Group 6 - Detection and Identification of Macromolecular Complexes in Cryo-Electron Tomograms Using Support Vector Machines

Omer Benami, Matan Lotem, Guy Rom, Saar Katz

September 18, 2017

Abstract

Cyro-electron tomography is the highest-resolving imaging technique to visualize biological samples in 3D under near native conditions. An SVM based tool for identification of multiple macromolecular complexes in a protein tomogram was implemented in python. The tool was trained and tested on artificial 2D tomograms constructed of geometrical shapes, and 3D generated tomograms constructed from density maps of different macromolecular complexes. The tool successfully identified over 90% of the objects in the tested tomograms. However, for for noisy tomograms, the object orientation is miss-calculated in over 50% of the cases. Nevertheless, the developed framework provides a good code base, on which further development and improvement of this tool can be implemented.

1 Introduction

In this workshop we implemented an SVM based method to identify multiple macromolecular complexes in a protein tomogram, based on the method described in [1]. Cyro-electron tomography is the highest-resolving imaging technique to visualize biological samples in 3D under near native conditions. In this method 3D density maps are re-constructed from 2D transmission electron microscope images of a frozen-hydrated sample at different tilt angles. The resulting images density maps have a resolution of 5-10nm, which are of the size scale of the macromolecular complexes constructing the protein, and so is sufficient to identify different macromolecular complexes. However the density maps suffer from low signal to noise ratio, and from limited tilt angle sampling. The state-of-the-art approach used to identify macromolecular complexes in tomograms, is based on constructing density maps from structural templates for specific macromolecular complexes at different orientations, and matching them based on (mostly local) correlations to the tomogram. These methods have limited success due to the above mentioned issues, and suffer from a high level of false positives.

The method discussed in this paper aspires to overcome the above mentioned challenges, by producing a list of candidates for placements of macromolecular complexes with a low rate of false negatives, at the cost of a high rate of false positives, and training an SVM to identify the true positives. The candidate list is produced in a similar way to the standard method, using correlations. For each candidate a set of rotational invariant features is produced and the SVM is then trained to classify the candidates based on these features. As there is a limited number of proteins that have been fully identified into specific macromolecular complexes, the SVM is trained on simulated tomograms.

We developed a python based tool with both a python and shell interface which can be trained to identify small density maps (templates) inside a large density map (tomogram). We implemented this tool for both 2 dimensional and 3 dimensional density maps. In this paper we will refer to the large density map as a tomogram and to the small density maps as templates for both 2D and 3D. The tool has three main workflows: training, identification, and testing, which can be accessed both from the shell and python interfaces. These workflows are constructed from several modules which can be accessed through the python interface only, and used to construct more workflows.

The tool was tested on 2D and 3D tomograms constructed from geometric shaped templates, as well as 3D tomograms constructed from PDB based templates. The tomograms were randomly generated, and different noise filters were applied. For 2D the SVM was trained on 100 tomograms of size 120×120 pixels and templates with angular resolution of 15°. For 3D, due the significantly higher runtime required, the SVM was trained on 10 tomograms of size $120 \times 120 \times 120$ and templates with angular resolution of 60°. For all cases, the identification flow produced an over 90% success rate of assigning the correct templates, with a negligible number of false positives. However for noisy tomograms, the correct tilt was produced only for 45% of the identified templates.

2 Method Outline

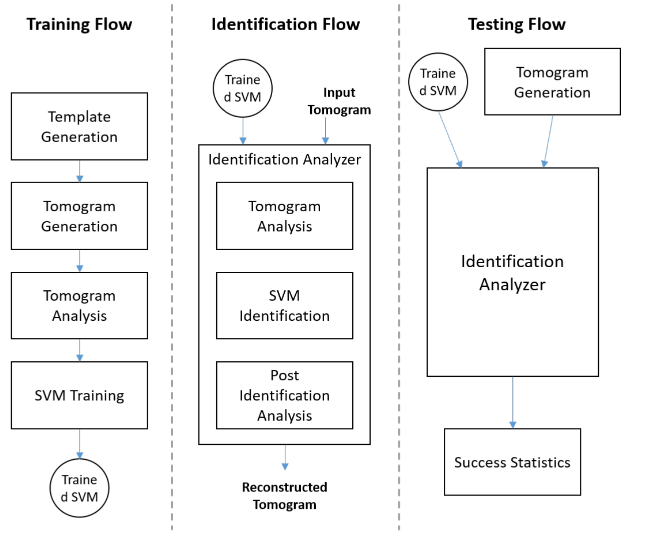

Figure 1: The major tool workflows.

