Cellpack Parameter Exploration

Magdalena Schwarzl*
University of Vienna

ABSTRACT

The project described in this text is the development of a visualization tool, helping a user to explore the parameters used by his own software. Many different parameters have an influence on other parameters and on the final output. The outcome of the user's software are 3D geometrical models of biological structures as for example HIV. They are generated using a packing approach, placing geometrically described molecules and proteins in a volume. The goal is to compute a model looking as similar as possible to the images examined under a microscope and therefore serving as a realistic representation. The final models can the be used in research and in teaching. Additionally to getting a good representation and communication basis for the biological structures, finding out how to set parameters to get similar results to those seen under a microscope might also answer some research questions or arise new ones. For the exploration of the parameter space and resulting models, the approach of inverse design [8] has been used and should improve the users' actual work flow.

Index Terms: Visulatisation, HIV, Cellpack, Model, Packing, Exploration, Image, Binning

1 INTRODUCTION

In biology a main subarea is visualization and representation of biological structures as proteins, molecules and cells. It is not only needed for teaching aspects but also for a better understanding of this complex tiny objects and doing research. Some very impressive results can be found in [9].
These representations used to be drawn by hand simply copying an image of what could be seen under a microscope. As the hardware improved, it is now possible to create virtual models in 3D. This can be done by modelling the geometry of each part individually in a modelling software as Maya [1], Blender [2] or Cinema4D [3]. It would take hours of work to build this models and doing changes later if some new research arise could lead to another dozens of hours. A better approach here is to compute this models, by describing proteins and molecules as a sub-structure once and repeat to create the final structure as they occur multiple times in for example viruses. Of course this sub-structure-molecules influence each other and are not fixed to single positions as molecules in real life interact with each other. The location of a molecule gives important characteristics to the biological structure and functionality.
If one wants to create a good model, it is important to understand the biology behind to correctly implement the behaviour of the proteins and molecules between each other and among the whole object. Therefore, when trying to build the model one has to learn about the biology first and then try to model the observations by reconstructing them virtually. Once this was done, there might be some changes needed to improve the model and make it a more realistic view. This does not only help to get a better model but also might give interesting insight in the behaviour of different parts in the model and their interactions among each other as well as influences on the whole structure.

*e-mail: magdalena.schwarzl@gmx.at

2 MOTIVATION

Graham Johnson and his team developed a software called cellpack [10]. It is a free tool that can be used stand alone or as a plug in for Blender [2], Cinema4D [3] and other modelling programs. The tool computes 3D models of biological cells and molecules (in this work we focus on HIV) using a packing approach and bioinformatical data. Each model is described in a recipe consisting of ingredients (modeling e.g. proteins), other recipes (combinations of ingredients modelling for example molecules) and some parameters to reflect the preferred location for each protein. Each ingredient can place itself multiple times in the final model as the same protein will occur multiple times in the virus (when using plugins for Maya [1], Blender [2] or Cinema4D [3] and similar tools, the geometry is loaded once an copied then). Additionally to a geometrical description each ingredient-type has its own attributes (parameters) and preferences where to pack itself(choose a position for placement inside the volume without overlapping other ingredients that have already found their place). These include different packing algorithms as PandaBullet and jitter (varying in precision and computation complexity and therefore also in time) and more biological aspects as molarity and different binding probabilities to other ingredients. The parameters model preferences of proteins to pack close to other proteins or close to the surface. Additionally the final position also depends on the order in which ingredients are packed (the first ingredients packed will easily find a place, later once might have difficulties), therefore each ingredient has a packing priority.
Using the software, a user can decide which ingredients and recipes to use to create the desired structure. Additionally it is possible to set values for the parameters. Another option is to load some predefined recipes as for example HIV.
Apart from the settings per ingredient, there are also general settings for the entire model (again packing approach, random seed, ...), which may be overwritten by per ingredient settings or used as default for ingredients without special parameters.
At the moment a lot of this parameters are simply set to heuristic default values and there is no documentation about the influence that each parameter has on the final output. This is not only a problem for a potential user, who will have difficulties in finding out, which parameters to change to get the desired improvement on the model but also for the developers themselves, as the parameter space got huge and it is difficult to keep an overview of all the parameters and their behaviour. This complexity is also increased as the parameters can influence each other and may be overwritten by others.
The experts' goal is to improve the available models and make the software usable to a wider community. This requires documentation about parameters, finding out useful default values, validating them and define ranges that are useful as well as knowing the influence of parameters between each other and on the final model.
In the case we focused in this project, a good model would have uniform distribution, meaning that there are no empty spaces in the volume. All the ingredients are spread over the whole volume and not all ingredients packed together in one corner. From a biological viewpoint, inside a cell, the different molecules would move around and spread over the whole cell using all the space they have.

This is where the visualization tool implemented in this project should aid. It should help to explore the parameter space more efficiently, systematically and finally find out the influence of the parameters on each other and on the final model created.

