High Dimensional Images and Low Dimensional Representations

Thomas Breuel

UniKL

Head Rotation

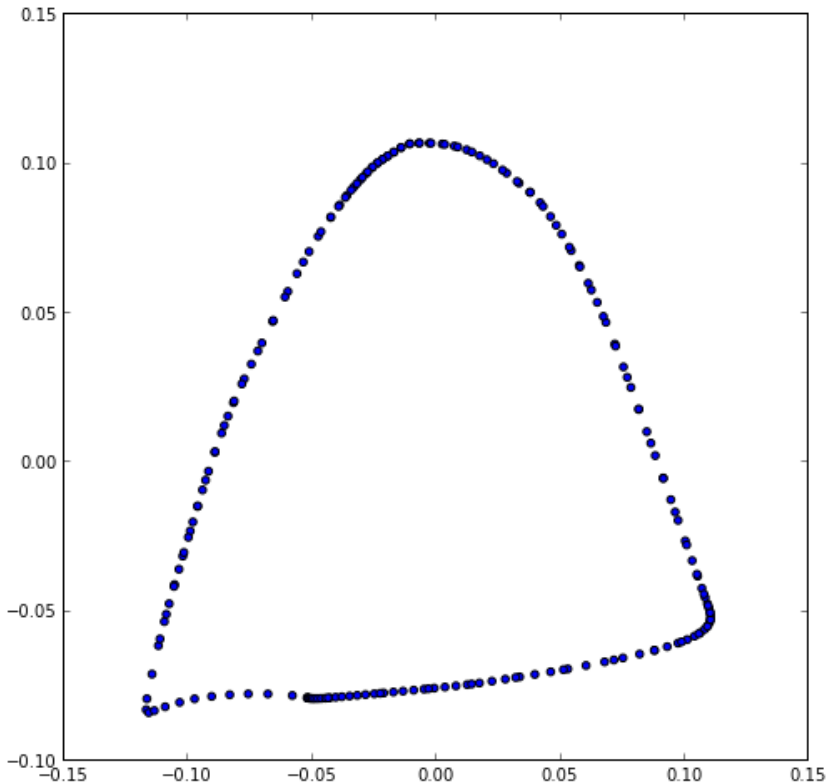
```
frames = cv2.VideoCapture("headrot.webm")
print frames.get(cv2.CAP_PROP_FRAME_COUNT)
data = []
for i in range(200):
    _,frame = frames.read()
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    data.append(gray.ravel())
data = array(data)
del frames
```

```
251.0
```

```
from sklearn.decomposition import RandomizedPCA
pca = RandomizedPCA(10)
lo = pca.fit_transform(data)
print lo.shape
print lo[0]
```

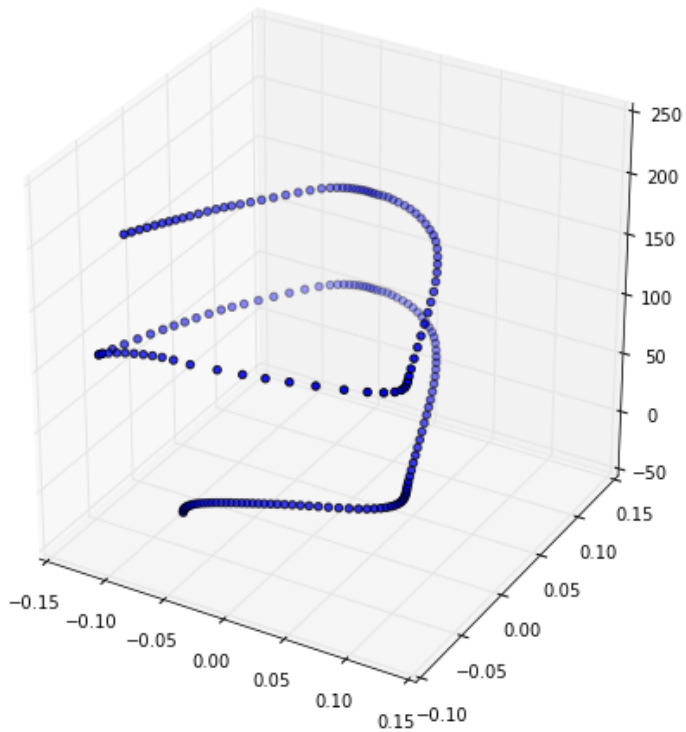
```
(200, 10)
[-4958.87141816  1066.76497348   526.50462093  2966.77444885  1029.42127905
  -799.54184935  -254.05764835  -456.85720198  -988.66067974    52.12461804]
```

```
1 mds = manifold.LocallyLinearEmbedding()
2 vl = mds.fit_transform(lo)
3 scatter(vl[:,0],vl[:,1])
```



```
1 from mpl_toolkits.mplot3d import Axes3D
2 gcf().add_subplot(111, projection='3d')
3 gca().scatter(vl[:,0], vl[:,1],arange(len(vl)))
```

```
<mpl_toolkits.mplot3d.art3d.Patch3DCollection at 0x5954110>
```



