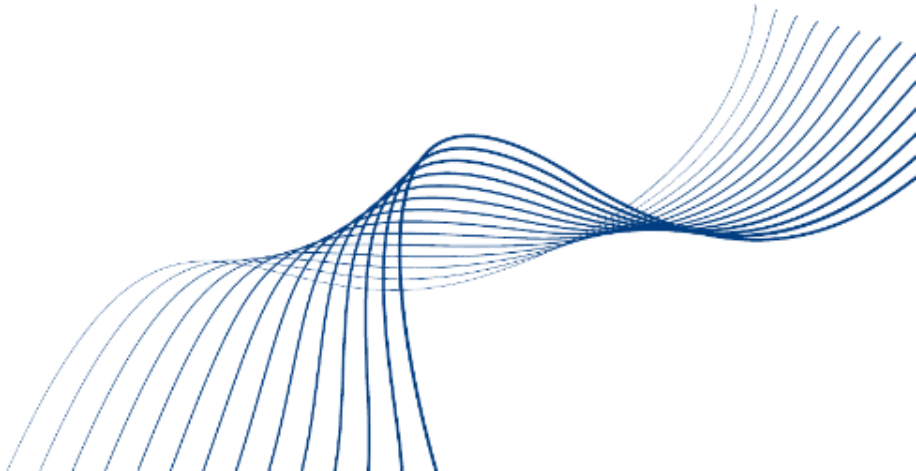
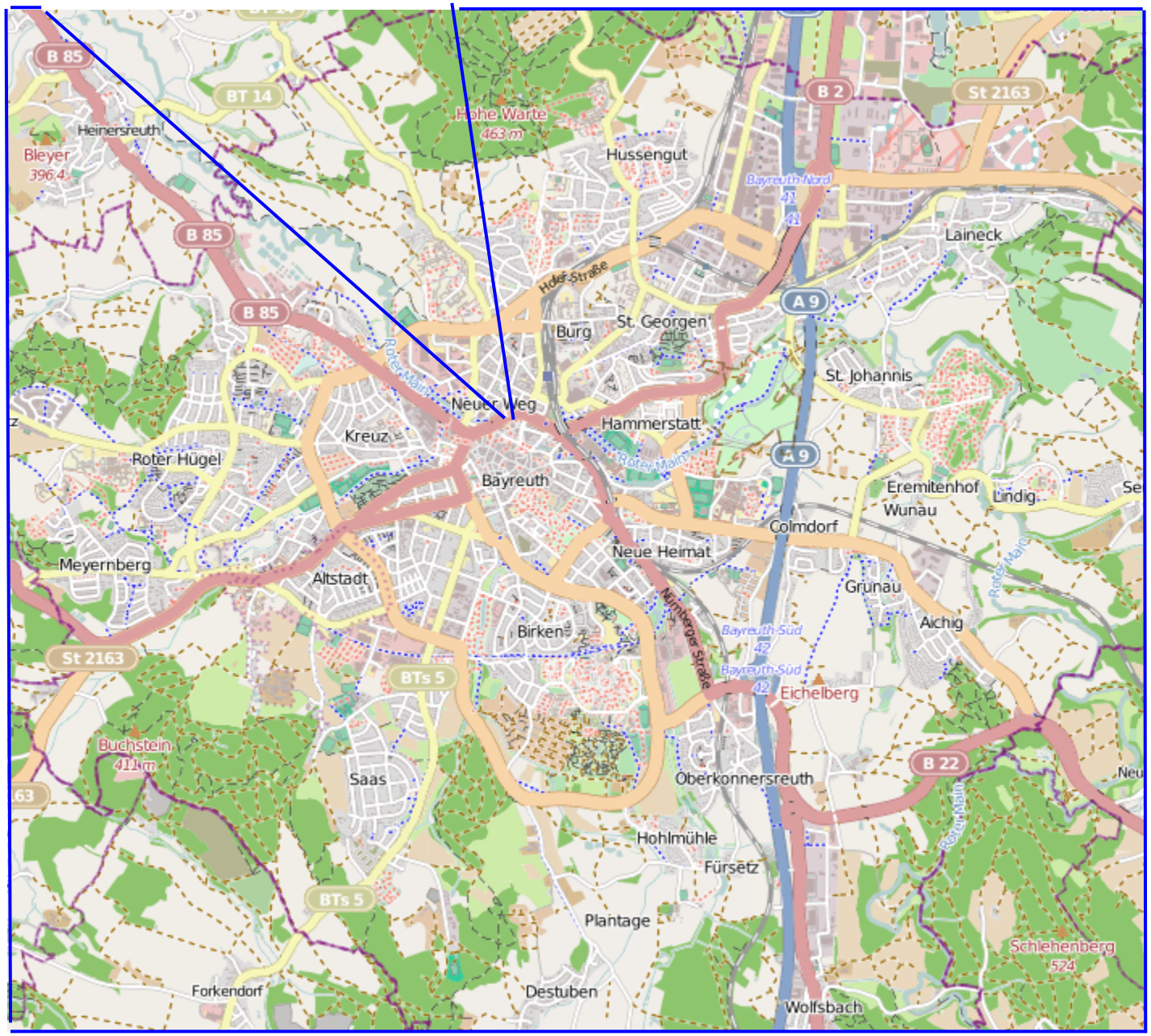
GP = GP

