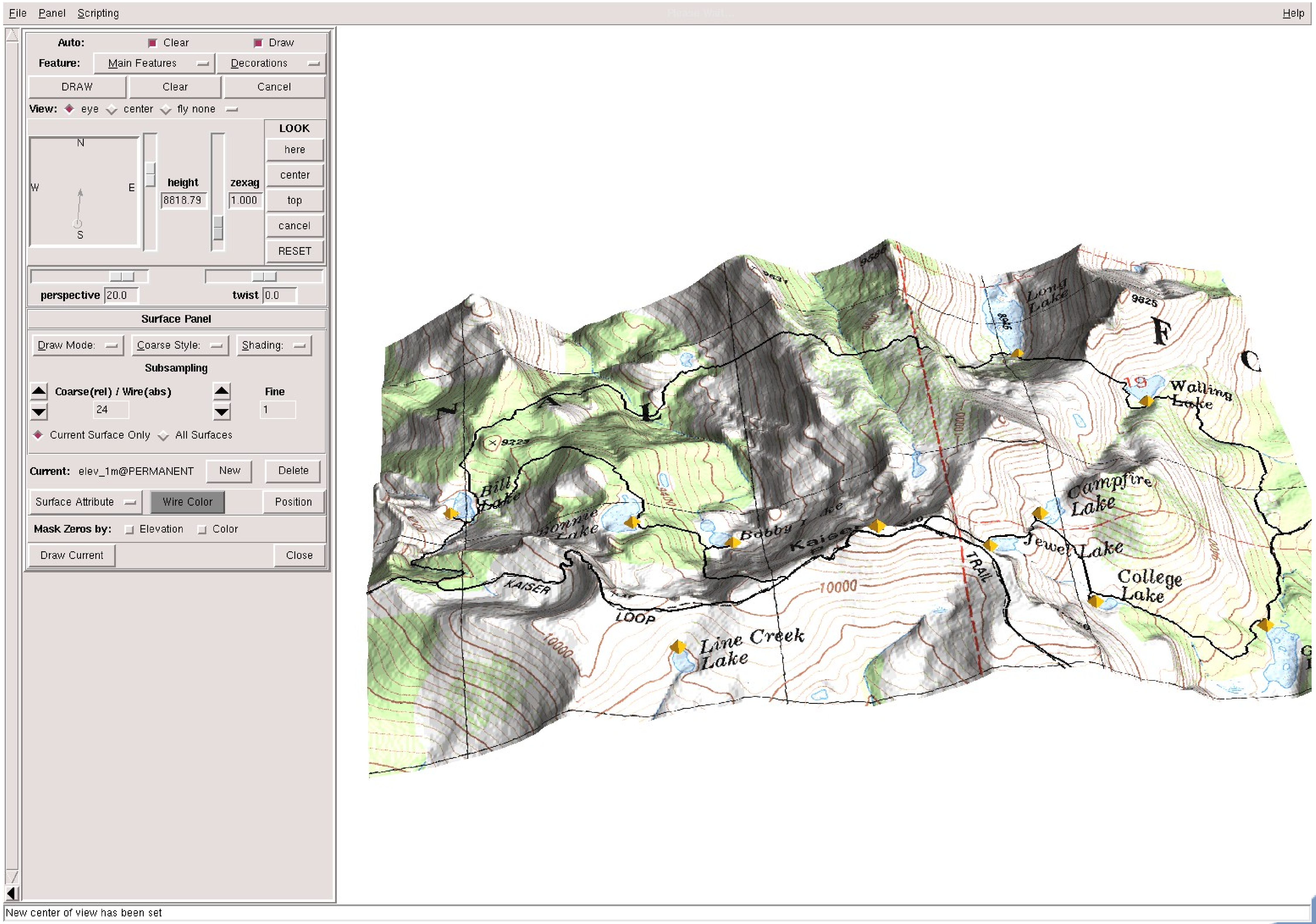
numerous tools for the selection of routes based on network analysis tools

