– Metamorphers –
A Formalization of Transitions in Illustrative Molecular Visualization

A, B, and C

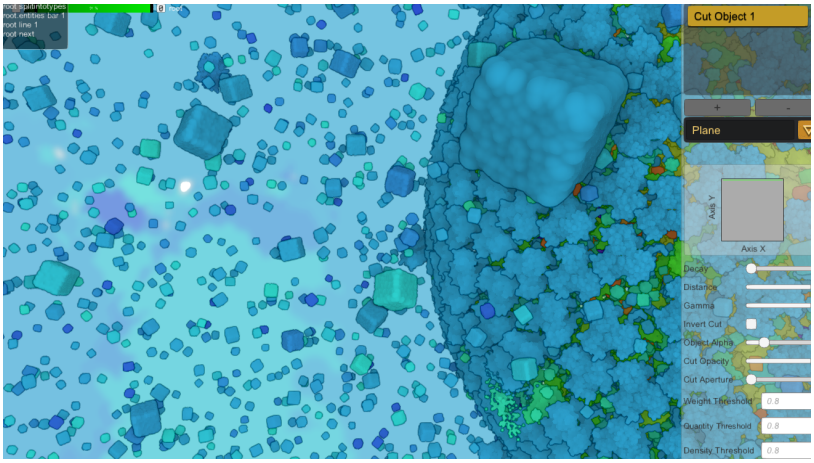

Fig. 1. tease me, then please me

Abstract— Illustrators in molecular biology often use animated transitions as a means to communicate complex phenomena. While these animations are popular, they still have to be created manually in 3D modelling software. We therefore provide illustrators with a more streamlined way for creating such transitions between various representations of molecular data.
We propose a formalization of the process of specifying these data representations in the form of a conceptual pipeline as our primary contribution. The pipeline consists of eight stages that can be grouped by their high-level functionality: target state representation, transition definition, and story helpers. As a result, data representations are specified in such a way that they form a continuous space. This space can be arbitrarily sampled to create a transition between the representations, e.g., in the form of animation or small multiples.
Our secondary contribution is a demonstration of our pipeline based on a proof of concept implementation. We created multiple operators for each pipeline stage that can be combined to achieve different types of transitions for arbitrary molecular data sets. One of these transitions that we designed in collaborations with illustrators, is a novel illustrative rendering approach for automatic schematization of mesoscopic data. In a qualitative discussion of our results, we received positive feedback from domain experts in illustration.

Index Terms—Molecular Visualization, Illustrative Rendering, Animation.

1 Introduction

The traditional approach for the visual communication of scientific insights is handled by illustrators and animators. Their goal is to attract the interest of broad audiences, and to convey complex phenomena in an accessible way. In recent years, we have seen an increase in the communication of topics from molecular biology **cite[drew berry, gj, david bolinsky, Gal McGill, janett iwasa]**, such as the splitting of DNA, the transport of oxygen in the the blood stream, or the comparison of molecular structures within a cell.

Illustrators have access to different narrative elements that can be combined with each other in order to convey their stories. The element of temporal sequence is used to convey changes of data states over time. The element of visual transformation creates different representations of the data in order to convey otherwise hidden aspects.

Visual transformation is necessary, because the molecular data data is complex. [explain complex data here or within the following examples?]Illustrators employ occlusion handling techniques, such as exploded views, to reveal the inner structures of a dense cellular data set. They employ focus & context techniques, such as the simplification of unimportant details of millions of atoms, in order to enable the presentation and comparison of cellular structures. While these two examples correspond to a transformation within the same visualization space, the depiction of abstract quantities or relationships, e.g., by bar charts or line graphs, correspond to the transformation to a different visualization space altogether.

For a successful knowledge transfer, viewers have to understand the relations between the original and the transformed data. How hard these relations are to grasp depends on how strongly an illustrative representation differs from the original data, how familiar the audience is with the presented data, and how complex the original data is (the number of objects that need to be related to each other, the amount of occlusion). It is therefore helpful for the user's understanding, to convey the transition, or *metamorphosis*, from one representation state to the other.

Animations are a powerful tool for conveying these metamor-

- *Author A is with Institute X. E-mail: a@x.com.*
- *Author B is with Institute Y. E-mail: b@Y.com.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org.
Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx/

phoses, since they are capable of displaying them in a continuous way. The continuous presentation supports an intuitive understanding of the relationship between representation states. The presentation is self-explanatory and enjoyable to audiences [11]. Still, animations are also critically regarded by the visualization community. Tversky describes the fleetingness of the displayed information and the potential sensory overload and distracting nature of badly designed animations.