The tool developed has three main workflows as shown in figure 1: training, identification and testing, each consisting of several stages, which roughly (but not exactly) correspond to the different python modules which will later be discussed. The training workflow consists of the four stages: (1) template generation (2) training-set tomogram generation (3) tomogram analysis (4) SVM training. The output of the training workflow is a trained SVM and the generated templates, which are both required for identification. The identification workflow receives as input a trained SVM, corresponding templates (used during the training), and an input tomogram, and consists of 3 stages: (1) tomogram analysis (2) SVM identification (3) post identification analysis. The output of this stage is an identified tomogram, i.e. a list of of templates with position and orientation, which should match the composition of the input tomogram. The testing workflow receives as input a trained SVM (and templates) and consists of 3 stages: (1) test-tomogram generation (2) run of the identification flow (3) success statistics calculation. The different workflows can be applied both in 2D and 3D, depending on the templates used.

The python modules used to construct the different workflows are defined as follows: (I) Template generation module which is used to produce density maps of the different templates that should later be identified. This module can be used to provide different types of templates, namely 2D and 3D geometric templates, and 3D PDB based templates. (II) Tomogram generation module which is used to construct tomogram density maps, using the templates outputted by the template generation module according to different heuristics. (III) Tomogram analysis module which applies the major part of the logic on a given tomogram, and is divided into several sub-modules: (i) correlations calculation for other submodules (ii) candidate selection of suspected template locations (iii) candidate features extraction for SVM training and identification (iv) candidate labeling - either as ground truth for SVM training or based on SVM identification (v) post processing of candidates (relevant only for identification flow). (IV) SVM training module which receives the output of the tomogram analysis module for a set of tomograms and trains the SVM. (V) Statistics module which receives a tomogram with a known ground truth and the output of the SVM training module, and calculates different success measures. In the following section we will describe the different modules and the ways in which they are connected.

2.1 Template Generation Module

The template generation module is used to generate density maps of known macromolecular complexes in different orientations. We support the generation of artificial templates for testing purposes both in 2D and 3D, and the generation the 3D templates of real macromolecular complexes from PDB files. For each template a set of tilted templates are generated for a discrete set of tilts on a sphere (circle for 2D), specified by an angle resolution parameter. The number of tilts, $\#_{tilts}$, is a function of the angular resolution parameter α (given in degrees) as follows:

$$\#_{tilts}^{(2D)}(\alpha) = \frac{360}{\alpha}. \qquad (1)$$

$$\#_{tilts}^{(3D)}(\alpha) = \left(\frac{180}{\alpha} - 1\right) \cdot \frac{360}{\alpha} \cdot \frac{360}{\alpha} + 2 \cdot \frac{360}{\alpha} \sim \frac{1}{\alpha^3}$$

The tilted templates are created such that the center of mass of the template is centered in the density map, and all templates are padded to the same cube (square) size with an edge length s. The 2D templates are an integral part of the tool, and are generated at the start of every run. On the other hand, the 3D templates generation depends on external Chimera based code, and the generated templates are stored as files. These templates are then lazily loaded by the tool, in order to save memory.

The geometric templates prove useful for testing and development purposes. We therefore support templates with different symmetries: circle, square, L shaped and mirrored L for 2D, and sphere, cube and an L shaped prism for 3D. The template generation module receives as input a list of required templates (specified by geometric shape name or PDB file path), template specific parameters (such as such as sphere radius or PDB resolution), and an angle resolution parameter. It outputs the density maps of the tilted templates and corresponding template metadata. The runtime of the analysis, which will be discussed in the following sections, depends linearly on the number of tilted templates, that is on $n \cdot \#_{tilts}$, where n is the number of different templates and $\#_{tilts}$ is calculated according to equation (1).

