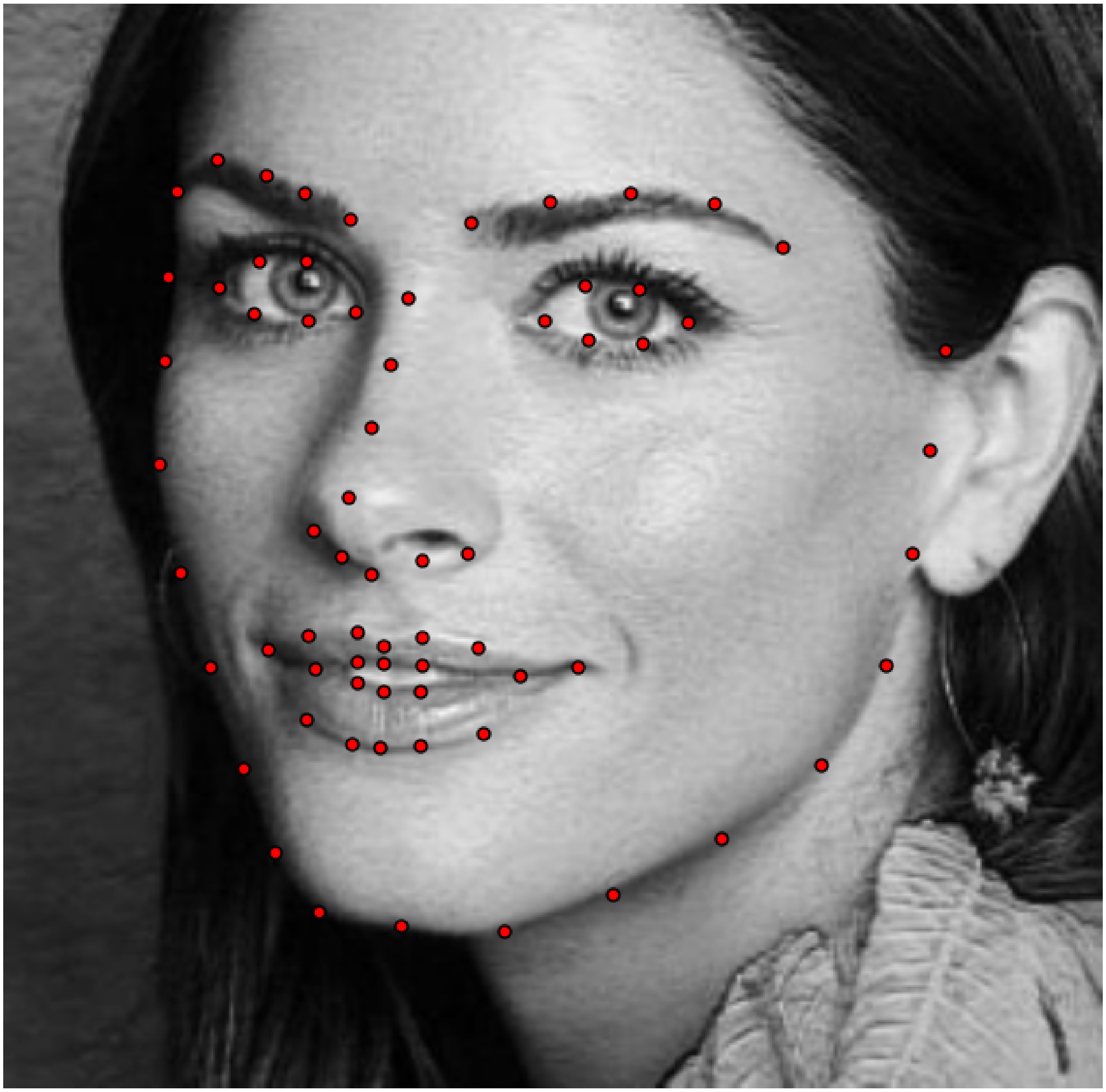
In [1]:

```
#import libraries needed from menpo project
%matplotlib inline
import menpo.io as mio
from menpo.visualize import print_progress
from menpo.landmark import labeller, face_ibug_68_to_face_ibug_68_trimesh
from menpowidgets import visualize_images
from pathlib import Path

#here we have stored all the images from training
path_to_images = 'C:\\Users\\ch9fod\\Documents\\lfpw\\trainset'
training_images = []
for img in print_progress(mio.import_images(path_to_images, verbose=True)):
    # convert to greyscale
    if img.n_channels == 3:
        img = img.as_greyscale()
    # crop to landmarks bounding box with an extra 20% padding
    img = img.crop_to_landmarks_proportion(0.2)
    # rescale image if its diagonal is bigger than 400 pixels
    d = img.diagonal()
    if d > 400:
        img = img.rescale(400.0 / d)
    # define a TriMesh which will be useful for Piecewise Affine Warp of Holistic
    labeller(img, 'PTS', face_ibug_68_to_face_ibug_68_trimesh)
    # append to list
    training_images.append(img)
```

```
Found 811 assets, index the returned LazyList to import.
[====================] 100% (811/811) - done.
```

