Processing Energy-Resolved X-Ray Spectroscopy Images
PETRA III, Beamline P64

Severin Angerpointner

Technische Universität München,
Ludwig-Maximilians-Universität München
Germany

September 4, 2018

Abstract

We develop algorithms to automatically correct and process two-dimensional energy resolved X-Ray spectroscopy images. They are used to analyze images generated by a Von Hamos type spectrometer, but in principle apply to a wide range of experimental setups which require calibration, ROI finding or image correction techniques. We also explore machine learning methods to fully automatize the process. In particular, boosted cascade pattern detectors offer a fast and widely reliable alternative to the previous methods.

Contents

1 Introduction

Beamline P64 at PETRA III features a Von Hamos type spectrometer to measure X-Ray fluorescence spectra at high energy resolution. Due to the geometry of its crystals, X-Rays emitted from the sample get focused in one direction and dispersed in another, and projected onto a two-dimensional photon detector via Bragg reflection. A more detailed description of the spectrometer and its properties can be found in [1].
The obtained spectra can eventually be used to study electronic or spin structure as well as to obtain high energy resolved fluorescence detected XANES spectra. In the following report, we will solely be concerned with the extraction of the X-Ray fluorescence spectra from the measured images and explore methods to automate this process.

2 General Procedure

In order to obtain the X-Ray energy spectrum from the detector image, we have to determine the energy-position relation first, i.e. we have to calibrate the detector. By using at least two different incident X-Ray energies well away from any fluorescent transitions, we obtain two focused spots on the detector, with the position of each spot corresponding to the incident energy. With those pairs of energy-position points, the mapping from the horizontal position on the detector to an energy scale can be easily determined and the actual measurement of a fluorescence spectrum can be analyzed. Fig. 1 shows a typical example of a fluorescence spectrum and two elastic line features as well as the final projection (averaged) of the spectrum to the x-axis.

However, the spectrometer at hand is able to project multiple copies of the spectrum onto the detector simultaneously in order to collect as much of the incoming X-Ray fluorescence as possible. This also means that each projection on the detector has to be calibrated separately, since slight changes in the positioning of the spectrometer crystals prior to each experiment will generally also change the exact position and orientation of the spectra on the detector. To treat each spectrum separately, we can define regions of interest (ROIs) around them and then perform the calibration within each region. After obtaining the correct energy-pixel mapping for each ROI, we can average them to get the final spectrum of the image.

Before the discussion of how to implement these steps using automation whenever possible, we have to start by preparing the image itself and correct potential errors, which would influence the outcome of the measurement.

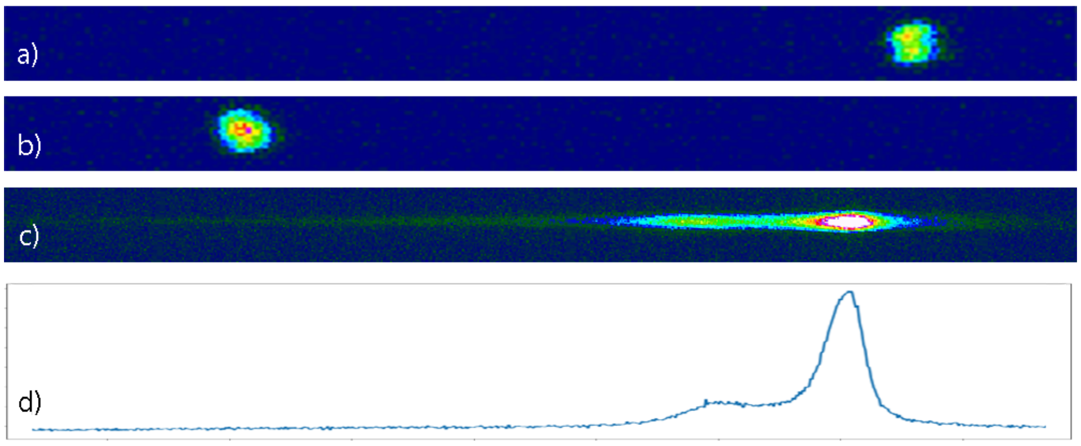

Figure 1: Example images of elastic line features used for calibration (a,b) and a fluorescence spectrum (c) with its averaged projection to the horizontal axis (d). The features are not drawn to scale, as the elastic lines are usually weaker and smaller in size than the fluorescent spectra.