2.1 domain research questions

In general, the domain expert's goal is to improve the models generated by cellpack [10] and increase the number of users, which is at the moment limited to the developers themselves. The models are validated by domain experts, performing several tasks as comparing different models to microscopic data, testing different parameter combinations and their results, finding parameters and algorithms to model biological attributes (e.g. binding probabilities between molecules) and finally deciding the tradeoff between accuracy and computational complexity.
At the moment the research focus is on finding a way to control and modify the interactions between two ingredients (namely MA and ENV form the HIV 0.1.6 recipe). The resulting models are compared to real microscopic data and should of course be as equal as possible to them. For this task Figure 6 in the developers' publication [10] (for further explanation of the actual data used see below) is used. It shows at which location the specified molecules are found in the packing area at which frequency (the yellow areas indicate a higher frequency of molecule). To model these interactions, probabilities for packing at a specific location and next to a specific other ingredient are influenced. This means, there is a probability for MA to pack next to an existing MA (MA-MA), next to an existing ENV (MA-ENV) and also for ENV, to pack next to an existing MA (ENV-MA), or next to an existing ENV (ENV-ENV). When modifying the ingredient specific and the global parameters, the resulting models also change. The goal is now, to find a parameter set (a value for each parameter), creating correct models (compared to the microscopy data).

Arising research questions would be:

1. Which parameter sets lead to uniform distribution? Which sets actually give uniform distribution? Which sets do not and why?

2. How do different parameters influence each other concerning the distribution of objects? Which parameters influence the distribution at all? What effects are caused by changing specific global or local settings?

3. What are useful ranges and default values for parameters? Which parameter set(s) models the ENV-MA interaction best and why? For each set, what is the distance between ENVs and MAs?

2.2 work-flow

2.2.1 current work flow

At the moment the work flow is very inefficient. For each parameter a value is specified, which will be referred to as parameter set (one specific value for each parameter creates one parameter set) in the following. Having a value for each parameter, the resulting set is used to create a single model of the biological structure. As there are also random numbers generated for the computation of the model, many models using different random seeds but the same parameter set are used, to be able to see the effect on the distribution. All the models are analyzed together, to see the influence of the actual parameter set, without a bias of the random seed. Once the parameter set is analyzed , a new parameter set is specified an another run of models with different seeds is computed. The computation time of one model depends on the complexity of the objects, the number of runs with different seeds used (1000 for the analysis in [10]) and also the actual values for the parameters (e.g. influence on the granularity of the packing grid). Additionally to a long computation time within each iteration, a lot of iterations will be needed to find a good result as only one parameter set is explored in each iteration. This leads to many repetitions of the Waiting-Step, which makes the whole work flow inefficient Figure 1 shows the current work flow.

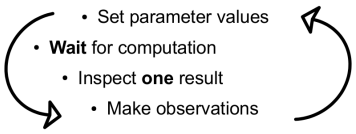

Figure 1: Shows the current work flow, computing only one model (many models with the same set but different seeds) within each iteration and therefore needing a lot of iterations to find a good model, resulting in a lot of time wasted for waiting.

2.2.2 new work-flow and solution approach

The new work-flow uses the idea of inverse design, described in [8]. Similar to the actual work-flow , the users starts specifying values for the parameters. But instead of single values, for some selected parameters (active parameters) ranges instead of specific values are going to be specified, for parameters that should not be inspected in detail in a specific iteration, default values not ranges as in the old work-flow are used. Within the specified ranges, a lot of parameter sets are created by randomly selecting values within the ranges for each of the active parameters. Each of this sets is again run with many different seeds as before. Afterwards the results of all the parameters are clustered together using a k-means algorithm at the moment based on the distribution of the ingredients. The actual values used as vectors for the k-means are the heights of the bars, described in Figure 3. The goal is to put similar result models together leading to clusters that have similar output models combined. The user now can analyze the models, find the desired ones and see which input parameters brought good results. This should decrease the number of iterations needed to find a good setting which will make the whole work-flow more efficient and enable the user to have one waiting step that could be used for any other activity instead of multiple small ones, that would just be a waste of time as doing something else in between is not efficient. Figure 2 shows the new work-flow .

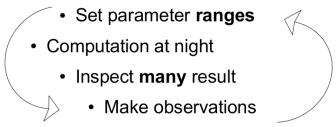

Figure 2: Shows the improved work-flow , computing many models (each model also with different seeds) within each iteration and decreasing the number of iterations needed.

2.3 users

The primary users of the tool is the developer team of cellpack [10], whose goal is to improve the models of their own software changing the values of different parameters. As they are all experts on cellpack and working on the models for a long time now, they are definitely all expert-users, bringing in some background knowledge about statistical methods and computational analysis as well, which is also required for their own software. Additionally they generally know what different parameters are doing - or what they would want them to do - and how they could influence the computed

model, so they understand the parameter names and different values for them. They also have an idea about what values would make sense for specific parameters, and whether to use them as boolean values or integer typed ones.
At a later point the tool might be used by some students that also work with cellpack, which is why it is important that the tool is easy to understand without a long introduction. The goal is that a user knowing cellpack and the principal problems can easily start analyzing the data and parameter space without a previous explanation of the additional analysis tool created in this project.

2.4 data

The data will be generated using cellpack and the analysis approach described in the publication [10] of the developers. Here, one parameter set is run thousand times with different random seeds. If the uniform distribution is achieved, each ingredient is packed into each subarea with the same frequency. For the analysis using the new work-flow , many models with different parameter sets will be computed, running each set with the same but different seeds.
The real data would be 3D complex objects as proteins. In order to simplify the exploration and speed up the computation for the use-case of exploring the distribution the data can be simplified.