2.2 Tomogram Generation Module

The tomogram generation module receives as input a set of tilted template objects and tomogram generation criteria, generates a tomogram density map, and outputs a tomogram object containing the density map and the placement of the templates in it. Both 2D and 3D tomograms can be generated, depending on the supplied templates. The size of the tomogram is specified as an input parameter, and the positioning (location and orientation) of the templates can either be manually specified or randomly distributed.

Different methods of random distribution were explored. Poisson disk distribution [2], which was expected to produce tightly packed tomograms, was implemented for 2D tomograms, but proved highly time consuming for 3D tomograms. Therefore, a naive randomization scheme, i.e. randomly select a coordinate and try to place the template, was implemented. This scheme however results in sparse distribution of the templates, due to the complexity of preventing template overlap. The tilt of template is randomly selected from the supported tilts.

Once the density map is created, different noise filters can be applied. Two such noise filter were implemented: (1) Random noise, i.e. adding a randomly distributed density map to the tomogram, with a parameter specifying the magnitude. (2) An overall Gaussian blur with specified σ and window size. In our implementation both filters are applied on a generated tomogram in order to produce a noisy tomogram.

2.3 Tomogram Analysis Module

This module receives an input tomogram and a set of (tilted) templates, and outputs a list of candidate placements of the templates on the tomogram. Each candidate consists of the position of its center of mass, its orientation (specified by Euler angles), a set of features for SVM training or identification, and a suggested label associating the candidate with a specific template. The module runs either in “training mode” in which the label is acquired by comparison to the known composition of the tomogram, and is then used as the ground truth for SVM training, or in “identification mode” in which the label is provided by a trained SVM. The candidates do not necessarily correspond to the templates that compose the tomogram and therefore a junk template label, which indicates the candidate was not associated with any template, is included.

Candidate Selection: The candidates are produced by the candidate selection sub-module, based on maximum correlations with any one of the tilted templates. The candidate list should contain a candidate for every position in which a template appears in the tomogram, disregarding which specific template and what tilt it appears in. The SVM will then be trained or used to classify the candidates, either to the correct template or as a junk candidate. As the SVM only considers the selected candidates, if no candidate was selected for a position in which a template appears in the tomogram, then that position will be missed in the identification process. Therefore the candidate selection sub-module is tuned to miss a minimal amount of such positions, at the cost of many junk candidates. On the other hand there is no need to produce more than one candidate for the same position (just because that position shows a high correlation with several templates), as later analysis stages are anyway indifferent to how the candidates were produced, and will therefore treat the multiple candidates identically.

The selection process is thus implemented by finding local maxima in a matrix consisting of the maximum correlations over all templates and tilts. As different tilts or templates might produce slightly different maxima which correspond to the same position, a Gaussian blur is applied to the maximum correlations matrix before running the peak detection. The position of the candidate might then be slightly shifted in relation to the actual placement of the template and this has to be taken into account in the subsequent analysis stages. The peak detection method also has a threshold for local maximum detection, which can be tuned in order to minimize missing positions of templates composing the tomogram.

Features Extraction: For each candidate the features extraction sub-module is then used to calculate a features vector which will be used for SVM training and identification. In this implementation the features vector consists of the maximum correlation over all tilts for each template. Therefore the length of the features vector is equal to the number of templates. As different tilts might produce slightly different maxima (due to the discrete representation of the the templates and the blur induced in the peak detection), the maximum for each template is taken over a small area of the maximum correlations matrix, around the selected position of the candidate.

A disadvantage of the choice of features method implemented is, that if only a small number of different templates are considered, then the features vectors produced are short and the SVM has a limited amount of data to learn from. One possible solution to this problem is introducing a set of templates which do not appear in the tomogram, but generate more features. These templates can either be artificial templates which span the vector space of 3D (or 2D) matrices such as spherical harmonic based templates, random templates, or just some other macromolecular complexes which are not expected to appear in the tomogram. A different approach, suggested in [1], is using spherical harmonics decomposition based features, in which case the length of the features vector can be controlled by choosing the order up to which the spherical harmonics decomposition is applied. The artificial template solution is already supported by the existing framework, but was not tested, while the spherical harmonics decomposition requires further development.