In [2]:

```
%matplotlib inline
from menpowidgets import visualize_images
visualize_images(training_images)
```

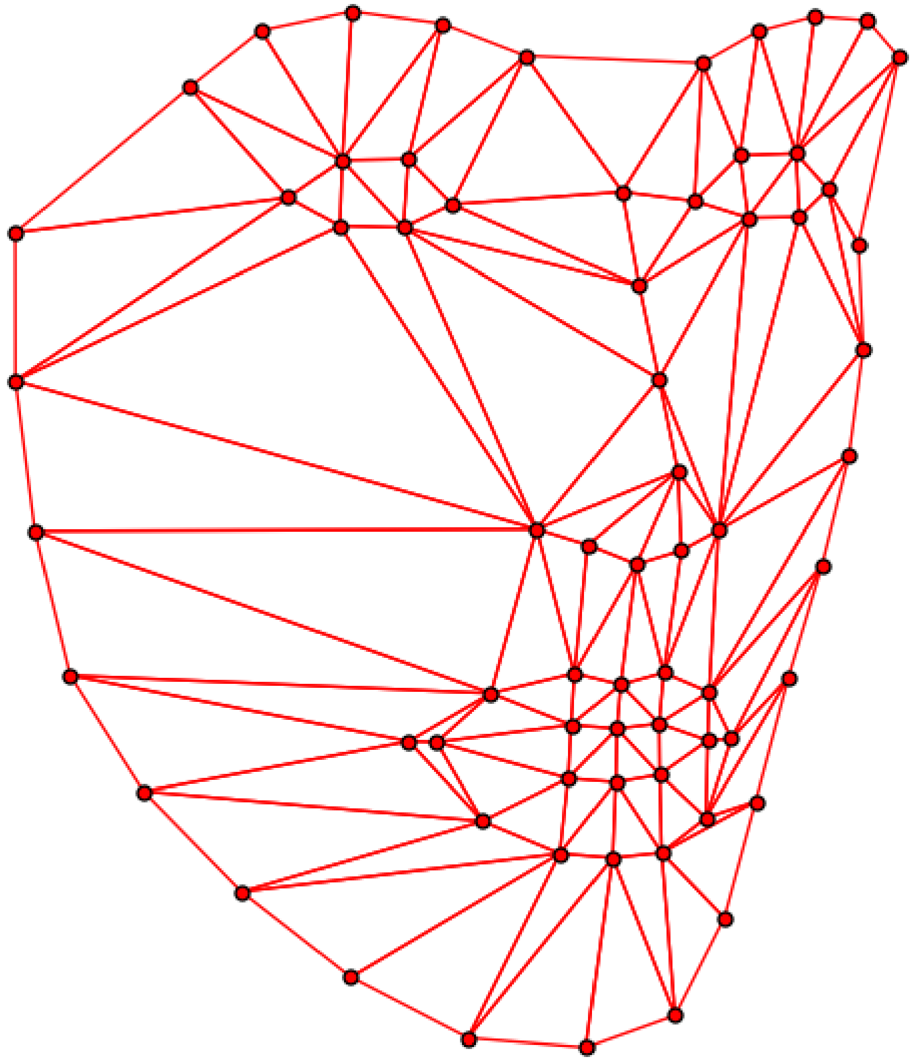


In [2]:

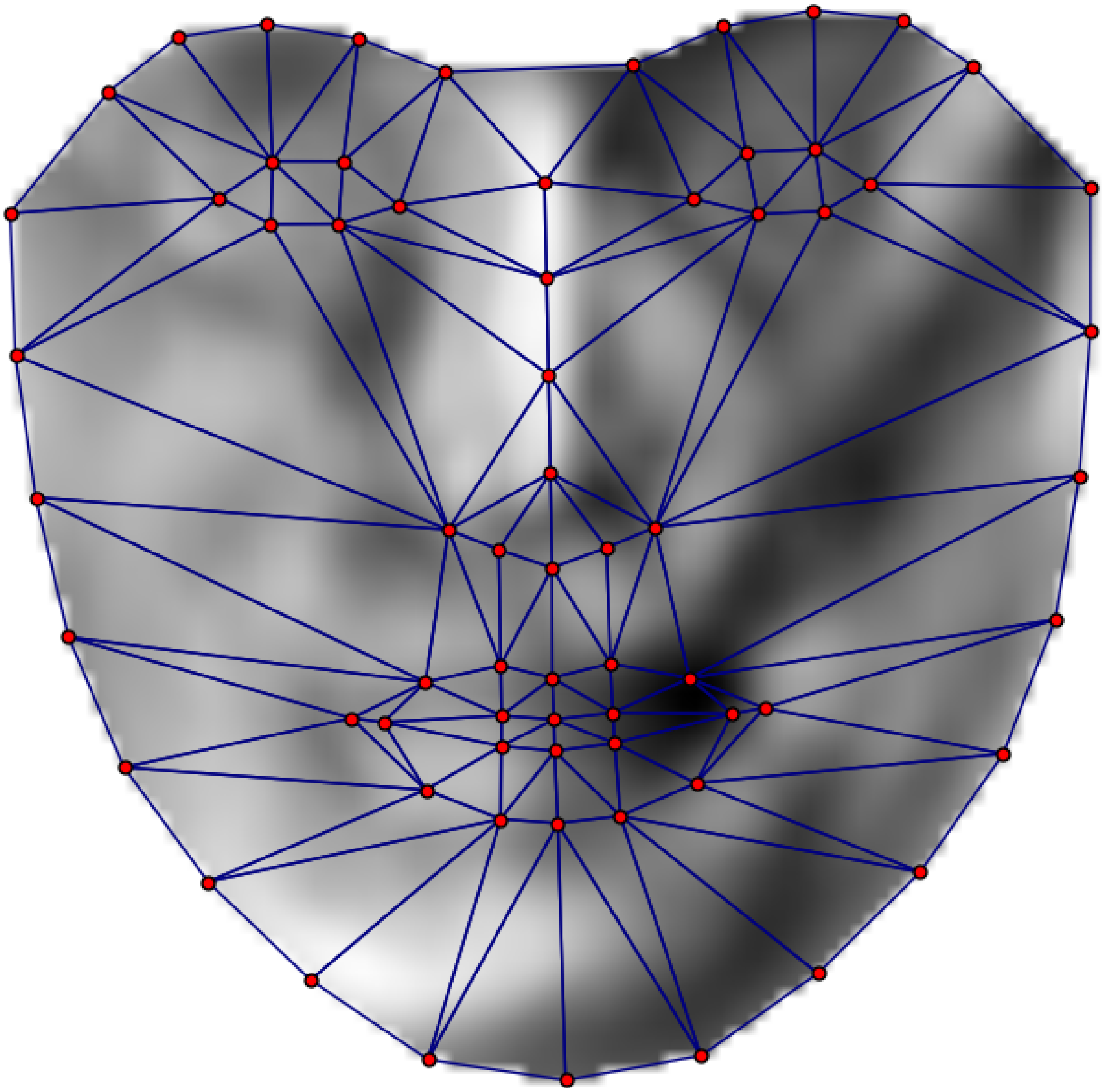
```
from menpofit.aam import HolisticAAM
from menpo.feature import fast_dsift

# build Holistic AAM
aam = HolisticAAM(
    training_images,
    group='face_ibug_68_trimesh',
    verbose=True,
    holistic_features=fast_dsift,
    diagonal=120,
    scales=(0.5, 1.0)
)
```

```
- Computing reference shape
 Computing batch 0
- Building models
  - Scale 0: Done
  - Scale 1: Done
```