Alternative solutions for conveying transitions alleviate the fleetingness of the displayed information by sacrificing the continuous aspect of the representation. Examples are narrative sequences of small multiples or static images that supplement the missing information of the continuity with glyphs. While solving the problem of fleetingness, they have to be well designed to be understandable.

These approaches are most suited for circumstances where little to no interaction is possible or desired, such as in presentation scenarios. In scenarios where interaction is possible or desired, other approaches, such as coordinated multiple view (CMV) visualizations can be used.

However, no matter which medium is used for the illustrative depiction of these stories, these transformed representations and the transitions to them have to be modeled manually by the illustrators which is a laborious and time-consuming task. The authors build their visualizations with complex molecular models, e.g., of a cell, consisting of millions of atoms that form tens of thousands of molecules, that are loaded into the scene of a 3D modeling tool. Considering that new discoveries are made frequently in molecular biology, these representations and their transitions have to be re-modeled when new insights are found, which makes the effort for maintaining them even higher.

We therefore want to assist illustrators by presenting our primary contribution: a formal description of a uniform method for creating target representations for spatial biological data sets, as well as the continuous transitions to these representations - describing the complete metamorphosis of the data representation, so to say.

The output of this method is a continuous space between two representation states. This space can be arbitrarily sampled to create a transition between these representations, e.g., continuously, in the form of an animation, or sparsely, in the form of small multiples.

The method is described by a pipeline that consists of eight stages that can be grouped by their high-level functionality: target state representation, transition definition, and story helpers. Each stage supports operators, so called metamorphers, that are responsible for one aspect of the metamorphosis.

As our secondary contribution, we demonstrate a proof-of-concept implementation of this pipeline based on three use cases that we developed in collaboration with illustrators. One of these results, a novel illustrative rendering approach for automatic schematization of mesoscopic data, is our third and final contribution.

2 Related Work

Animation and transitions between visual and data representations have been employed in many contexts. They serve as powerful tools for the dissemination of complex relations in space, time and abstract dimensions. They are frequently used in visual story telling and for the depiction of correspondences between data representations as well as visual representations.

2.1 Visual story telling

Kosara and Mackinlay [7] note that for a long time the focus of visualization research was on exploration and analysis but that presentation of findings especially using elements of story telling should be a research focus of equal importance. They list prominent examples of story telling that include animated transitions for trend analysis and dissemination.

Robertson et al. [11] have looked at trend visualizations in depth. Three methods including animated transitions were analyzed. They found that animated transitions of data was reported to be more enjoyable and exciting to users. They also found that it was significantly faster when used for presentation but less exact and less effective for analyzing data. These results encouraged us to use animated transitions to facilitate presentation of complex molecular data.

Segel and Heer [12] analyze story telling with narrative visualization and point out different approaches. They distinguish between author- and reader-driven elements in a narrative and speculate that "exploration of transitions between them presents an exciting area for researchers and practitioners". Along these lines Wohlfart and Hauser [15] presented a guided interactive volume visualization approach. Their system enables authoring and editing of visualization stories that give the user partial control over the exploration.

Ma et al. [10] describe the need of new scientific visualization methods that can be used for scientific story telling. Among other issues they suggest that novel transitions between different coordinate systems are needed to address the overwhelming complexity involved in today's scientific story telling.

2.2 Transitions between visual representations

Kosara et al. [8] use 3D scatter plot matrices to establish a link between physical layout and the abstract dimensions of the data. Their work is an early attempt to connect information visualization and scientific visualization using points as data primitives.

Guilmaine et al. [3] compare different animated transitions of tree structures. They find that hierarchical animation is better suited for tracking of changes. Basch [1] describes possibilities for animated transitions between volumetric rendering and abstract views like histograms and scatter plots. Heer and Robertson [4] show animated transitions between different visual representations of statistical data. These works make use of staging and staggering which are techniques that reduces occlusions and clutter. Our approach is based on these findings and formalizes animated transitions between spatial and abstract attributes.

Huron et al. [5] present the visual sedimentation metaphor for the animation of data streams. This metaphor allows to transition from discrete visual elements to a continuous representation. We employ a similar metaphor for the transition of spatial objects to abstract charts.