`d.path` `v.net.path` `v.net.iso` `v.net.salesman`

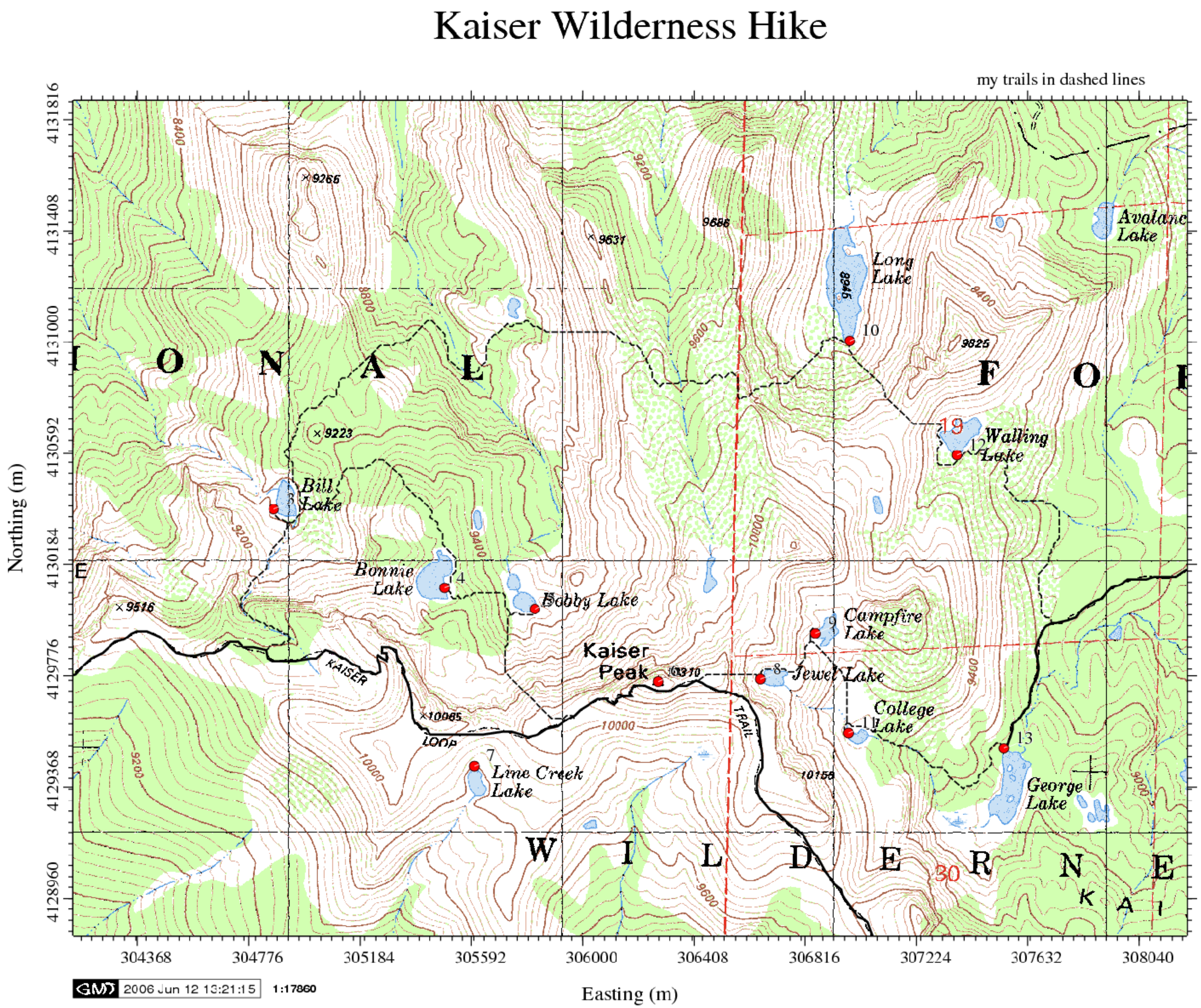
GRASS Basics: Exploring the Output in 2D with QGIS



GRASS Basics: Exploring the Output in 3D with NVIZ



GRASS Basics: Producing a Printed Map with GMT



where2.0

Further Possible Analysis

Influence of sun position on shadows / lighting

Best time of the day to take pictures of given peak, from where?

When and where will I be hiking in the shade?

If there was a massive rainfall event…

Which streams would be the safest to cross?

Where would I expect to find the most erosion?