Candidate Labeling: Each candidate is also given a label associating it with a specific template. This is the only part of the analysis which differs between "training mode" and "identification mode". In "training mode" the composition of the tomogram is known, such that a candidate can be associated with a template at a specific position if the distance between the two positions is smaller than some threshold distance. If the candidate has no closely positioned template in the tomogram, it will be labeled as a junk candidate. The distance threshold must be tuned together with the radius of the Gaussian blur in the peak detection, in order to avoid mislabeling of candidates. The label in this mode will be used as the ground truth for the SVM training, and therefore must be accurate. In "identification mode" the candidate is given a label by the trained SVM.

Post Analysis: By definition the features extracted are rotationally invariant, and therefore each candidate can only be associated by the SVM with a specific template, but not with a specific orientation. Once a candidate has been associated with a specific template, it can also be associated with the tilt that produced the maximum correlation for that template. This will be taken as the output tilt of the analysis. Notice that this approach means that the the angular resolution of the tilted templates is the resolution in which the the orientation of a candidate can be determined. In addition, the candidate's position can be updated to the position that produced the maximum correlation for this template, fixing small shifts due to the choice of a specific position in the peak detection.

Correlations Calculation: The correlation of each template with the tomogram is used multiple times during candidate selection, features extraction, and post analysis. The runtime of a single tomogram analysis, $T_{analyze}$, is dominated by the correlations calculation, and therefore it is calculated once at the beginning of the tomogram analysis, and the results are then used by the other sub-modules. The correlations are calculated using FFT and therefore the required runtime scales as

$$T_{cor}(N) \sim N \log N \tag{2}$$

where N is the size of the larger of the two objects compared, in this case the tomogram. The total analysis time therefore depends on the dimension d (2D/3D) and edge length S of the tomogram according to equation (2), the number of templates n, and the number of tilts $\#^{(d)}_{tilts}(\alpha)$ which depends on d and the angular resolution according to equation (1):

$$T^{(d)}_{analyze}(S, n, \alpha) = n \cdot \#^{(d)}_{tilts}(\alpha) \cdot T_{cor}\left(S^d\right) \tag{3}$$

$$T^{(2D)}_{analyze} = C^{(2D)} \cdot \frac{n \cdot S^2 \log S}{\alpha}$$

$$T^{(3D)}_{analyze} = C^{(3D)} \cdot \frac{n \cdot S^3 \log S}{\alpha^3}$$

where $C^{(2D)}, C^{(3D)}$ are constants, and $\frac{C^{(3D)}}{C^{(2D)}} \approx 10^5$. In order to reduce memory demands, only two tomogram sized matrices for each template are kept - one with the maximum correlation value taken over all tilts, and the other with the tilt that produced the maximum correlation for each coordinate. These two matrices contain all the information required by the other sub-modules, and this scheme results in reduction of the memory cost by a factor of $\#^{(d)}_{tilts}(\alpha)$ (which for $\alpha = 30°$ in 3D is ~ 1000).

2.4 SVM Training Module

A support vector machine (SVM) [3] is a supervised learning model for classification of a set of data points. An SVM is trained on a set of data points in known different categories, and builds a model which can then be used to classify a second set of data points into these categories. The basic SVM scheme is constructed to distinct between two categories. This is done by finding a hyper-plane that separates the training data points of the two categories. The so-called kernel trick is employed in order to map the data points to a high dimensional space where such a hyper-plane exists [4]. A second set of data points are then classified with respect to this hyper-plane. SVM method can be generalized to distinct between multiple classes.

The SVM training module receives as input a set of data points consisting of a features vector (which is interpreted as coordinates) and labels. The label links the features vector to a specific template class, or the junk template class. A trained SVM is coupled to the templates on which it was trained. The output of the SVM training module is therefore a trained SVM object and the corresponding templates. The “scikit-learn” python library SVM implementation was used as the core of this module.