2.5 data reduction

The first abstraction, that is done here is to use simpler ingredients (circles) and pack them on a 2D plane instead of a 3D volume.
The resulting model is then pre-analyzed using binning: The packing area is split in to several sub-areas (9 in the current case). For each of these sub-areas, the number of ingredients packed inside is counted. This could either be done for each ingredient individually or in total for all ingredients. If the ingredients are equally distributed in the area, the number of occurrences within each sub-area should be approximately equal, resulting in a bar chart of bars having all the same height. Figure 3 shows this binning step on the abstracted data for a good model and for a bad one.

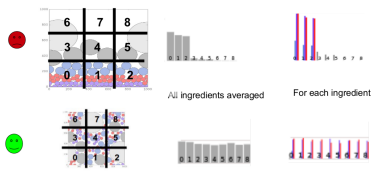

Figure 3: The upper example shows a bad model, it is easy to see here, that all the circles packed to the bottom of the rectangle, one ingredient type after the other. The bars of the bar charts are much higher for the areas at the bottom. Areas with id 6,7 and 8 have a count of 0, as each circle is count for the sub-area in which its centre is located. The lower example shows a result of a good model. Here the circles are well spread over the whole rectangle, also mixing different ingredient types. The resulting bar charts approximately have the same height for all bars. In the third column each ingredient is count individually, which makes visible, that there were more red and purple circles packed than grey ones.

3 Related Work

A lot of ideas for the prototypes and the final implementation were inspired by [12], which was written to serve vis classes. A lot of important visualization techniques, data types, different representations and interesting tools are summarized here. Additionally the content of the visualization class [11] in which this project was created also had an impact on proposed prototypes as well as on the final tool.

3.1 similar visualization tools

The data analyzed is a huge number of models presented through 2D images, that need to be compared and analyzed . This is a very complicated task and also worked a lot on in the vis domain. One application, that combines many images is called average explorer [15]. Here the users task is to explore a lot of similar images doing user-guided clustering and user-guided alignment. To fulfill this, a weighted average of images is presented. By selection of specific areas and using well known techniques as strokes, brushes and wraps, the user can influence the weights that are used to create the averaged image and thereby change the presented image interactively. The idea of this tool is used for the density plots described later, as these also show an average over multiple images.
Another tool that uses a quite similar approach is called fluid explorer and presented in [7]. It is also available as a plug in for Maya [1]. The model that is computed here is an explosion. Setting different parameters, users can change the appearance and behaviour over time. A similar approach, also using inverse design is used here. Instead of setting the parameters, waiting and looking at the output, the user gets a presentation of different output flames. The selection of the best flame can now be done based on the appearance of the final model. A big difference in the data domain is, that the flame is time varying, as it is a short movie clip similar of an explosion, while the data in this project is just static images and for the simplified version used for analysis only 2D.

For a more abstract prototype, a similar idea as presented in [13] was created. Here derived metrics as distances between the distribution of a model to the desired uniform distribution have been used. As this approach was not really well understood by the user, it was not considered in further steps of the project.

4 Approach

4.1 prototypes

In order to find a good interface, several paper prototype using different approaches were used. The different ideas were all based on showing clustered data, having a window where you could see an impression of all clusters, similar to a web-shop (Figure 4, where you have a matrix showing you all possible items that fit your actual search. Additionally all prototypes had the possibility to select a cluster and get some details about it (following the overview first, details on demand idea), which can be compared to a web-shop again, when clicking on an item and getting detailed information as customer feedback and ratings. In some prototypes there was a different view for the comparison of clusters, which was combined with the detail view in the end, to keep the interface simple and safe some screen space. The differences between the prototypes were mainly in the overview and filtering view. Some were using more abstract things as Bubbles (inspired by [13], Figure 5), a hierarchical clustering (Figure 6 approach or a simple average of all clusters.

4.2 decision

After presenting the prototypes in class and to the user, it came out that the easiest approach would be probably the best for the first implementation to make the user familiar with the new work-flow . After the first implementation the principles stayed the same but some graphs had to be changed, as they did not scale and show the desired information. A major change was done in the view presenting the values for active parameters inside a cluster. The old version is presented in Figure 7 and the improved one in Figure 8.

Another major change was to integrate the input view in the same window as the analysis view. The selection of active parameters is now in the same window as the further analysis, instead of having

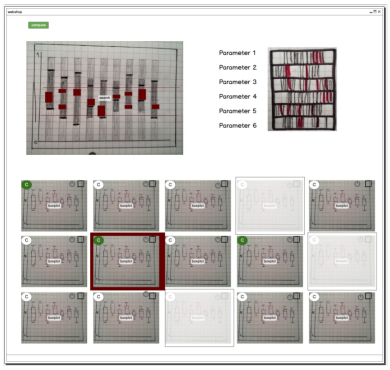

Figure 4: This approach is very similar to the implemented one, which will be described in more detail in the the results section. On the upper left, an overview is given, enabling the user to filter the clusters, which get transparent if they are filtered. On the upper right the parameter view, as described in Figure 7 is visible.