3 Image Corrections

3.1 Common Methods

There are several standard procedures in image processing to correct unavoidable errors. The ones we apply here are *dark field*, *flat field* and I_0 corrections. We will not go into more detail about those, since their implementation is straightforward.

3.2 Bad Pixels

More difficult errors to correct are *bad pixels* – pixels with unphysical intensities that cannot be caused by actually detected photons. They appear as bright spots on the image in different shapes and sizes and intensities. They often appear or disappear from image to image, even within one experiment. Their description as “bright spots”, however, already suggests a way to identify and remove them reliably.

A meaningful quantity to decide whether a pixel is good or bad is the gradient in the direction of its neighbors. A very high gradient immediately hints at bad pixels, since it implies both the high intensity as well as the locality of the bright spots on the image. A rather simple algorithm to scan a whole image for bad pixels is described in Fig. 2

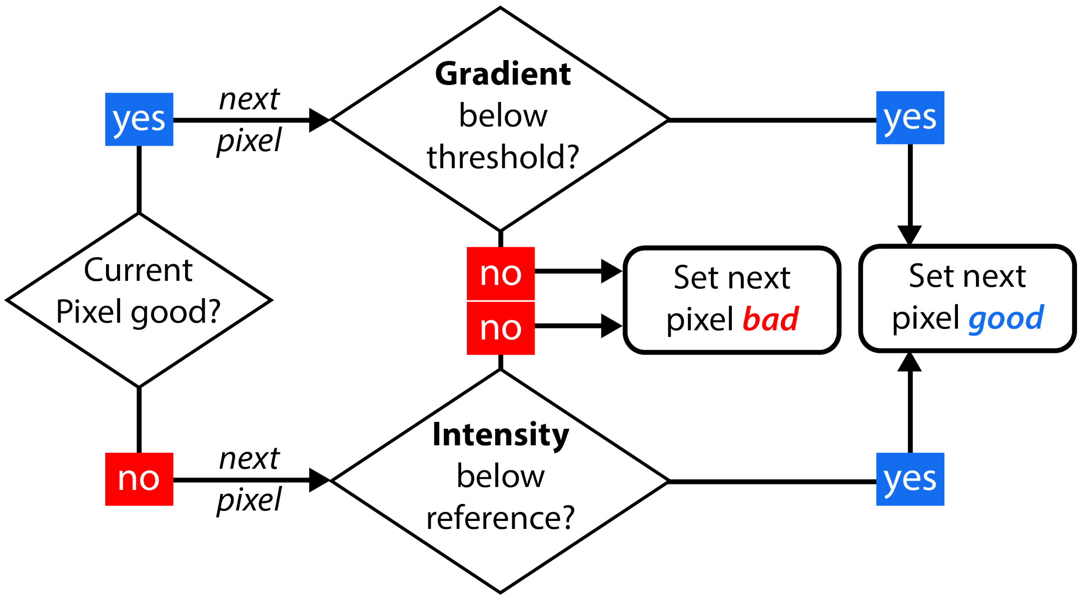

Figure 2: An algorithm to spot bad pixel in the detector image. We start by initializing the first pixel as good and then follow this decision tree iteratively for ever pixel in the image.