2.5 Statistics Module

No.	Category	Definition
1	True Label and Tilt	A candidate corresponding to a composition template was classified correctly.
2	True Label False Tilt	A candidate corresponding to a composition template was classified correctly, but was assigned the wrong tilt.
3	True Junk	A candidate with no corresponding composition template was correctly identified as a junk candidate.
4	Wrong Label	A candidate corresponding to a composition template was classified as a different template.
5	False Junk	A candidate corresponding to a composition template was classified as a junk candidate.
6	False Existence	A candidate with no corresponding composition template was associated with some template.
7	Missing Candidate	A composition template has no corresponding candidate

Table 1: Identification results classification. First 3 result are considered as a success and the rest as failure.

In this module, the output of a trained SVM identification workflow, run on a tomogram with a known composition, is compared with this composition. In this section "candidate" will refer to an output (position and label) of the identification workflow, and "composition template" will refer to a specific template (position and label) composing the evaluated tomogram. The results of the comparison can be classified into 7 categories as shown in table 1. The first 3 classifications can all be regarded as a successful identification. Classification 2 corresponds to a successful SVM identification, but the post analysis producing a wrong tilt. The other classifications are all regarded as failed identification. Classification 4,5,6 can be attributed to the SVM assigning a wrong label, while classification 7 is attributed to the candidate selector not assigning any candidate to a composition template.

3 Experimental Results

In this section we will first discuss 2D tomograms for which results can be easily visualized, and then analogously discuss 3D results

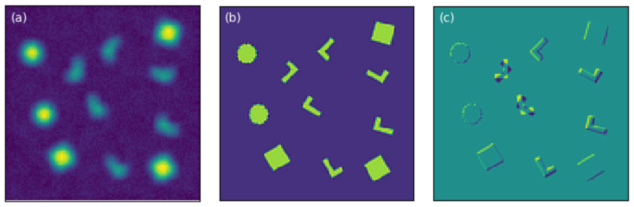

Figure 2: (a) Randomly generated noisy tomogram. (b) Reconstructed tomogram. (c) Difference between reconstructed tomogram and exact reconstruction.

Results for 2D tomograms with and without noise showed a high success rate. The SVM was trained on a set of 100 randomly generated tomograms of size 120×120 pixels, once with and once without noise, constructed from 2D geometrical templates (circle, square, L, mirrored L) with an angular resolution of 15°. The trained SVM was then used to identify 100 randomly generated tomogram of similar make. The noise applied consists of both Gaussian blur and random noise and is illustrated in figure 2, panel (a). Notice that the templates in the reconstructed tomogram are slightly shifted with respect to the original tomogram construction as can be seen in panel (c), and two L templates were mislabeled.

Without noise, 100% of the templates were identified correctly by the SVM, out if which 95% were assigned the correct tilt, and the remaining 5% were miss-assigned due to rotational symmetry of squares and circles. No junk candidates were selected. With noise, 92% of the templates were identified correctly, and the remaining 8% were mislabeled (no false negatives). However only 45% out of the successfully identified candidates were assign the correct tilt. This is again attributed to the rotational symmetry of some of the templates, which dominates once noise is added. All selected junk candidates were labeled correctly as junk.

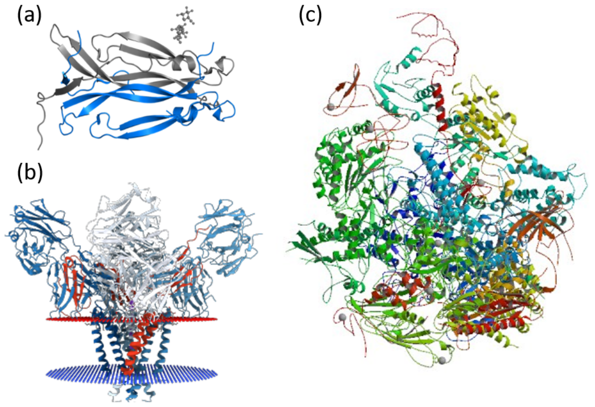

Figure 3: PDB templates used for training and identification: (a) 5n92 (b) 1k4c (c) 1y1v.