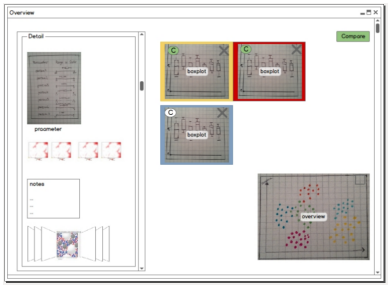

Figure 5: Here the overview uses derived metrics. Each point in the lower right graph presents one cluster or sub cluster of a cluster. The size of the point shows the number of models (the root would be one big bubble). The vertical position of a point indicates the distance to the optimal uniform distribution, horizontally, the difference to the other points decides the position. On the left side, a detail view is given. At the top box plots show the distribution of parameter values assigned to that cluster, subcluster or run. Below, density plots as in the final implementation are given for different ingredient types. Additionally a text box for notes and some of the results are shown.

different windows and the switching between them. This helps to remember what was selected without switching view and therefore follows the eyes over memory.
Additionally some redundant views were removed to save some screen space.

5 Implementation

The final implementation is an Electron web-app (prior named Atom Shell) [4] based on Node. It can also be used in the Browser running a python server locally or later an online server for the data computation. The frontend and graphics are implemented using d3 [6] which is based on javascript and svg elements inside an HTML interface. For the density plots d3 was not fast enough, which is why they are pre computed using matplotlib [5] and stored as images which as well as the resulting models.
For the computation of the data, the cellpack software, createing the data to be examined is used. Cellpack is written in python and uses several python libraries. For the interaction between the GUI and the backend, a python server is running in the background waiting for a request to compute and preprocess the data (discretize, binning, clustering, density plots).
In the future the tool might be used as a client-server software,

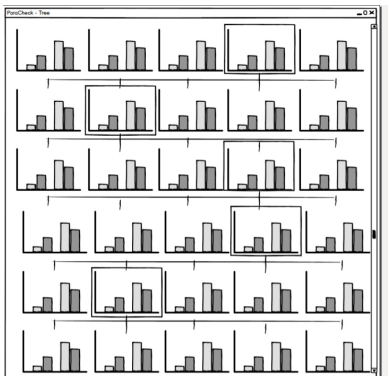

Figure 6: This prototype is very simple and could be used in combination with the others. Hierarchical clustering was used to generate a tree, by clicking on one of the clusters in a generation, the children of that cluster are opened. The user can continue this process until a specific level of accuracy is reached and switch to a detailed view then.

Figure 7: Shows horizontally the sampling range and in grey vertical lines all the values that occurred in sampling the parameter range. The coloured vertical lines represent values achieved in the clusters. A selected cluster gets a colour assigned and parameter values that were used for models assigned to that cluster get coloured accordingly. As you can see here, it is very hard to detect, which lines have which colour and therefore are related to which cluster. Additionally because of the colour soup, it is hard to see any patterns for the different coloured bars.

having the server running and people working from anywhere else. This would help to keep the data in a single place and reload some interesting previous computations. Additionally users could then easily also work on data sets created by other users and get an idea about their results.
At the moment data is stored in simple .csv and .json files. At a later point, this will be exchanged by a database, which will further improve the software.

6 Results

The interface is composed of multiple views. In the following each of these views is described an their interacts with each other, as well as used methods are explained.

6.1 input view

location and size: The view is located at the lower left of the screen and takes about 20% of the screen.

functionality: This view is the starting point for the work-flow . A user selects default values for all cellpack parameters or uses the predefined ones, selects interesting parameters to be active and specifies ranges them. The parameters are grouped according to their behaviour, using the same grouping as in cellpack, to help the users finding them in their known place.
Parameters in the packing group are general settings for the sampling like the granularity of the grid.
Ingredients settings influence the packing of the ingredients (here circles). These include a packing priority, deciding the order in

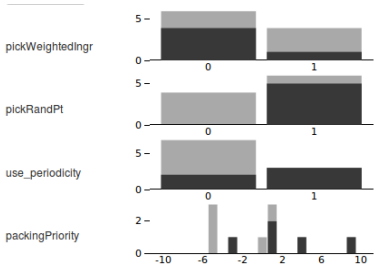

Figure 8: Shows a bar chart for each active parameter. The grey bars represent the values used in the sampling for all selected clusters (now not a single colour is assigned to a selected cluster, all selected clusters get grey), hovering over a cluster marks it black the parameter values assigned to this cluster also get black. Compared to the old representation it is clear here, which values occured in the hovered cluster and in which relation this is to other selected clusters.

which circles are placed and how often an ingredient tries to find an empty place in the grid until it gives up and does not pack itself. This group globally affects all ingredients.
The same settings can be set for each ingredient type individually (for the grey, the red, the blue,.. circle) using the last 5 drop-down options which have the same parameters as the ingredient parameter group. Beneath the cellpack parameters, settings for the actual running can be set. These include the number of runs, the number of seeds per run and the number of clusters that should be computed. Figure 9 and 10 show the input view in different stages.