After initializing the very first pixel as good,[1] we iterate row by row through the image and check for each pixel the gradient to its neighbor. To reduce computation time, we choose a single direction along which we calculate the gradient and scan the image. By choosing *left to right*, we can exploit the horizontal extension of the spectra, i.e. their smoothness/ low gradient in that direction (see Fig. 1), and thus reduce the risk of marking parts of the actual spectrum as bad. Once we find a gradient above the threshold, we mark all subsequent pixels as bad until the intensity drops back to the level encountered before the bad pixels.
In order to generalize this method to work for most images without the need for manual adjustments, the gradient threshold has to be determined accordingly. Tests showed that choosing the threshold proportional to the intensity of the current pixel produce satisfying results on a wide range of experiments while further reducing the risk of false positives within the spectra. The proportionality factor remains a tunable parameter, but needs to be changed only very rarely.

[1]This choice is arbitrary. Setting it to good will make it hard to spot bad pixels in the first few pixels of the image. Setting it to bad will result in very few bad pixels in the corner of the image, regardless of their actual intensity. But this region should not contribute to the measurement in any measurement setup.

4 Data Extraction

After correcting (most) errors in the images, we can continue with calibrating and extracting the spectra. As mentioned in Section 2, the first step is actually to find suitable ROIs which can then be calibrated separately. The ROIs can in principle be obtained using any image of an experiment (calibration or fluorescence), but it turns out that fluorescence images are generally better suited for the task. This is due to the larger extension of the fluorescence spectra compared to the elastic reflection spots and due to potential angular deviations which are impossible to detect with only one calibration image containing small elastic line features.

4.1 Regions of Interest

Due to the horizontal alignment of the spectra on the detector with only small angular deviations, finding the ROIs effectively boils down to partitioning the image along the y-axis. Given n, the correct number of ROIs, we can rater easily find them by projecting the image onto the y-axis and choosing some range around the n most prominent peaks in intensity. By converting the image to black/white first (every pixel below the maximum noise level[2] is mapped to 0, all others to 1), we can enhance even very faint features in the image. Fig. 3 shows an example image and its projection to the y-axis after converting it. There are seven easy to spot peaks and the corresponding ROIs can be defined by a drop-off in intensity to a certain percentage.

Additionally, after conversion to black/white, finding the ROIs is much more stable with respect to potentially remaining bad pixels, as they get the same weight as any other pixel of an actual signal. Lastly, the peak intensities become roughly of the same order since only the spatial extension of the white areas determines the intensity profile of the projection. A comparison of the projection with and without conversion is seen in Fig. 4.

While unnecessary in most cases, we might also want to correct angular deviations after finding the ROIs (see Fig. 3), which can in our case arise from small misalignments of the spectrometer crystals. We can again facilitate the horizontal extension of the fluorescent spectra. The *signal to noise ratio*[3] of a single row spectrum should be maximal when the fluorescent feature is completely aligned in the horizontal direction. Thus, by maximizing the maximal row-wise signal to noise ratio within each ROI we can find the correction angle around which to rotate the ROI.

[2]We can obtain the "maximum noise level" by sectioning the image into 10×5 equal chunks and selecting the one with the lowest average intensity. We assume this one to consist of noise only (this assumption can of course be contested, but it turned out to work well for all images encountered so far) and thus simply select the maximal value inside this chunk.

[3]We use the standard definition of the signal to noise ratio as mean value divided by the standard deviation of the noise.