Results for 3D PDB based templates also showed a high success rate. 3 PDB based templates were generated for: 5n92 - Crystal Structure of Human IL-17AF [5], 1k4c - Potassium Channel KcsA-Fab complex [6], and 1y1v - Refined RNA Polymerase II-TFIIS complex [7]. All 3 were generated in $35 \times 35 \times 35$ pixel cubes, with an angular resolution of 60°. Random tomograms of size $120 \times 120 \times 120$ pixels, with or without noise, were generated both for training and identification. According to equation (3), the analysis runtime for a tomogram as such is greater by a factor of ~ 600 than the runtime for the 2D tomogram. Therefore the SVM was trained on only 10 such tomograms and then 5 tomograms were identified.

Without noise 100% of the templates were identified correctly by the SVM, and assigned the correct tilt. No junk candidates were selected. With noise 93% of the templates were identified correctly by the SVM, out of which 46% were assigned the correct tilt. The remaining 7% were mislabeled by the SVM. All junk candidates were correctly classified as junk. We attribute the high success rate to a significant difference in the total weight between the different templates. However, we still do not have an explanation for the the high tilt miss-assignment rate.

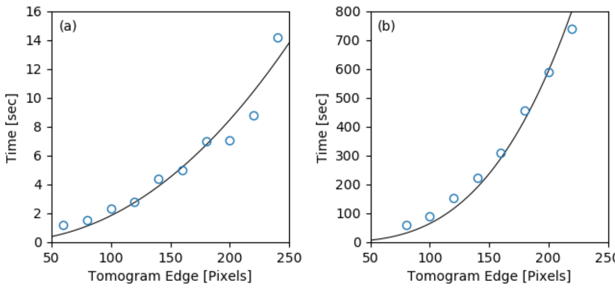

Figure 4: Runtime scaling for analysis for different tomogram sizes. (a) Runtime of consecutive analysis 10 2D tomograms plotted for different edge size S. A trend line $\sim S^2 \log S$ is plotted in black. (b) Runtime of analysis of a single 3D tomogram plotted for different edge size S. A trend line $\sim S^3 \log S$ is plotted in black.

In figure 4 the tomogram analysis runtime for different tomogram sizes is plotted for 2D (a) and 3D (b) as a function of the tomogram edge size. The runtime measurement was taken on an Intel Core i7-4510 laptop. For 2D the runtime was measured over analysis of 10 tomograms, with 4 different templates and angular resolution of 15°, i.e. 96 tilted templates, leading to 960 2D correlations calculations in total for each tomogram size. For 3D the runtime was measured over the analysis of a single tomogram, with 3 different templates and angular resolution of 60°, i.e. 249 tilted templates, and the same number of 3D correlations calculations for each tomogram size. For both graphs a trend-line according to equation (3) is plotted, and the runtime scales roughly correspondingly. As expected the runtime for the 3D case is significantly longer.

4 Conclusions

The tool developed during this workshop produced high successful identification and low false positive rates, even with significant noise in the evaluated tomograms, for both 2D and 3D. Without noise, the tool identified correctly 100% of the templates, and assigned the correct tilt to over 95% of them (the remaining 5% due to rotational symmetry of some of the templates). With noise, created by adding a random density map and applying a Gaussian blur to the tomogram, the SVM still identified the correct templates in over 90% of the cases, the rest were mislabeled, and all false candidates were correctly labeled as junk. However, for the noisy tomograms, less then 50% of the identified templates were assigned the correct tilt. In 2D this can be attributed to the rotational symmetry of some of the templates, which is broken by the noise in a random direction. However for 3D this explanation does not hold and further investigation is required.

The major part of our work in this workshop was directed towards producing a stable tomogram analysis framework. The 2D templates and tomograms proved a crucial tool for testing, understanding and debugging our results. Many effects occurring in the 3D tomograms could be reproduced in 2D, visualized and quickly analyzed, thus accelerating the development. Each new feature was first developed in 2D, where runtime costs are significantly lower, allowing for shorter testing and debugging cycles, and only once it was stabilized, it was tested on 3D.

However, the training set 3D tomograms are still far from mimicking real tomograms, and further development steps should be directed in this direction. Therefore the tool was not tested on a real tomogram. The most significant difference is the spacing of the templates, which in real tomograms are expected to be tightly packed and almost overlapping, but in our test tomogram are only sparsely spaced. We also expect different macromolecular complexes, composing a test tomogram, to be of similar size and weight with respect to each other, while in our test tomograms the templates had significant size differences. Once these two improvements have been implemented, we suspect that the SVM success rate will drop. Tweaking some of the thresholds might improve results, but we expect further development of the identification scheme will be required.