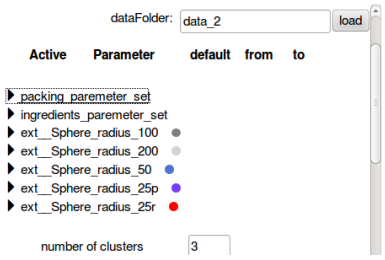

Figure 9: Here all the cluster groups are collapsed. At the top the user can specify the output folder to store the actual data in. Clicking on a group opens it and makes parameters inside that group visible.

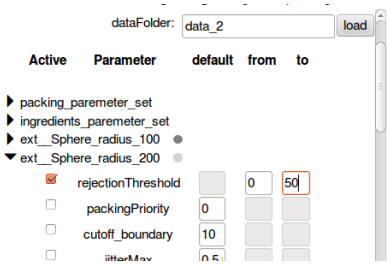

Figure 10: ext_Sphere_radius_200 was opened. The parameters for this ingredient type can now be selected as active and a range can be specified. Here "rejectionThreshold" was set to active in a sampling range from 0 to 50. "packingpriority" was set to the default value 0.

6.2 overview and search view

location and size: The view is located at the top centre of the screen and takes about 10% of the screen.

functionality: This part is used to get a general overview of the whole data-set. It shows a bar chart, summarizing all computed models (all clusters). The user has the option to move the two horizontal lines to filter out clusters that have a bar higher or lower than the value specified by the position of the line. This can help the user to filter out clusters that combine bad models, as they will have single bars much higher than the others. A potential problem that might occur here is that the average could be influenced enormously by some outliers, which could give a completely wrong representation for the overall summary. Another problem that occurred, when using the software with real data was the fact that there are some ingredients (here the big grey circle) that has a very small number of repetitions over the packing are. For this reason, the bars for this single ingredient will always be much shorter than those for ingredients having a higher repetition (e.g. red and purple circles). Additionally it is not clear exactly at the moment on which value the filtering should be applied to, as the user has the option to switch between the summarized view and the bar charts showing the ingredients individually. This decision will be made, once the users have used the software and can decide whether it works for them or not. Figure 11 and 12 show the overview view in under different settings.

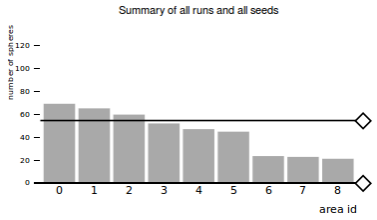

Figure 11: The view is set to show the sum of all ingredients, the filter was moved down to filter out uninteresting clusters.

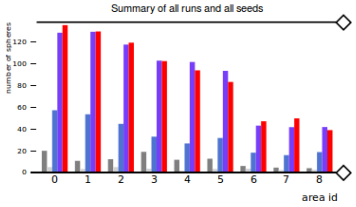

Figure 12: The view is set to show the each ingredient individually, the filter is at the initial position, meaning no clusters should be filtered out. Here it is also visible, that the grey bars are nearly unvisible compared to the red, blue and purple ones.

6.3 cluster representation view

location and size: The view is located at the bottom centre of the screen and takes about 15% of the screen.

functionality: Here for each cluster the average is presented in a bar chart. As for the overview it is possible to select the sum of all ingredients or the version showing individual bars for each ingredient. If a cluster got filtered out from the overview, it gets transparent. An important aspect here is that filtered clusters do not just move but stay at their position. This should help the user to see which cluster fades out at position of the filter and enables a selection of filtered clusters as well. At some points it might be useful to see parameters that created a bad result and combine them to parameter sets creating good results. A negative aspect in this

view is that the average is shown without giving an idea about the variance inside a cluster. An easy approach here is the showing MIN/MAX option, which draws two additional horizontal lines for each bar representing the minimum and maximum value for that area count inside a cluster. This is also an aspect, that will be improved later. Another important part, not modelled ideally now is the number of models that are inside each cluster (the size of a cluster), which is indicated by the number in the upper right corner at the moment. This might be exchanged by a different presentation in future changes of the interface.
If the user hovers over a cluster it gets a black border box. Clicking on a cluster creates a grey border box, opens the detailed representation of the cluster and shows the parameter values for that cluster in the parameter view. To deselect a cluster it can be clicked again, which will remove the cluster form the detail and parameter view.
Hovering over a selected cluster still marks it black but as it is opened in the parameter and the detail view on that status, the information there also gets black to link the different views together.
Figure 13 and 14 show the cluster view in under different settings and at different exploration stages.

Figure 13: Showing the count for all ingredients, Cluster1 and Cluster2 were filtered.

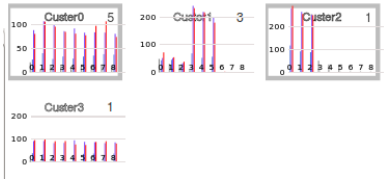

Figure 14: Showing the count for each ingredient, Cluster0 and Cluster2 have been selected.