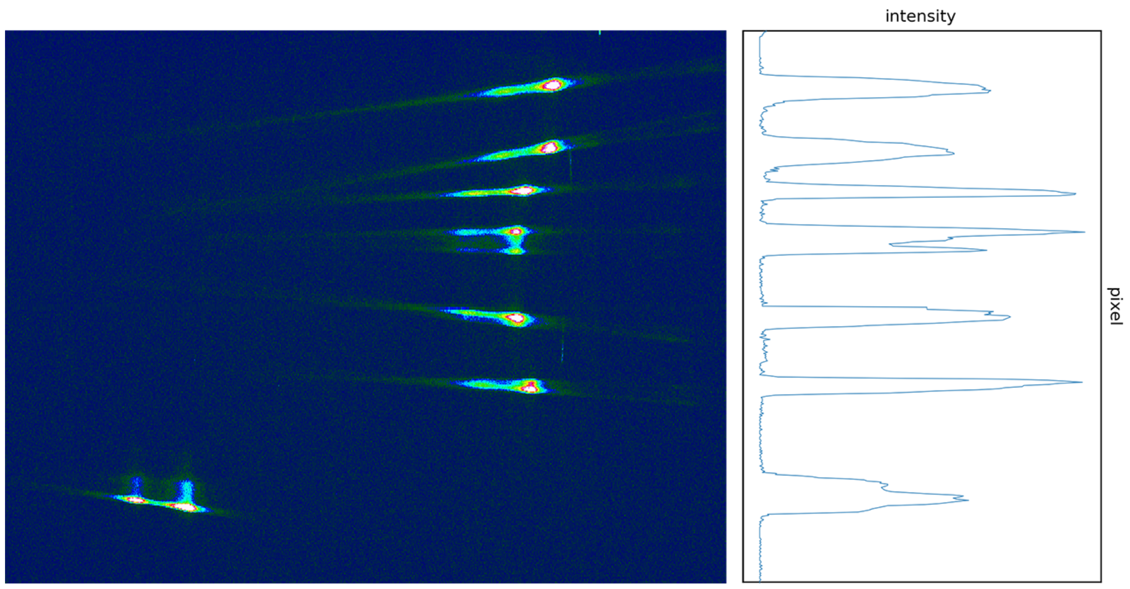

Figure 3: Example of a fluorescence image and its projection to the y-axis after conversion to black/white (not shown). We can clearly identify the ROIs as peaks in the projection. Double peaks, which can occur depending on the focusing of the spectrometer crystals, can be handled by introducing a distance threshold below which tho peaks are treated as one and the same.

4.2 Calibration

After finding the ROIs from a fluorescence image, we can carry them over to the calibration images and determine the exact horizontal position of the elastic line within the ROI (after potentially applying a small rotation mentioned above). Finding the positions works analogously to finding the ROIs. We convert now each single ROI to black/white, project it to the x-axis and look for a single "region of interest" containing the elastic line. Different to the initial ROI search, we perform an additional Gaußian fit to the original (non-black/white) intensity projection and read off the calibration position from the maximum of the fit curve. This extra step ensures that we retrieve important information about the actual maximum of the elastic line which was lost by converting to black/white. Repeating these steps for one or more calibration images with different incident energies yield several energy-position pairs for each ROI which can be used to convert the x-axis to an energy scale and proceed with the extraction of actual fluorescence spectra.[4]

A schematic overview of the whole ROI finding and calibration process is sketched in Fig. 5.

[4]The relation between horizintal positions and energies can be determined by using Bragg's formula. We should notice that the relation is not linear, but it can be obtained straightforwardly and thus we will not be concerned with the details here.

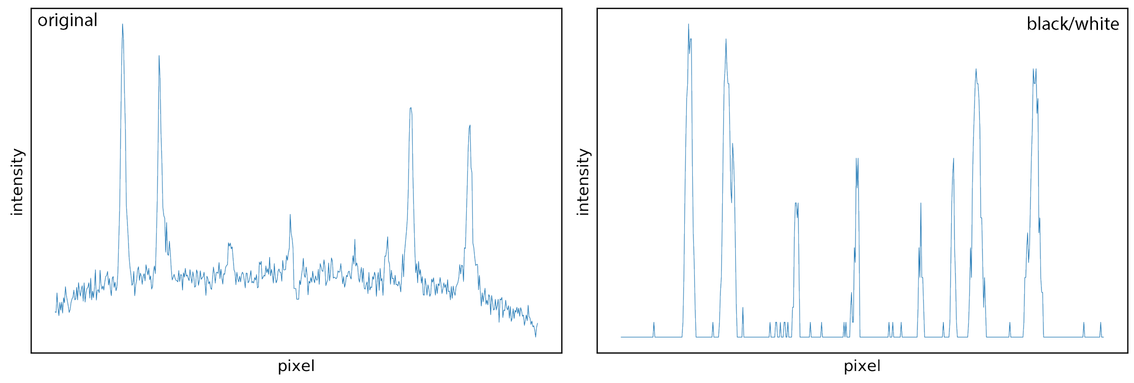

Figure 4: Comparison of the intensity profile along the y-axis before (original) and after converting the image to black/white. While we can barely make out the center peaks in the original graph, they become clearly visible and comparable in height to the outer peaks in the black/white projection.