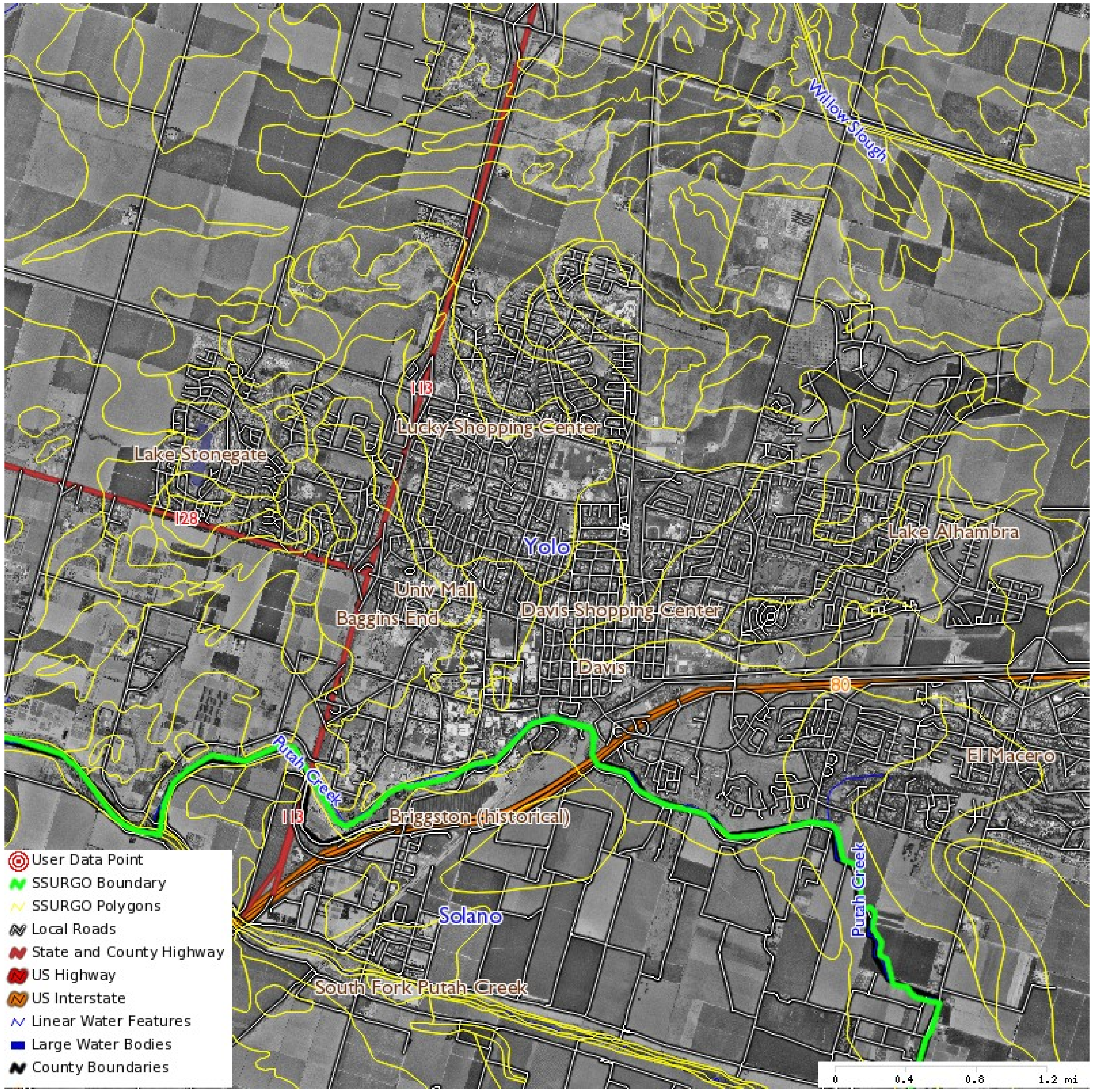
With GRASS you are limited only by your creativity