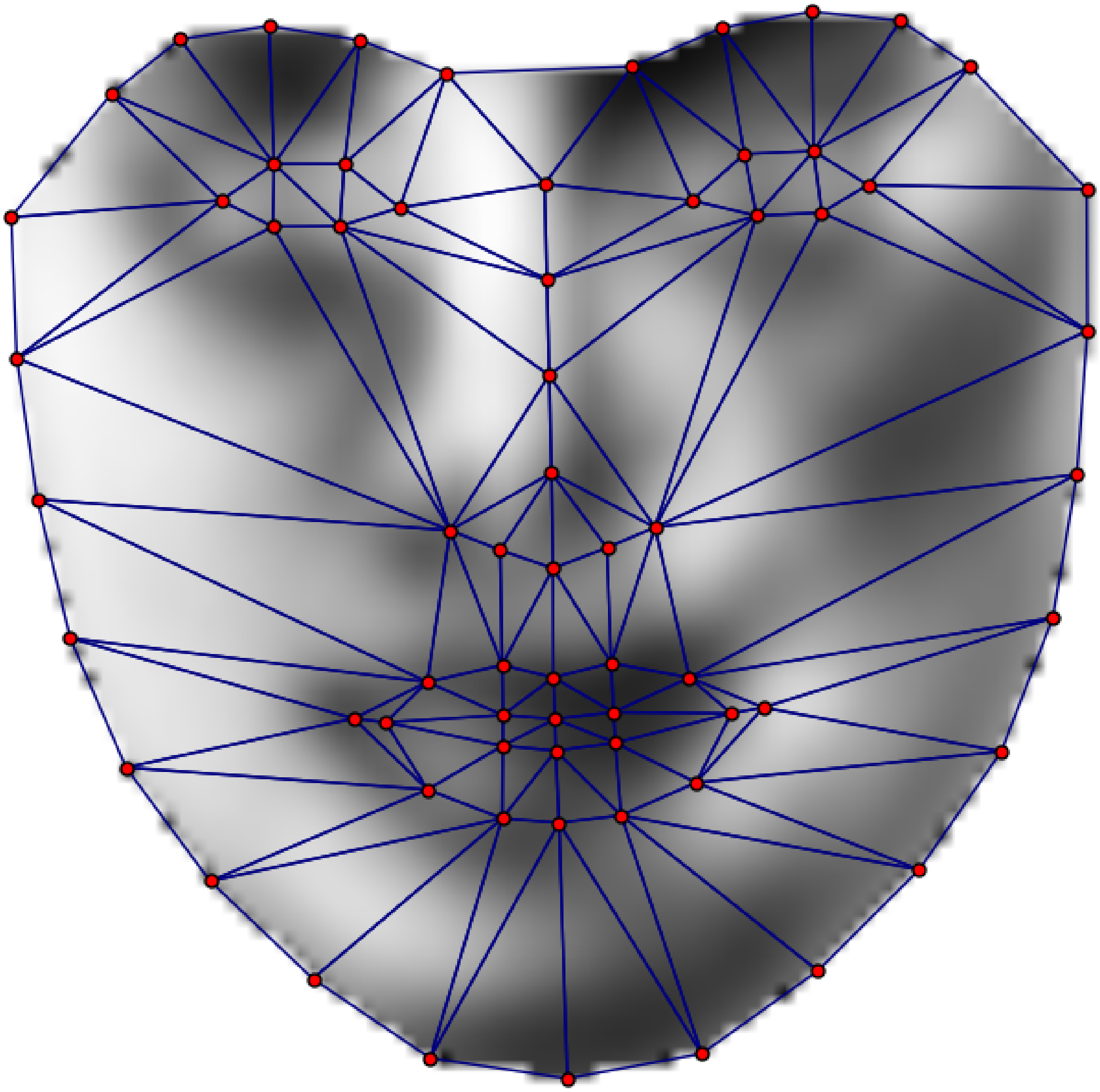
In [3]:
```
#with this widget we can view the shapes model
aam.view_shape_models_widget()
```



```
In [4]: #with this widget we can view the appearance model
        aam.view_appearance_models_widget()
```



In [5]:
```
#with this widget we can view both parameters
aam.view_aam_widget()
```



In [6]:
```
#I use a LucasKanade fitter that uses the previous model created
from menpofit.aam import LucasKanadeAAMFitter

fitter = LucasKanadeAAMFitter(aam, n_shape=0.9, n_appearance=0.9)
```

In [7]:

```
path_to_lfpw = Path('C:\\Users\\ch9fod\\Documents\\lfpw\\')

# load test
test_images = []
bboxes = []
#for i in mio.import_images(path_to_lfpw / 'myset', max_images=12, verbose=True):
for i in mio.import_images(path_to_lfpw / 'myset', verbose=True):
    # crop image
    i = i.crop_to_landmarks_proportion(0.2)
    # convert it to grayscale if needed
    if i.n_channels == 3:
        i = i.as_greyscale(mode='luminosity')
    # append it to the list
    test_images.append(i)
```

```
Found 32 assets, index the returned LazyList to import.
```

In [8]:

```
from menpofit.fitter import noisy_shape_from_bounding_box

fitting_results = []

# fit images
for i in test_images:
    # obtain ground truth (original) landmarks
    gt_s = i.landmarks['LJSON'].lms

    # generate initialization shape
    initial_s = noisy_shape_from_bounding_box(gt_s, gt_s.bounding_box())

    # fit image
    fr = fitter.fit_from_shape(i, initial_s, gt_shape=gt_s)
    fitting_results.append(fr)

    # print fitting error
    #print(fr)
```

In [9]:

```
#here we view the fitting results
from menpowidgets import visualize_fitting_result

visualize_fitting_result(fitting_results)
```