5 Machine Learning

We have previously found methods to extract X-Ray spectra from the detector images reliably and in many cases without the need to adjust any parameters. Still we would like to reduce the necessary amount of user input, e.g. the number of ROIs in the images, to speed up and simplify the whole process. This is why, in the following sections, we will explore two different machine learning approaches to the task at hand – *neural networks* and *pattern detection* algorithms.

5.1 Neural Networks

Neural networks are nowadays used in many different areas of image processing and general data analysis, mostly due to their flexible design. We will, however, only use the simplest network architecture (fully connected) and only briefly discuss its structure in this section.

A fully connected neural network consists of an *input layer* of N neurons, each corresponding to one available input quantity, e.g. the intensity of each pixel in the image, and an *output layer* of M neurons, representing the quantities we want to determine from our input, e.g. the number, positions and orientations of the single spectra. By adding *hidden layers* in between with variable numbers of neurons, we can model any non-linear mapping $\mathbb{R}^N \longmapsto \mathbb{R}^M$.
For adjusting this map to return the correct output for each image, we need to create a training set of images with known outputs to feed the network. For more details on the

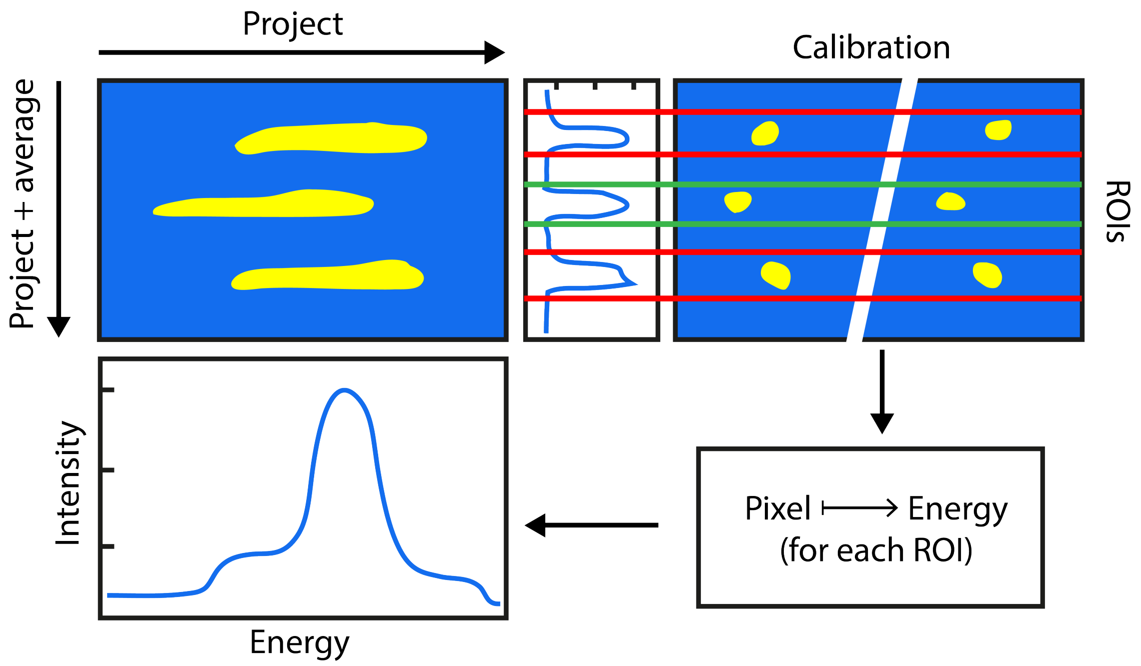

Figure 5: Schematic overview of the data extraction process. We start with a fluorescence image, project it (in black/white) to the y-axis and obtain ROIs. Those are applied to the calibration images to determine the elastic line position for each ROI. With this pixel to energy calibration, we can return to the original image, project the ROIs to the x-axis and get a single, averaged energy spectrum.

functionality of neural networks, a comprehensive study of the theory can be found in [2], particularly chapters 1 and 4.