http://grass.itc.it/

debeaudette@ucdavis.edu

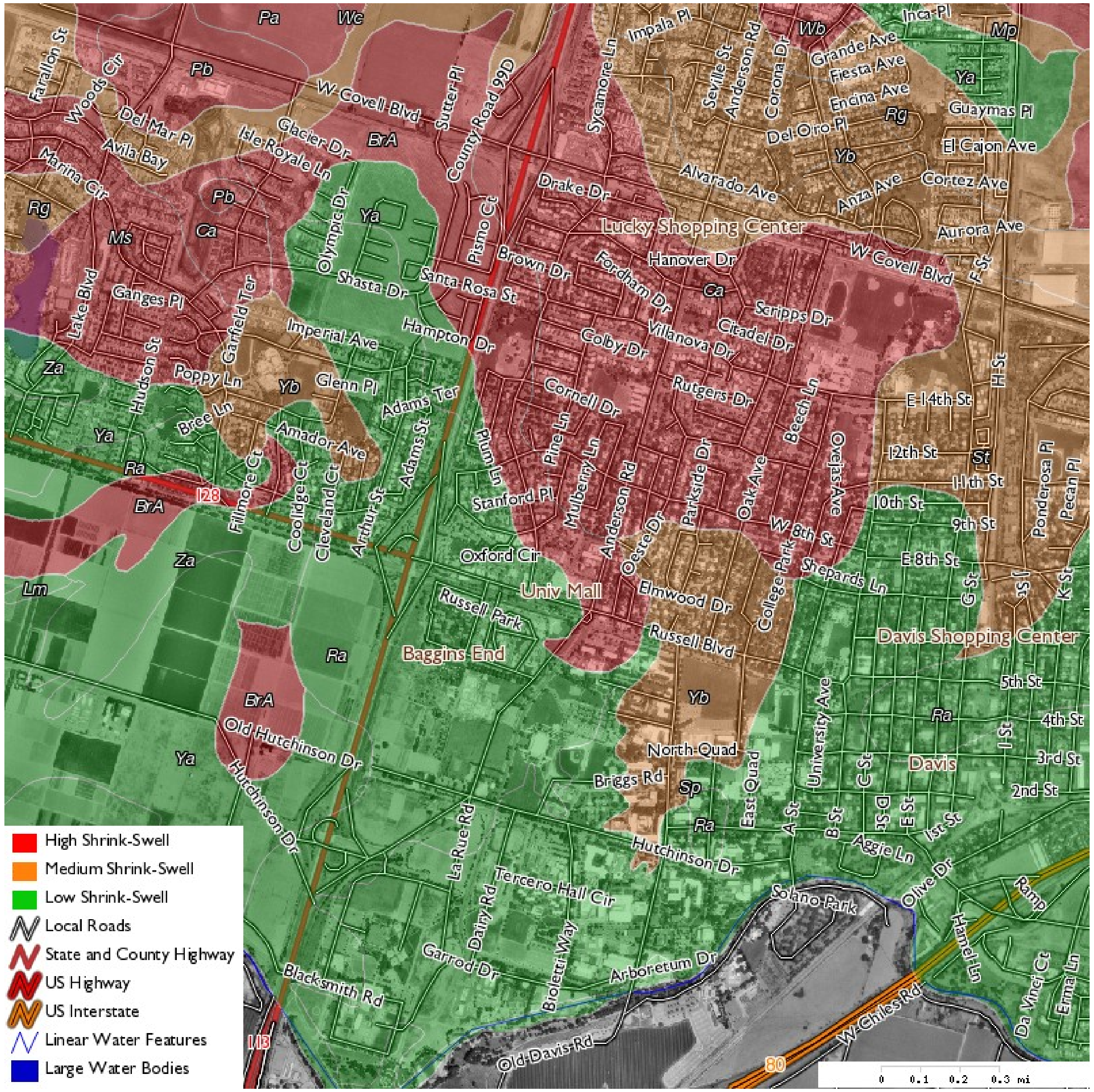
http://casoilresource.lawr.ucdavis.edu

PostGIS Examples: Soil Survey Data



http://casoilresource.lawr.ucdavis.edu/drupal/node/27

PostGIS Examples: Soil Survey Data

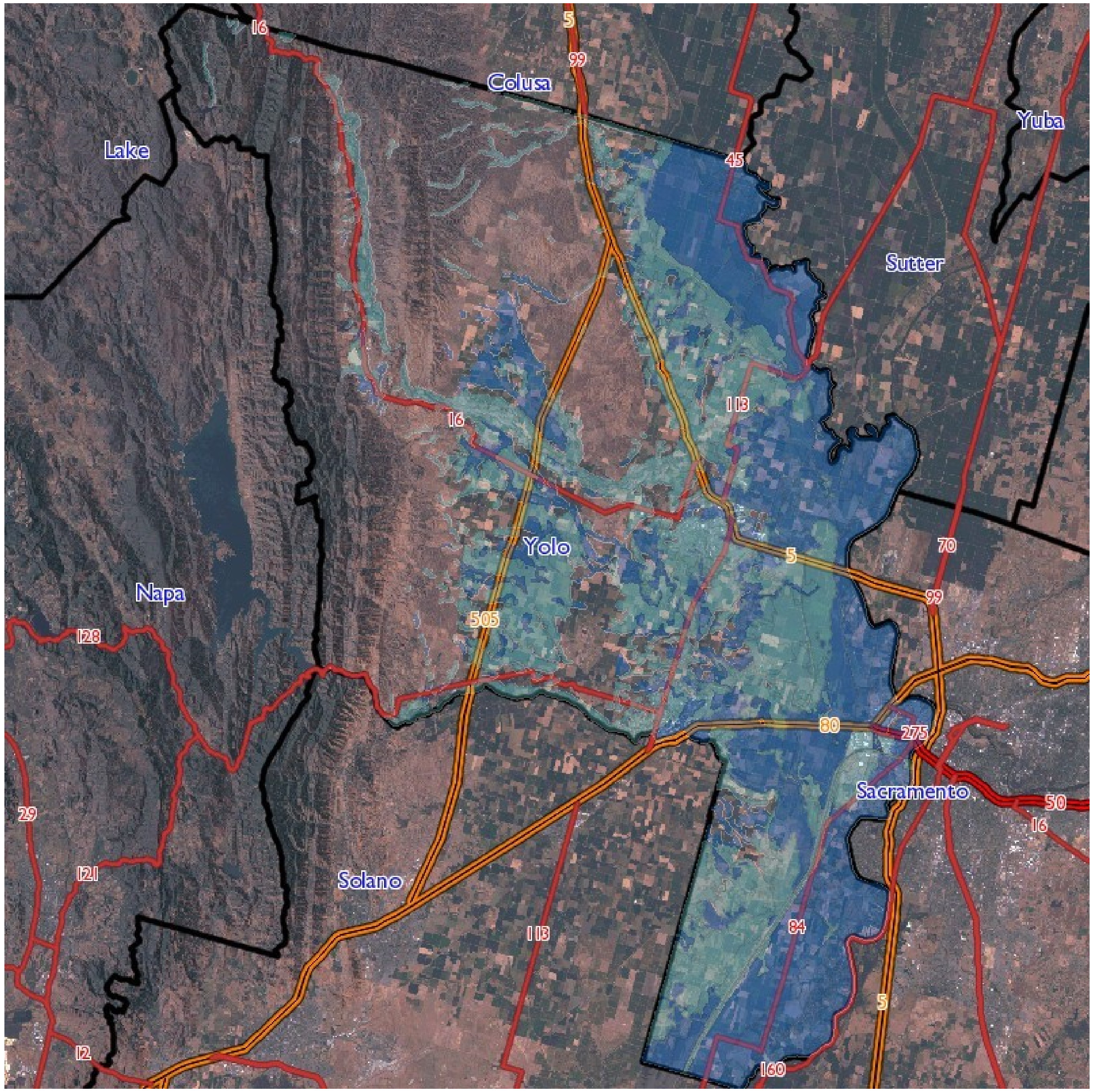



```
create table yolo_shrink_swell as
select mapunit_poly.wkb_geometry, mapunit_poly.musym,
mapunit_poly.mukey,
CASE WHEN c.lep_ss < 2 THEN 0 WHEN c.lep_ss >= 2 and c.lep_ss < 5
THEN 1 WHEN c.lep_ss >= 5 THEN 2 END as shrink_swell
from
mapunit_poly
join
(
select b.mukey, sum( b.lep_mix ) as lep_ss
from
(
select component.mukey, component.comppct_r, a.lep_r,
(component.comppct_r::float / 100) * a.lep_r as lep_mix,
CASE WHEN taxclname ~* '.*vertic.*' or taxclname ~* '.*erts' THEN 1
ELSE 0 END as vertic
from component
join
(
select cokey, hzdept_r, hzdepb_r, lep_r from chorizon where
areasymbol = 'ca113' and hzdept_r = 0
) as a on component.cokey = a.cokey
where component.majcompflag = 'Yes'
) as b
group by b.mukey
having sum( b.lep_mix ) is not NULL
) as c
on mapunit_poly.mukey = c.mukey ;
```

Location of shrink-swell soils

http://casoilresource.lawr.ucdavis.edu/drupal/node/268

PostGIS Examples: Soil Survey Data




```
create table yolo_wet_polys as