6.4 detail view

location and size: The view is located at the left of the screen and takes about 35% of the screen.

functionality: This view is only opened for selected clusters. For each selected cluster one column is drawn. The first 5 images show density plots for each ingredient type. They are created by overlaying all models assigned to this cluster and filtering different ingredients to see some patterns that might get lost through the binning step and the representation used in the cluster view and overview. After the density plots, a sample of one image inside that cluster is given. Clicking on the image loads a new model of that cluster. The interaction here is not optimized and there are probably better ways to do this. As this is a rare use-case, the focus was directed on other parts for now and this was planned for later.
Below the result, a note box is appended, enabling the user to make some notes or comments about observations for a cluster. This not only helps the user to recognize clusters easily and remember findings that were made but also gives the possibility for different users to work on the same data set and share their observations.
At the moment the opacity for the density plots is set to a default value, to better see the different runs in the images, it will be necessary to set the opacity based on the number of runs assigned to a cluster or the pixel that has the highest number of circles overlayed. Figure 15 shows the detail view for 4 clusters in the selection. Some patterns are visible, as for the third column specific ingredients types (specific circles) always packed at different positions.

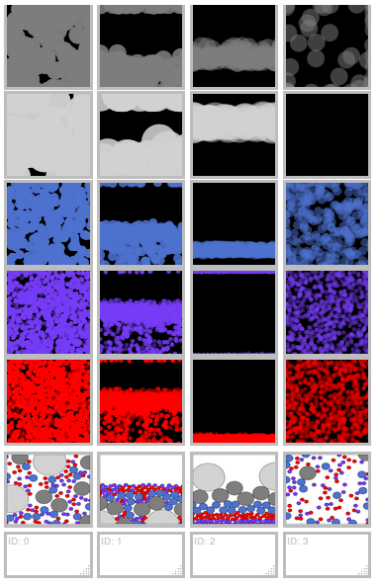

Figure 15: 4 clusters have been selected. The left column has the best result, the last but one has the worst results packing all ingredient types always at the same vertical position. The opacity problem is visible, as in the first column, too many plots were overlayed, leading to no visible opacity. Compared to the last column, where one can see the difference between more and less frequently packed areas.

6.5 parameter view

location and size: The view is located at the top left of the screen and takes about 15% of the screen.

functionality: This view also only draws the representation for clusters that are in the selection. For each active parameter (represented in one row) a histogram is drawn, with an x-axis representing the specified sampling range. On the vertical axis, the number how often a parameter had this value in the selected subset of clusters is marked. As most of the variables are integers, there are no problems caused by binning floating point values. If a cluster is in the selection and additionally hovered, the parameter values according to that cluster are highlighted in black, to see which part of the total counts in the selected cluster subset is related to the specific cluster hovered at the moment. Figure 16 shows a selection of some clusters and additionally the highlighting of a hovered cluster.

6.6 view settings

location and size: The view is located at the top right and can be opened by a click on the button, it only takes screen space when clicked.

functionality: This view is not an important part of the analysis. Here the user simply has the option to show or hide some additional information as the minimum and the maximum in the cluster representation. Changing between sum and individual ingredient count is also done here. To give more flexibility, a user

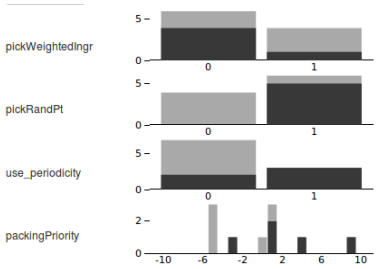

Figure 16: The cluster hovered over has the parametervalues marked in black, the values for all other selected values are displayed in grey. ”pickRandPt” was always set to 1 for the hovered cluster, and set to 0 in most of the other clusters in the selection. ”packingPriority” had the values -5, -3, 0, 1, 4 and 9 in the whole subset and -1, 2, 4 and 9 for the additionally hovered cluster

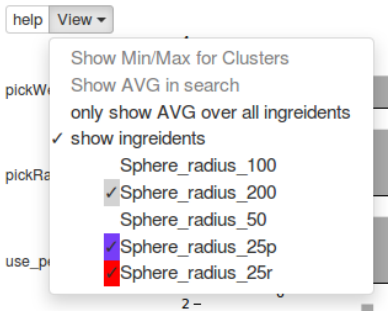

Figure 17: The dropdown is opened. The view is set to show each ingredient individually and to hide Sphere_radius_100 and Sphere_radius_50.

can select or deselect different ingredients, which might be helpful if the focus is on one specific ingredient at the moment.

In the following, some examples of the whole screen are given (Figure 18 and 19) to see the interaction between the views at different steps of the analysis work-flow . To link the different views, a hovered cluster gets highlighted in black in all views in which it is opened. Of course the filter affects the cluster representation by making filtered clusters transparent. The selection of a cluster causes the detail and parameter view to change. The input represents the loaded data set and enables starting a new run, which would cause all other views to change, as a new data set is loaded.

6.7 Performance

As described in other sections, the performance is a main problem at this point of the project. The system is interactive only for a small amount of data and does not scale for the desired complexity. As d3 [6] in general is fast and interactive, there will be some changes made on data loading and complex graphs will be pre-computed in python unsing matplotlib.
Another part that takes a lot of time is the computation of the actual data, which is done by cellpack. As this is the step in the work-flow that is planned to be done over night, it is not a critical point at the moment. It could also be speeded up enormously by using parallel systems, as each of the runs itself is independent and could be computed on a different system.

6.8 Feedback

Multiple presentations in class and meetings with the user brought helpful feedback and also improved the overall design of the system. Some major points are outlined below.