In our concrete case, the detector images have a size of 1556×516 ($\simeq 800\,000$) pixels. Typical image classifiers using neural networks are working with much smaller images of several hundred or maybe thousand of pixels. Feeding the whole image as input to the network during training exceeds our available computation power by far, which is why we have to find a different way to extract information from the image.

Simply resizing the image is a common procedure in image recognition or classification programs, but is ill-advised in our case. Most features of interest in our images are rather close together, which means that rescaling of the image makes it even more difficult to distinguish ROIs or find calibration positions.
So besides changing the overall architecture of the network to cope with large numbers of input neurons, we can also change the input itself. When finding the ROIs, we are effectively making a decision row by row along the y-axis whether to appoint a row to a ROI or not. Thus, we can try to take only single rows of 1556 pixels as input and a binary classification $\{0, 1\}$ as output, encoding if the row belongs to a ROI or not. This way we

greatly reduce the computational efforts and produce fairly satisfying results using only a small set of training data. An example classification using the neural network is shown in Fig. 6. For this classification, a network with four layers ($1556 \rightarrow 100 \rightarrow 30 \rightarrow 2$) was trained over 25 epochs with 19 pre-classified training images. The efficiency on new test images after this training is at roughly 93%.

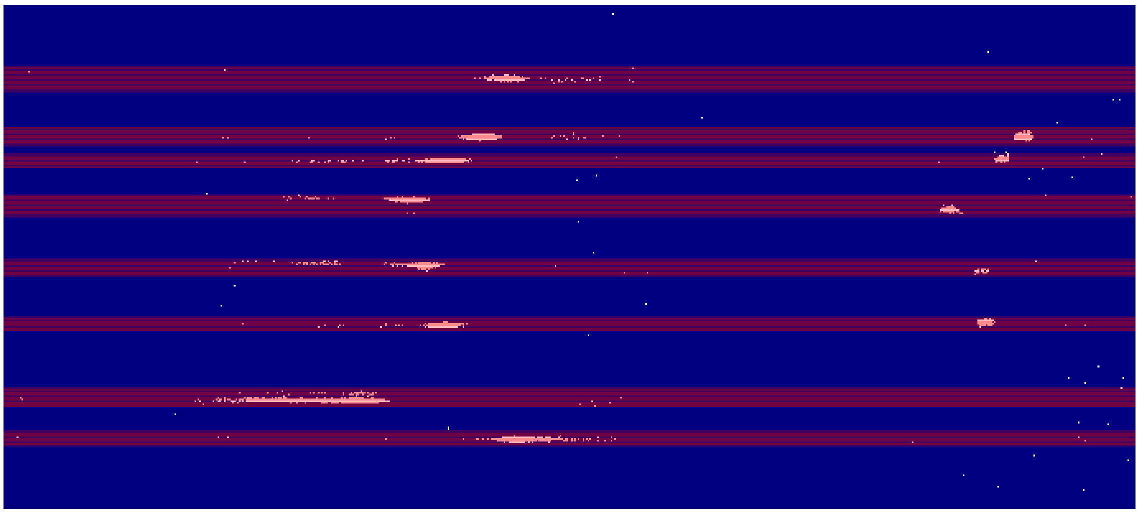

Figure 6: ROI detection of an example fluorescence image using the row-wise neural network classification. The underlying image was converted to black/white for better visibility only.

While we have now found a method to find ROIs and their number in particular, there still remains the task to find the correction angles and calibration positions which requires either the methods discussed in section 4 or at least a second network with different architecture and training sets.