-- join to polygon data with mukey
select mapunit_poly.wkb_geometry, mapunit_poly.mukey,
c.aq_element, c.hyd_rated from
(
-- filter out only the map units that are rated with a '1' in one of the
two categories
select b.* from
(
-- aggrigate based on mukey (collapse components into single mukey)
retaining flag with max() operator
select mukey , max(aq_flag) as aq_element, max(hyd_flag) as
hyd_rated
from
(
-- flag all components in this survey area based on two criteria
select mukey, CASE WHEN taxclname ~* '.*aqu.*' THEN 1 ELSE 0 END
as aq_flag, CASE WHEN hydricrating = 'Yes' THEN 1 ELSE 0 END as
hyd_flag from component where areasymbol = 'ca113'
) as a
group by a.mukey
) as b
where b.aq_element = 1 or hyd_rated = 1
) as c
join mapunit_poly on mapunit_poly.mukey = c.mukey ;
-- create spatial index
create index yolo_wet_polys_spatial_idx on yolo_wet_polys using gist
(wkb_geometry gist_geometry_ops) ;
-- add unique key
ALTER TABLE yolo_wet_polys add column ogc_fid serial;
CREATE UNIQUE INDEX ogc_fid_idx_yolo_wet_polys on yolo_wet_polys
(ogc_fid);
-- grant access privs
grant select on table yolo_wet_polys to soil ;
VACUUM ANALYSE yolo_wet_polys ;
-- register geometry columns
INSERT INTO geometry_columns values
('','','yolo_wet_polys','wkb_geometry',2,9001,'POLYGON');
```