Maenner die auf Pixel starren



http://gigapixel-bayreuth.de


B 85
BT 14
Heinersreuth
Bleyer
396.4
Hohe Warte
463 m
Hussengut
B 2
St 2163
Bayreuth Nord
41
Laineck
B 85
Hohe Straße
A 9
B 85
Burg
St. Georgen
Roter Main
St. Johannis
Neuer Weg
Hammerstatt
Kreuz
Roter Hügel
Bayreuth
Roter Main
A 9
Eremitenhof
Lindig
Wunau
Colmdorf
Neue Heimat
Meyernberg
Altstadt
Grunau
Nürnberger Straße
Aichig
Birken
Bayreuth Süd
42
St 2163
BTs 5
Bayreuth Süd
42
Eichelberg
Buchstein
411 m
Saas
Oberkonnersreuth
B 22
Hohlmühle
Fürsetz
BTs 5
Plantage
Schiehenberg
524
Forkendorf
Destuben
Wolfsbach


B 85
BT 14
Heinersreuth
Bleyer
396.4
Hohe Warte
463 m
Hussengut
B 2
St 2163
Bayreuth Nord
41
Laineck
B 85
Hohe Straße
A 9
B 85
Burg
St. Georgen
Roter Main
St. Johannis
Neuer Weg
Hammerstatt
Kreuz
Roter Hügel
Bayreuth
A 9
Eremitenhof
Lindig
Wunau
Colmdorf
Neue Heimat
Meyernberg
Altstadt
Grunau
Nürnberger Straße
Aichig
Birken
Bayreuth Süd
42
St 2163
BTs 5
Eichelberg
Buchstein
411 m
B 22
Saas
Oberkonnersreuth
Hohlmühle
Fürsetz
BTs 5
Plantage
Schiehenberg
524
Forkendorf
Destuben
Wolfsbach