5.2 Pattern Detection

An approach quite different to neural networks is using techniques of pattern detection in images. Most mobile phones nowadays are equipped with pre-configured facial detection and recognition models, finding (and recognizing) faces in a picture within fractions of a second. Instead of faces, we will try to find the elastic line features in the detector images to obtain the calibration positions, ROIs and correction angles all in one go.

The method we will apply is called Local Binary Pattern (LBP) detection and calculates a mask of 0s and 1s from a window of fixed size. This mask can be used as feature vector to decide whether the window is likely to contain the pattern of interest or not. By moving the window all across a given image, the algorithm finds areas which it thinks contain the desired pattern. For further reading, the original description of boosted classifier cascade algorithms can be found in [3] and an introduction to using it in comination with LBPs in [4].

To train such a pattern detector, we need to provide two sets of images – one *positive set*, containing exactly one of the patterns we are looking for, and one *negative set* of images completely without any of the patterns. An example of images we use to train our specific program is shown in Fig. 7. For a very simple proof of concept study, we use typical elastic line features from several images as positives and pure noise as negatives.

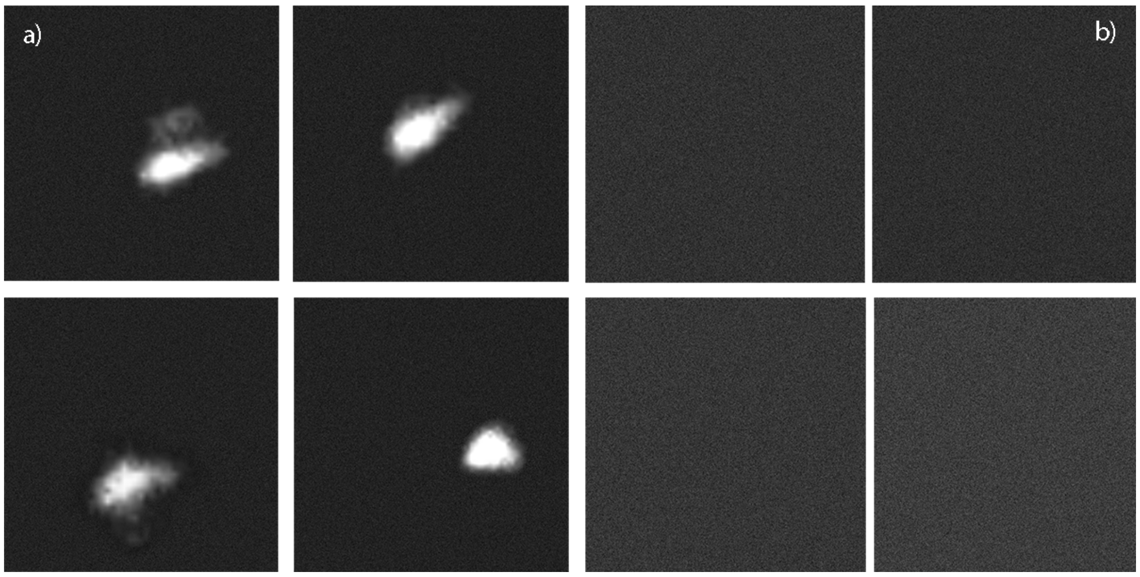

Figure 7: Examples of training data. Positive image set (a) with randomly placed, rotated and resized elastic line features and negative image set (b), consisting only of noise with different average intensities.

Fig. 8 depicts examples of detected elastic line features in different images. While the algorithm currently detects most elastic line features successfully, there are still many false positives coming either from de-focused lines which are detected as two separate lines or from completely different features, e.g. the fluorescence spectra (see Fig. 8). We can most likely avoid these kinds of errors by extending the training sets, in particular by adding the double peak elastic lines to the positive image set and images containing fluorescence features to the negative image set.