We suspect the feature vector extracted at the moment might be degenerate, with only 3 values for each vector. For our test cases this proved not to be a major problem, however for templates of similar size and shape this might be. A possible solution for this issue is introducing artificial templates, which will only be used for feature extraction, producing a longer features vector, and will be transparent to the labeler and SVM. These features can be chosen to be other PDB templates, or designed artificially to better span the 3D density map space. A different possible approach is using an all together different features scheme, based on spherical harmonics decomposition [1], in which the length of the features vector is controlled by a parameter setting the order of the decomposition.

Training the tool on a large set of 3D tomograms, or with high angular resolution, proved to require long runtimes. This is due to the analysis of a 3D tomogram being significantly more demanding than that of a 2D tomogram. Each correlation calculation in 3D is more costly by a factor of the length of the tomogram's edge. In addition, the number of calculations required to calculate, scales linearly with the number of tilts, which in turn scale cubicly with the angular resolution, including a high constant factor. These issues can be addressed by parallelization, and two parallelization schemes are proposed: (1) parallel analysis of different tomograms, and (2) parallel correlations calculation.

The first scheme can easily be implemented, without major changes to the code, as the SVM is only trained on the accumulated output of all analyses, which are independent of each other. This scheme is expected to yield a linear improvement by the number of cores involved. The different tomograms can then be analyzed either on different cores, or on completely different machines. The second scheme requires some changes in the correlations calculation sub-module. The correlations are calculated for each tilted template separately, and only later aggregated. Therefore, the calculation for each tilted template can be done separately, again yielding a linear improvement according to the number of cores involved. Distributing these calculations between different machines might be a little more complicated but still possible. We suggest implementing both schemes, such that different tomograms can be analyzed on separate machines, while the analysis for each specific tomogram utilizes all the machine's cores.

We therefore conclude that a basic template identification tool for both 2D and 3D was successfully developed over the past several months. The tool produced reasonably good results for its test cases. There are still many possible improvements, which we believe can be implemented on top of our developed framework.

References

[1] Chen, Y., Hrabe, T., Pfeffer, S., Pauly, O., Mateus, D., Navab, N., & Förster, F. "Detection and identification of macromolecular complexes in cryo-electron tomograms using support vector machines", 9th IEEE International Symposium on Biomedical Imaging, 1373-1376, 2012. https://doi.org/10.1109/ISBI.2012.6235823

[2] Dunbar, D.; Humphreys, G. (2006). "A spatial data structure for fast Poisson-disk sample generation". ACM Transactions on Graphics (TOG), v.25 n.3. doi:10.1145/1141911.1141915

[3] Cortes, C.; Vapnik, V. (1995). "Support-vector networks". Machine Learning. 20 (3): 273–297. doi:10.1007/BF00994018.

[4] Boser, B. E.; Guyon, I. M.; Vapnik, V. N. (1992). "A training algorithm for optimal margin classifiers". Proceedings of the fifth annual workshop on Computational learning theory – COLT '92. p. 144. ISBN 089791497X. doi:10.1145/130385.130401

[5] Goepfert, A.; Lehmann; S.; Wirth, E.; Rondeau, J.M. (2017) "The human IL-17A/F heterodimer: a two-faced cytokine with unique receptor recognition properties" Sci Rep 7: 8906. doi:10.1038/s41598-017-08360-9

[6] Zhou, Y.; Morais-Cabral, J.H.; Kaufman, A.; MacKinnon, R. (2001) "Chemistry of ion coordination and hydration revealed by a K+ channel-Fab complex at 2.0 A resolution". Nature 414: 43-48. doi:10.1038/35102009

[7] Kettenberger, H.; Armache, K.J.; Cramer, P. (2004) "Complete RNA polymerase II elongation complex structure and its interactions with NTP and TFIIS". Mol.Cell 16: 955-965. doi:10.1016/j.molcel.2004.11.040