Recovering Low Dimensional Structure of Views of Real Objects

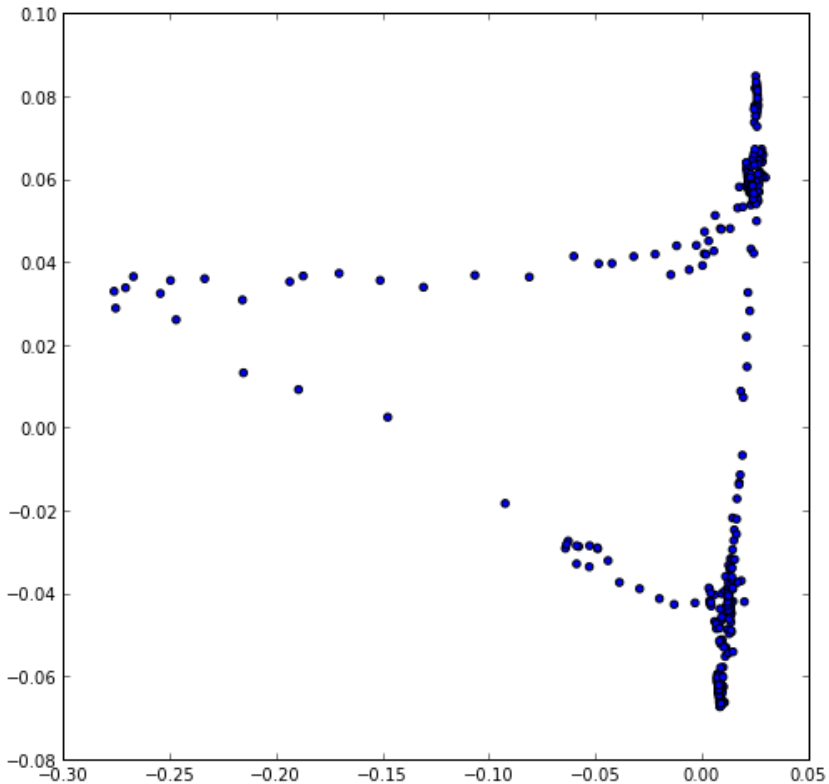
```
frames = cv2.VideoCapture("bunny.mp4")
print frames.get(cv2.CAP_PROP_FRAME_COUNT)
data = []
for i in range(350):
    _,frame = frames.read()
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    gray = gray[::2,::2]
    data.append(gray.ravel())
data = array(data)
del frames
```

```
359.0
```

```
from sklearn.decomposition import RandomizedPCA
pca = RandomizedPCA(20)
lo = pca.fit_transform(data)
print lo.shape
print lo[0]
```

```
(350, 20)
[ 3853.40952636  2180.11119734  2171.40068491  -206.92715815   215.6530972
  2157.56203323   316.11037864 -2269.19097953 -1318.46861416    35.84289751
  -594.64097527  1022.32363534   -29.46706416   308.2711417    229.60280484
  -712.37283502  -170.98457044  1484.92789905 -1202.73114523   160.54615569]
```

```
1 mds = manifold.LocallyLinearEmbedding()
2 vl = mds.fit_transform(lo)
3 scatter(vl[:,0],vl[:,1])
```



```
1 from mpl_toolkits.mplot3d import Axes3D
2 gcf().add_subplot(111, projection='3d')
3 gca().scatter(vl[:,0], vl[:,1],arange(len(vl)))
```

```
<mpl_toolkits.mplot3d.art3d.Patch3DCollection at 0x84dad50>
```