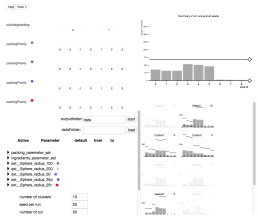

Figure 18: In this screen shot, the user already computed a data set, which is stored in the folder ”data”. The parameter groups are all collapsed, if they were opened, the user could see the specified default values and ranges selected to create this data set. In the parameter view it is visible that ”pickWeightedIngr” and ”packingPriority” for different ingredients (the relation to the ingredient type is given by the accordingly coloured circle next to the each label, no circle shows that this was a global setting) have been selected as active parameters. ”PickWeightedIngr” is a boolean parameter and only has the options for 0 and 1, the packingPriority is an integer variable and has a range from -3 to 3 for all ingredient types. The user already started the work-flow and moved the upper filter line down, which faded out Cluster0, Cluster4, Cluster5 and Cluster7. For the cluster representation, the MIN/MAX lines are shown. No cluster is selected at the moment, which leads to an empty detail and parameter view.

1. Presentations of the low fidelity prototypes to the user brought the decision to go for the simplest design, as it was clear and understandable. More complex approaches using a lot of derived attributes were not easy to explain to the user, which might be related to not fitting his mental model of the data.

2. After a presentation of the high fidelity prototype, another main aspect was changed. Before the colour channel was used to link the different views concerning clusters. Each selected cluster got a colour assigned which should create the connection of detail, parameter and cluster representation view. The proposed solution was to use colour to encode different ingredient types, as they already have colours assigned in the output images and only use grey values to show the connection of different views, with the additional hovering option marking clusters black in all views.

3. A very important change, that was partly initialized by feedback in class was the change of the parameter view. The old implementation using the ”bar-code” did not show the information in a way easy to understand. It also did not scale for larger data sets and did not help to find the desired patterns in the parameter values (Figure 7 and 8).

4. The idea of including the input view in the analysis window was a big change, that was proposed at a late stage of the problem, as the input view was neglected in the first prototype iterations.

5. Feedback that might be included in later steps is to use some derived metrics once the users are familiar with the new work-flow. This will for example include the distance between objects. As already mentioned, in the first iteration the goal was to keep it simple and understandable for the user, which is why these metrics are not used in the current visualization.

7 Discussion

This section summarizes the strength and weaknesses of the domain and shortly mentions the most important problems that were faced during the project. A lot of details will be added later, when the software was tested by the users.

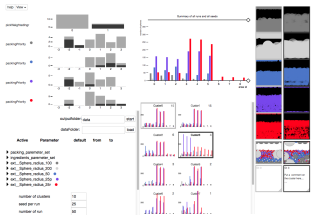

Figure 19: As in Figure 18 the data set is already loaded and the input groups are all collapsed. The same data set as for Figure 18 is presented here. The user changed the view to show bars per ingredient, which now presents that red circles were packed more frequently than grey ones. The filter does not set any constraints, which is why all the clusters are visible. Cluster0 and Cluster3 were selected by the user and Cluster3 is hovered additionally. In the detail view, the two selected clusters are visible, Cluster3 is also highlighted by hovering. For Cluster3 the user put a note in the text box at the bottom of the column in the detail view. Some differences between the clusters can be analyzed comparing the different density plots for each ingredient. Looking at the parameter-view, is is visible that for Cluster3, "pickWeightedIngr" was always set to true. "PackingPriority" always was negative for the dark grey circle, positive for the blue ones and at different levels for the purple and the red circle.

7.1 strength and weaknesses

As the project is not finished at the moment, there are still some things that need to be changed before the tool can be handed over to the user. Some of the strength and weaknesses can be analyzed simply by using knowledge about visualization and usability in general. Some other aspects will be remarkable only after the tool has been used in reality by the final users and domain experts.
A big disadvantage at the moment is the limitation to 2D. Of course this was a discretisation step in simplifying the data even before the implementation started, but it is still a restriction of the tool that should be changed in the future to also enable the exploration of 3D data.
Another problem at the moment is the scalability. As the density plots are all drawn using d3 [6] at the moment, the interaction is not given anymore for larger data sets. Before the tool will be handed over to the users, this plots will be pre-computed using python so in the interactive work-flow steps simply the images have to be loaded which will be much faster.
The last major weakness to mention here is the storage format. At the moment, the resulting models are stored as json files and png images. The clusters in folders holding multiple csv files about some statistics and the indices of related models and parameter combinations. In the future it will be helpful to use a database instead, which will also increase the performance of the system and enable an easier handling of different data sets.
As some weaknesses have been discussed in detail, some positive aspect should be pointed out as well. The main strength of the tool is the easy representations that were chosen. At the moment, the goal was to keep the interface simple, all the graphics shown are kept very basic and are well known representations for the users, already used by them in previous work and their own publication [10]. Whether or not it is really easy and intuitively to use can only be proved after the software has been used in reality, which will be one of the next milestones in the project.
The second strength is the use of inverse design and the new work-flow which should increase the efficiency of analysis. A more detail discussion about the real improvements made will also be added after the system has been tested.