4 GP, 135mm Objektiv, ~500 Einzelbilder

62 GP, 300mm Objektiv, ~1500 Einzelbilder

Ueberlappung

Giga

Rendern

VM in vSphere Cluster

VM in vSphere Cluster

4 Kerne 16 GB RAM

40 GB HDD

VM in vSphere Cluster

8 Kerne 16 GB RAM

40 GB HDD

VM in vSphere Cluster

8 Kerne 24 GB RAM

40 GB HDD

VM in vSphere Cluster

8 Kerne 32 GB RAM

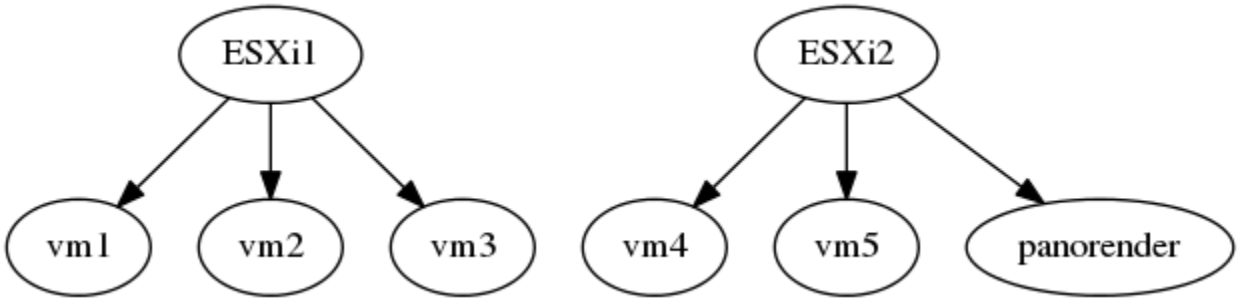
40 GB HDD

VM in vSphere Cluster

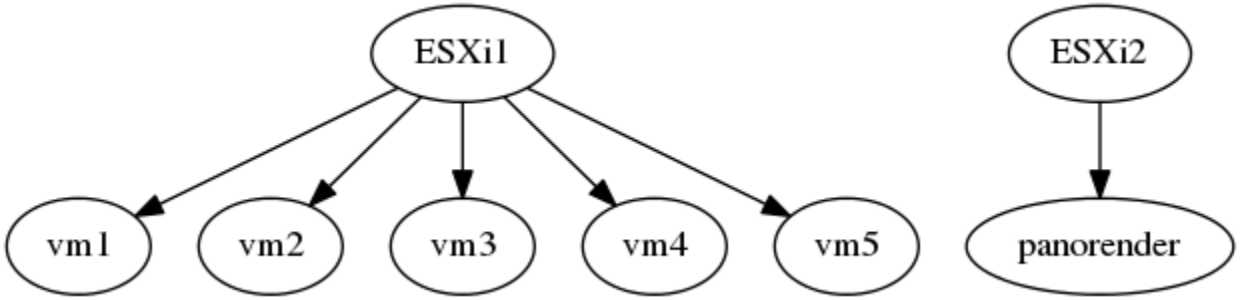
8 Kerne 32 GB RAM

150 GB HDD

Vorher



Nachher



Balancing mit vMotion funktioniert!

Fuck...wir brauchen Blech

8 Kerne 64 GB RAM

10 TB HDD nfs4

10 TB /tmp nfs4

JPEG

PNG

TIFF

PSD

KRO

Format KRO

KRO Format specification

The KRO format is a simple image file storage format.

```
Header is 20 bytes long :
3 bytes : "KRO" signature in hex 0x4B 0x52 0x4F
1 byte  : 0x01 version
unsigned long : Width
unsigned long : Height
unsigned long : depth = > 8 bits, 16 bits, 32 bits
unsigned long : ncomp => number of compoment, 4 by default, RGB + Alpha
Data * : the data uncompressed
```

- Everything is stored in big endian form (header and datas)
- the data is stored RGBA, RGBA (or RGB, RGB if you have 3 component only)
- It's row major order, meaning, you store first row, then second row
- top row is the first row
- Within a row, are columns stored left-to-right
- 16bits format is a 16bits integer (0..65535), not floating point (it's not like half floating point in openEXR)
- 32bits format is pure 'c' float stored binary in big endian format (IEEE 754 binary floating point)
- no color space specification nor metadata stored
- file size limit is ruled by file system