Location of seasonally wet soils

http://casoilresource.lawr.ucdavis.edu/drupal/node/268

PostGIS Examples: Soil Survey Data

acres_yolo	pct_yolo_area	max_ph
58	0.01	7.4
66	0.01	6.5
107	0.02	7.6
260	0.04	8.7
8494	1.30	7.3
8754	1.34	7.2
20766	3.18	6.1
21772	3.33	7
23301	3.57	6.2
24318	3.72	8.8
34640	5.30	8.5
73391	11.24	6.7
109929	16.83	7.9
139437	21.35	8.2
170697	26.14	7.5

```
select round(sum(area(mapunit_poly.wkb_geometry)) *
0.000247) as acres_yolo, round(
(sum(area(mapunit_poly.wkb_geometry)) / (select
area(yolo_co.wkb_geometry) from yolo_co) *
100)::numeric, 2) as pct_yolo_area, c.max_ph
from mapunit_poly
join
(
select b.mukey, max(b.max_ph) as max_ph from
(
select component.mukey, component.cokey, a.max_ph
from component join (select cokey, max(ph1to1h2o_r) as
max_ph from chorizon where areasymbol = 'ca113' group
by cokey having max(ph1to1h2o_r) is not null
) as a on a.cokey = component.cokey
where component.majcompflag = 'Yes'
) as b group by b.mukey order by b.mukey
) as c on mapunit_poly.mukey = c.mukey
group by c.max_ph order by pct_yolo_area;
```

Area calculation of Yolo pH values

http://casoilresource.lawr.ucdavis.edu/drupal/node/265

Proj4 Examples

Forward and Inverse Projections

* Perform a forward projection to UTM Zone 10 North, NAD83 Datum

  file coordinate_list:

  ```
  --------------------------------
  -121 37
  -121.231231 37.482713
  --------------------------------
  ```

  proj +proj=utm +zone=10 +datum=NAD83 < coordinate_list

  ```
  677962.19 4096742.06
  656384.16 4149892.63
  ```
* Perform an inverse projection from UTM Zone 10 North, NAD83 Datum to Lat/Lon

  file utm_coordinates:

  ```
  --------------------------------
  677962.19 4096742.06
  656384.16 4149892.63
  --------------------------------
  ```

  invproj +proj=utm +zone=10 +datum=NAD83 < utm_coordinates

  ```
  121dW 37dN
  121d13'52.432"W 37d28'57.767"N
  ```
* Optionally print Lat/Lon as decimal degrees:

  invproj -f "%.6f" +proj=utm +zone=10 +datum=NAD83 < utm_coordinates

  ```
  -121.000000 37.000000
  -121.231231 37.482713
  ```