7.2 lessons learned

One of the major challenges in the project was the communication to the final users and developers of cellpack[10]. As the time difference of 9 hours was not always easy to handle, the users located in San Fancisco also have been very busy at some steps of the project. As mentioned in [14] one important part is to get the real data as soon as possible. In this project the data could not just be handed over but needed to be generated using a script to interact with cellpack. As using someone else's code and run it on your own machines always requires some time and solving some issues, this should be done much earlier if the project was started again.
Another pitfall also mentioned in [14] was to find out who the real users are. At first the focused user was Graham Johnson, during the last month, finding out, that a PhD student also was involved in the development and knew a lot about the system brought the idea to also involve him in the project. Starting to get in contact with him helped a lot to better understand the system and get a script that could simply be used for data generation. Additionally it was a good idea to also invite him to the meetings and get some feedback on the visualization from him.
Other important aspects learned was the high importance to involve the user as much as possible and get feedback of others and also other people as often as possible. Especially when creating a visualization tool, one falls into the problem of knowing everything too good. This causes some distortions, for the developer of the visualization tool everything seems to be clear and obvious, until he or she has to explain what is happening and what should be seen on the screen to someone else.
If the users are not available all the time, simply trying to explain any person what is going on helps a lot in changing the perspective from a developers viewpoint to the one of a user, who does not know what is happening behind each click and where on the screen the desired information will be displayed.

8 Future Work

As the project is not finished at the moment, there are some planned milestones. A major goal is to improve the performance of the system. Additionally a good representation to show the variance for each cluster needs to be found, as well as other possibilities to show the size of a cluster. Once this is done, the tool will be handed over to the users and tested by them.
After this step, the final evaluation process can be started and written down. Additionally some changes, based on the users' comments will be made to the interface and tested again.
For later steps and next iterations, some ideas already showed up. These include to expand the use-cases to also cover other ingredient types (e.g. ellipses, rectangles or concave objects). After further iterations, it might be possible to expand and also support the analysis of 3D models, which will require a lot of changes to the interface. And new prototyping.
If the users are familiar with the overall work-flow to set ranges instead of values for their parameter and the approach of inverse design, it might be possible to embed some new views, presenting some derived metrics. This could further increase the efficiency of their work-flow , as less computations would have to be done in the users brain.

9 Conclusion

To sum this all up, a new tool was created, that should improve a users work flow targeting to find out the influence of input parameters on an output model. To fulfill this, the approach of inverse design is used. The analysis is started at the output, comparing different results and finding those that fit best. For the best results, the used input parameters can be used to compute better output in the future and improve the system generating the models by getting to know the influence of the parameters much better. As the tool has

not been tested by the end users at the moment, it is not possible to make any statements about the improvement of the users' work flow or new insights that were made by using the visualization software. These will be included in future work.

ACKNOWLEDGEMENTS

The author wishes to thank the visualization class of university of Vienna in 2015 and especially Torsten Möller and Thomas Torsney-Weir, the teachers, for the helpful feedback and support. Additionally important feedback was given by Graham Johnson and Ludovic Autin, the future users of the tool and developers of cellpack .

REFERENCES

[1] http://www.autodesk.com/products/maya/overview urldate = 2015-06-27.

[2] https://www.blender.org/ urldate = 2015-06-27.

[3] http://www.maxon.net/en/home.html urldate = 2015-06-26.

[4] http://electron.atom.io/.

[5] http://matplotlib.org/.

[6] M. Bostock. Data driven documents. http://d3js.org/ urldate = 2015-06-27, 2015.

[7] S. Bruckner and T. M"oller. Result-driven exploration of simulation parameter spaces for visual effects design. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1467–1475, Oct. 2010.

[8] D. M. Coffey, C.-L. Lin, A. G. Erdman, and D. F. Keefe. Design by dragging: An interface for creative forward and inverse design with simulation ensembles. *IEEE Trans. Vis. Comput. Graph.*, 19(12):2783–2791, 2013.

[9] D. S. Goodsell. *The Machinery of Life*. Copernicus, 233 Springer Street, New York, 2 edition, 2009.

[10] G. T. Johnson, L. Autin, M. Al-Alusi, D. S. Goodsell, M. F. Sanner, and A. J. Olson. cellpack: a virtual mesoscope to model and visualize structural systems biology. *Nature Methods*, 12(1):85–91, Dec. 2014.

[11] T. Mo"ller and T. Torsney-Weir. http://vda.univie.ac.at/Teaching/Vis/15s/schedule.html urldate = 2015-06-27.

[12] T. Munzner and E. Maguire. *Visualization analysis and design*. AK Peters visualization series. CRC Press, Boca Raton, FL, 2015.

[13] H. Rosling. http://www.ted.com/talks/hans_rosling_shows_the_best_stats_you_ve_ever_seen urldate = 2015-06-26.

[14] M. Sedlmair, M. Meyer, and T. Munzner. Design Study Methodology: Reflections from the Trenches and the Stacks. *IEEE Trans. Visualization and Computer Graphics (Proc. InfoVis)*, 18(12):2431–2440, 2012.

[15] J.-Y. Zhu, Y. J. Lee, and A. A. Efros. Averageexplorer: Interactive exploration and alignment of visual data collections. *ACM Transactions on Graphics (SIGGRAPH 2014)*, 33(4), 2014.