XNview[1] can open and write .KRO files. KRPano Tools [2] are also able to open .KRO files to generate tiles

http://www.autopano.net/wiki-en/Format_KRO

```
header = struct.unpack_from('>3sb4L',fh.read(20))
```

```
jan@shae [01:34:21] [~/development/scm.
-> % python2 info.py ~/pano_small.kro
('KRO', 1, 91076, 20707, 8, 4)
```

```
def _getNeighborPixel(self,n):
    self._gotoPixel(n)
    p = [[False]*3]*3

    a,b,c,d = 0,3,0,3
    if int(n/self.width) != 0:
        self.fh.seek(-(self.width*self.ncomp-self.ncomp),1)
    else:
        self.fh.seek(-self.ncomp,1)
        a += 1
    if int(n/self.width) == self.height: b -= 1

    if n%self.width == 0: c += 1
    elif n%(self.width-1) == 0: d -= 1

    for i in range(a,b):
        for j in range(c,d):
            p[i][j] = self.fh.read(self.ncomp)
        self.fh.seek(self.width*self.ncomp-self.ncomp*d,1)
    return p

def _newColor(self,n):
    p = self._getNeighborPixel(n)

    usable_neighbors = reduce(
        list.__add__,
        [filter(lambda x: x[3] == 0,i) for i in str],
        []
    )

    if len(usable_neighbors) < 2:
            return (0,0,0,0)

    r,g,b,a = reduce(
        lambda x,y: [(x[i]+y[i])/usable_neighbors for i in range(len(x))]
        ,usable_neighbors
    )
    return (r,g,b,255)
```

```
        self.memory = deque(maxlen=self.width*self.nmem) # keeps 5 lines in memory to seek
        self.currPixAbsolut = int(((self.width*self.height)/4)*3)
        self.currPix = 0 # with respect to memory
        self.fh.seek(temp)

    def _populateMemory(self):
        self.fh.seek(20+self.currPixAbsolut*self.ncomp)
        for i in range(0,self.width*self.nmem):
            self.memory.append(self._read())
```

```
        self.memory = deque(maxlen=self.width*self.nmem) # keeps 5 lines in memory to seek
        self.currPixAbsolut = int(((self.width*self.height)/4)*3)
        self.currPix = 0 # with respect to memory
        self.fh.seek(temp)

    def _populateMemory(self):
        self.fh.seek(20+self.currPixAbsolut*self.ncomp)
        for i in range(0,self.width*self.nmem):
            self.memory.append(self._read())



    def _getNeighbors(self):
        return [
            [self.memory[self.currPix-self.width-1]
            ,self.memory[self.currPix-self.width]
            ,self.memory[self.currPix-self.width+1]
            ],[
            self.memory[self.currPix-1]
            ,self.memory[self.currPix]
            ,self.memory[self.currPix+1]
            ],[
            self.memory[self.currPix+self.width-1]
            ,self.memory[self.currPix+self.width]
            ,self.memory[self.currPix+self.width+1]
            ]
        ]
```

```python
import sys
import struct

# Usage: crop.py <dh> <dw> <x> <y> <filetocrop>
# All Input is relativ to width/height

with open(sys.argv[5]) as fh:
    header= struct.unpack_from('>3sb4L',fh.read(20))

    if header[0] != 'KRO':
        exit()
    width, height, depth, ncomp = header[2:]
    size_index = depth*ncomp/8
    img_format = 'RGBA'[:ncomp]

    fh.seek(int(width*int(height*float(sys.argv[1])))*ncomp,1)
    fh.seek(int(width*float(sys.argv[2]))*ncomp,1)
    w,h = int(width*float(sys.argv[3])),int(height*float(sys.argv[4]))
    #print >> sys.stderr, w,h,int(width*int(height*float(sys.argv[1]))),int
    sys.stdout.write(struct.pack('>3sb4L','KRO',1,w,h,8,4))
    for i in range(0,h):
        sys.stdout.write(fh.read(w*ncomp))
        fh.seek((width-w)*ncomp,1)
~
```

Legacy Image Pyramids
IIF
DZI
OSM
Custom Tile Sources

http://gigapixel-bayreuth.de

gp@hoeja.de
@nv1t