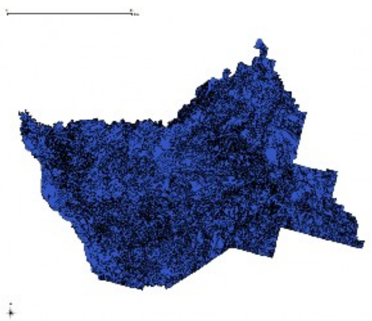
Conversions Between Coordinate Systems

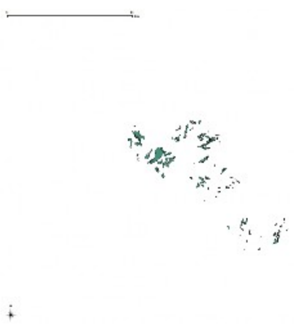
* Datum shift from NAD27 to NAD83

  cs2cs +proj=latlong +datum=NAD27 +to +proj=latlong +datum=NAD83

http://casoilresource.lawr.ucdavis.edu/drupal/node/105

GDAL Examples



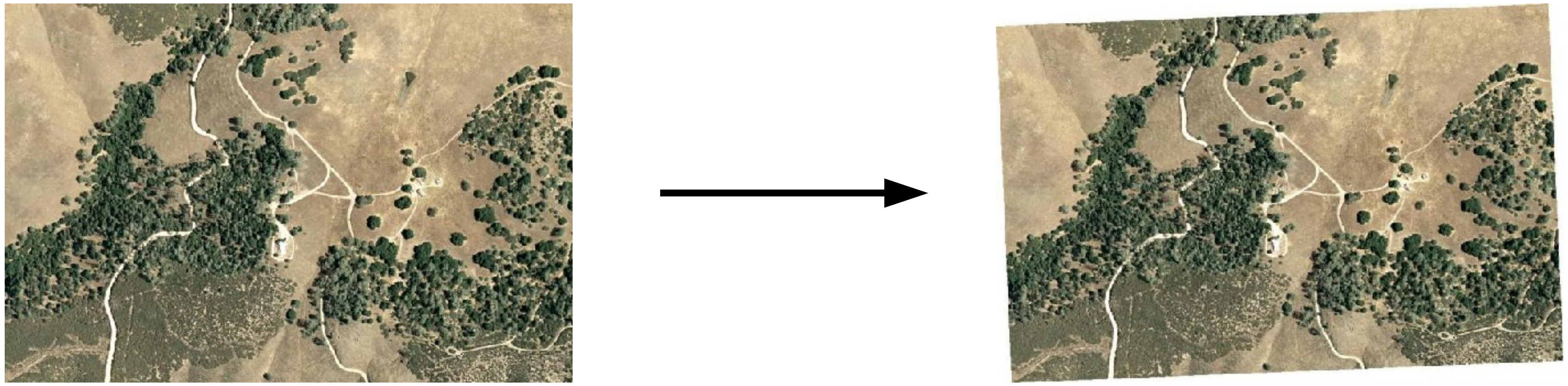


Vector Operations

* Get information about a shapefile:
ogrinfo -al ssurgo_geo.shp | less
* Reproject SSURGO mapunit data from geographic coordinates (Lat/Lon) to UTM zone 10:
ogr2ogr -t_srs "+proj=utm +zone=10 +datum=NAD83" ssurgo_utm.shp ssurgo_geo.shp
* Extract all polygons from a SSURGO shapefile where the mapunit symbol is 'ScA' :
ogr2ogr -where "musym = 'ScA' " ssurgo_ScA.shp ssurgo_utm.shp

Raster Operations

* Get information about a raster dataset
gdalinfo rasterfile.tiff
* Reproject an aerial photo in CA State Plane Zone 4 (Lambert Conformal Conic projection, units = feet) to UTM Zone 10 (units = meters), and rescale to 1 meter output resolution, and use:
gdalwarp -tps -t_srs '+proj=utm +zone=10 +datum=NAD83 +units=m' -s_srs
'+proj=lcc +lat_1=36.0000 +lat_2=37.2500 +lat_0=35.333333333333336 +lon_0=-119.0000
**+x_0=2000000 +y_0=500000 +datum=NAD83 +units=ft' -srcnodata 255 -dstnodata 255 -tr 1 1 **
884084.tif 884084-utm.tif
* Convert Multi-band GeoTiff file to JPEG:
gdal_translate -of JPEG 884084-utm.tif 884084-utm.jpg



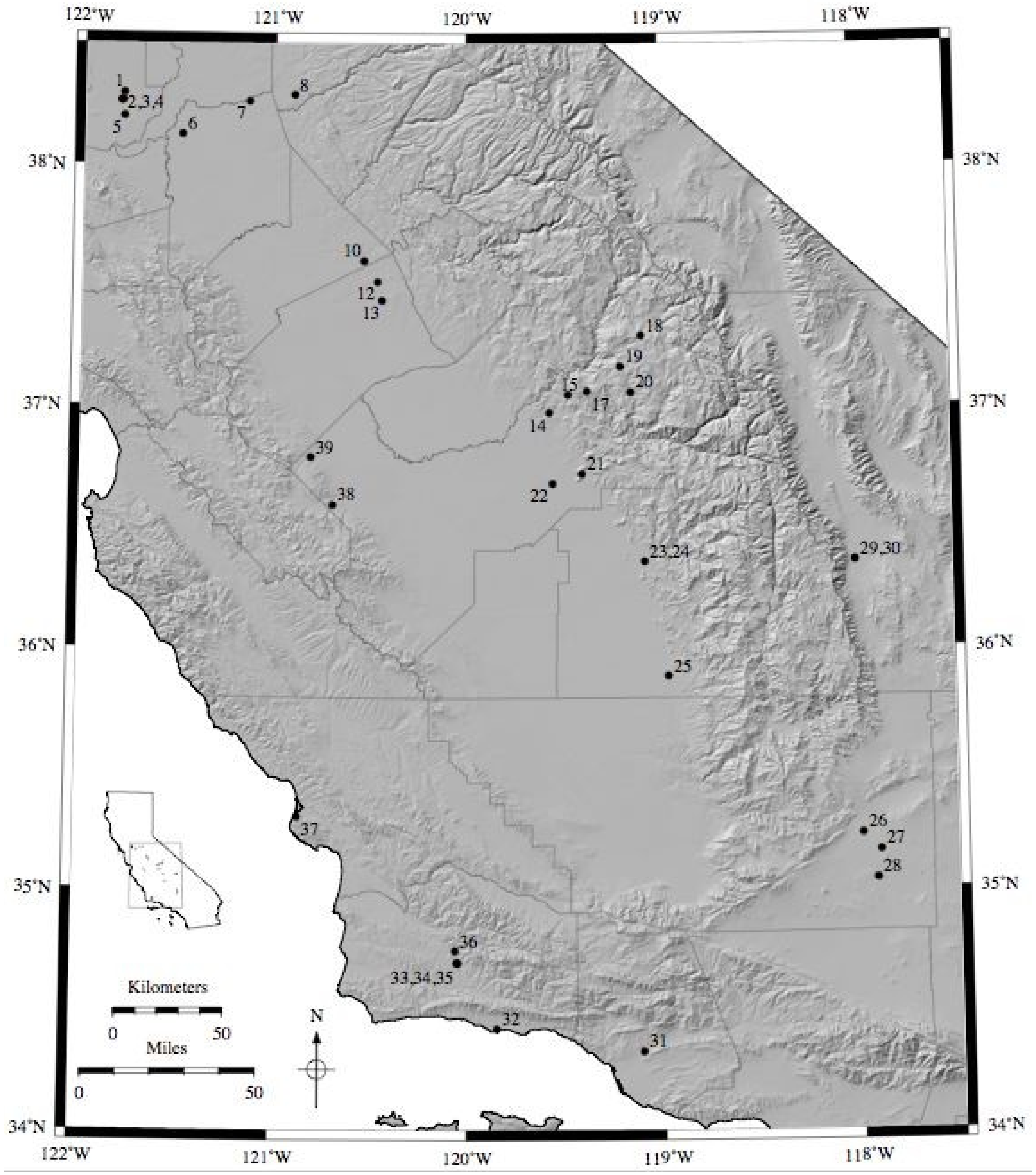
http://casoilresource.lawr.ucdavis.edu/drupal/node/98