Once we have detected all elastic line positions – now both in the x- and y-direction by performing two separate Gaußian fits – for two or more calibration images, we can immediately calculate the correction angle of the ROIs and already have the necessary information for calibration available. This whole process is in general faster than apply-

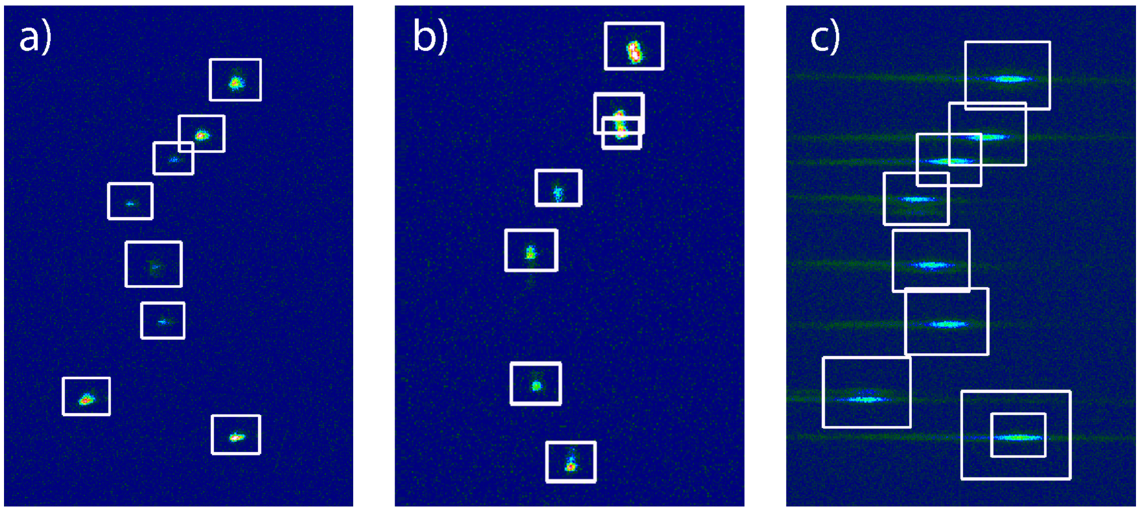

Figure 8: Comparison of the LBP detector applied to three different kinds of features. Typical elastic lines, even very faint ones are recognized accurately (a), defocused lines with double peaks are recognized as two separate lines (b) and entirely different features like fluorescence spectra are also recognized as elastic lines (c).

ing the method discussed in the previous section,[5] but most importantly, it requires no variable user input, provided that the pattern recognition works reliably.

6 Summary

As we have seen, there are simple, “hard coded”, algorithmic approaches to correcting errors and extracting X-Ray fluorescence spectra from the detector images. While they all rely on tunable parameters, within one experimental setup there is hardly any need for tuning and the methods apply to a great range of experiments with empirically determined default settings. As an alternative to these parameter dependent approaches, we have explored machine learning techniques and found that standard pattern detection algorithms offer a very fast and – provided proper training – reliable method to automatize the process.

[5]The pattern detection and numerical calculations are performed within $\mathcal{O}(10^{-2})$ seconds, whereas the computations steps described in section 4 require $\mathcal{O}(10^{-1})$, when including the angular corrections even $\mathcal{O}(1)$ seconds.

References

[1] Roberto Alonso-Mori et al. “A multi-crystal wavelength dispersive x-ray spectrometer”. In: *Review of Scientific Instruments* 83.7 (2012), p. 073114. DOI: 10.1063/1.4737630. eprint: https://doi.org/10.1063/1.4737630. URL: https://doi.org/10.1063/1.4737630.

[2] Simon S. Haykin. *Neural networks and learning machines*. Third. Upper Saddle River, NJ: Pearson Education, 2009.

[3] Paul Viola and Michael Jones. “Rapid Object Detection using a Boosted Cascade of Simple Features”. In: *Proceedings of the 2001 IEEE Computer Society Conference on Vision and Pattern Recognition* 1 (Feb. 2001), pp. I–511.

[4] Jo Chang-yeon. *Face Detection using LBP features*. 2008. URL: http://www.stanford.edu/class/cs229/proj2008/Jo-FaceDetectionUsingLBPfeatures.pdf.