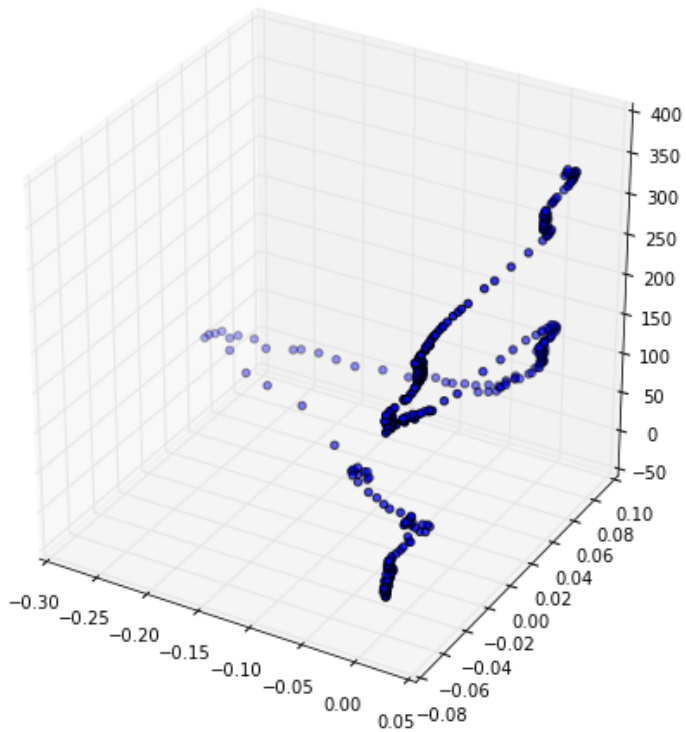


Image Normalization and Intrinsic Dimensions

Dimensionality for Real World Images

can compensate by 2D image processing:

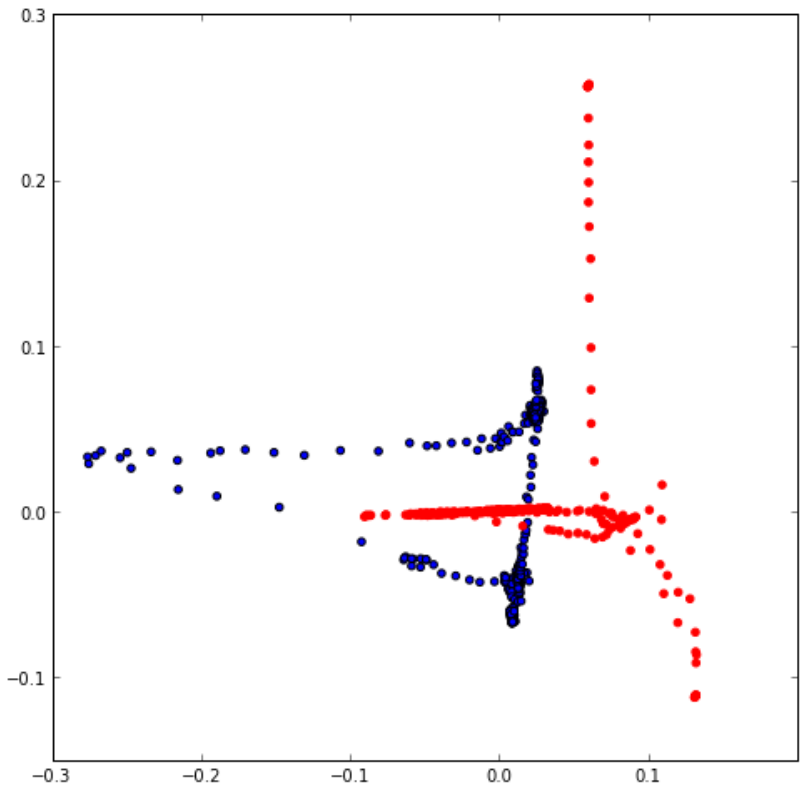
- 2D translations (2 dimensions)
- 2D rotations (1 dimension)

cannot compensate by 2D image processing:

- translation in depth / scaling (1 dimension)
- 3D rotations (2 dimension)

```
from scipy.ndimage import measurements,interpolation
frames = cv2.VideoCapture("bunny.mp4")
data = []
for i in range(350):
    _,frame = frames.read()
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    gray = gray[::2,::2]
    y,x = measurements.center_of_mass(gray<0.3*mean(gray))
    gray = interpolation.affine_transform(gray,diag((1,1)),offset=(
        y-120,x-180),mode='mirror')
    data.append(gray.ravel())
data = array(data)
del frames
```

```
from sklearn.decomposition import RandomizedPCA
pca = RandomizedPCA(20)
lo = pca.fit_transform(data)
mds = manifold.LocallyLinearEmbedding()
vl2 = mds.fit_transform(lo)
scatter(vl[:,0],vl[:,1])
scatter(vl2[:,0],vl2[:,1],color='red')
```



View "Manifolds"

The appearance of a rigid 3D object (including all the pixel values) is a function of the six pose parameters.

The appearance is usually piecewise smooth.

If it were completely smooth, it would be a *manifold*.