GMT Examples

```
* Setup some default parameters for GMT
  gmtset COLOR_NAN 255/255/255 DOTS_PR_INCH 300 ANNOT_FONT_PRIMARY Times-Roman \
  ANNOT_FONT_SIZE_PRIMARY 10 ANNOT_FONT_SIZE_SECONDARY 10 HEADER_FONT Times-Roman \
  LABEL_FONT Times-Roman LABEL_FONT_SIZE 10 HEADER_FONT_SIZE 10 PAPER_MEDIA letter \
  UNIX_TIME_POS 0i/-1i PS_COLOR CMYK Y_AXIS_TYPE ver_text PLOT_DEGREE_FORMAT DF

* Create a basemap, with lat/lon tics
  psbasemap -JB-120/40/34/40.5/7i -R-122/-117.5/34/38.5 -Xc -Yc -B1 -K -P -V > ssc105s.ps

* Plot the landform raster data with a grayscale color table
  makecpt -Cgray -T0/255/1 -V > grey.cpt
  grdimage shade.grd=2 -Cgrey.cpt -J -R -K -O -V >> ssc105s.ps

* Create scale bars in miles and kilometers
  psbasemap -J -R -Lf-121.5/34.5/34/50k:"Kilometers": -T-120.75/34.25/1.5c -O -K -P -V >>ssc105s.ps
  psbasemap -J -R -Lf-121.5/34.25/34/50m:"Miles": -O -K -P -V >> ssc105s.ps

* Plot some vector data: Ca outline, Ca counties, and pit locations
  psxy counties.xy -M -J -R -W1/100/100/100 -P -O -K -V >> ssc105s.ps
  psxy ca.xy -M -J -R -W3/0/0/0 -P -O -K -V >> ssc105s.ps
  psxy pit_locations.xy -J -R -Sc0.15c -W1/0/0/0 -G0/0/0 -P -O -K -V >> ssc105s.ps

* Label the pit locations
  pstext pit_locations.labels -J -R -Dj0.1c/0.1c -G0/0/0 -P -O -K -V >> ssc105s.ps

* Make a small context map showing the entire state
  psxy ca.xy -M -JB-120/40/34/40.5/1i -R-124.5/-114/32/42.5 -X0.25i -Y1.5i -W2/0/0/0 -P -O -K -V >> ssc105s.ps
  psxy pit_locations.xy -J -R -Sc0.01c -W1/0/0/0 -G0/0/0 -P -O -K -V >> ssc105s.ps
  psxy zoom_extent.xy -M -JB -R -W1/0/0/0 -P -O -V >> ssc105s.ps
```

http://casoilresource.lawr.ucdavis.edu/drupal/node/102

where2.0



O'REILLY

Links

Details / Source for this talk / notes

http://casoilresource.lawr.ucdavis.edu/drupal/node/95

http://casoilresource.lawr.ucdavis.edu/drupal/node/244