Exploded views [2, 9, 6] have been used in different contexts as a technique for illustrating spatial or hierarchical relations of parts of an object. They are frequently used in animations or interactive systems and reported to be effective in the dissemination of complex layouts. Our approach is similar to exploded views, since it animates the decomposition of the object into its parts. However, the spatial displacement is not only driven by the visibility of the individual parts, but also by additional data attributes.

2.3 Alternatives to animated transitions

Small multiples [13], coordinated multiple views, and static images in conjunction with annotations, traces and glyphs are alternatives that are typically used to visualize transitions, trends, correspondences or sequences [11]. Tversky et al. [14] summarize cognitive studies on the benefits of animation. Although, they conclude that animation alone has not been convincingly demonstrated to be superior to static illustrations, other findings of their study suggest a direction for research on animations in visualization. They report that animation together with basic interaction methods like pausing, partial re-playing, zooming and change of perspective might be the key to enhance the effectiveness of animation. Further, they acknowledge that the alternatives to animation (such as sequences of static illustration) are surprisingly hard to design and are therefore not an easy target for computer automation.

3 Background

[here we give more detail about the stuff that we couldnt talk about in more detail in the introduction && that is **background information** essential for understanding/following the rest of the paper.]

3.1 animated transitions in molecular biology

for what kind of knowledge transfer are they used (examples)? why are they essential?

(todo: outsource to Graham) basically ask him to tell us in his words (justify & motivate), why animations such as in his award win-

ning video are such a great tool for communication scientific insights. (doesnt matter if its verbose, we can shorten it.)

and also ask him to write a bit about the background of his field. what his tasks as illustrator & animator are, who are his customers / users, where does he get his data from, what does the data look like, what do his users/customers want to see?

to convey their message, illustrators apply a combination of the three high level tasks of occlusion handling, abstraction, and simplification on their data. subsequently, many pipeline stages are concerned with fulfilling these tasks with their respective measures. these can, in combination, then enable other lower level tasks, such as, comparison,.. **[check brehmer task taxonomy]**

3.1.1 molecular/mesoscale data

todo: describe cellpack format (outsource to mathieu)

[our data format:] this is the data that our pipeline operates on.

- atom: represents the smallest part of our data. represented by a unit sphere. always represented within a molecule alongside with other atoms. has information about its chemical element and its local position within a molecule.
- molecule: a single molecule. groups together a bunch of atoms. has inherent spatial attributes (pos/rot/scale) and an attribute vector (type). all molecules in the scene are instances of their respective type. the entities that our pipeline operates on. molecules can be hierarchically grouped to form subsets within a hierarchical scene graph.
- subset: is a group of entities/molecules. grouped according so specified attributes (spatial, by type, ..)

data is densely packed. depending on each molecules function, they can have a noisy distribution like in a liquid? or they can build rigid structures such as membranes. these attributes have to be taken into account when designing a target representation as well as the transition into it.

4 The Pipeline

4.1 Overview

our pipeline takes a supplied molecular data set in the described format and produces a user specified number of frames that show a transition from the provided original data to a specified target representation.

Our pipeline consists of eight stages that can be grouped into three high level tasks:

the first three stages can be parametrized to specify the target representation.

the next two stages can be used for guiding the narrative in the transition, i.e., by providing annotations and camera guidance.

the final three stages are used to define the transition from the original representation to the target representation.

each stage supports certain types of operators. operators basically determine how a stage proceeds with the supplied data.

they have to be implemented for each stage in consideration of desired output that the respective stage should produce. here is where design considerations come into play.

in order to achieve a desired result, the respective operators have to be selected and parametrized for each stage.

we give a discussion of our proof of concept operators in the next section.

depending on what kind of transition should be achieved, not all stages are mandatory.

also, the pipeline can be repeated for more complex stories or multi stage transitions.

in the following we will discuss each stage in terms of its function within the pipeline and in terms of how it operates on the supplied data.

- Target Representation Creation: Data Partitioning, Morphing & Blending, Layouting
- Narrative Support: Annotations & Glyphs, Navigation
- Transition Specification: Trajectory, Timing, Sampling

4.2 Target State

the visual appearance of the final representation is defined in these stages.

4.2.1 Partitioning Stage

as the first stage in our pipeline, the partitioning stage is concerned with preparing the molecular data for further processing in the consecutive stages.

illustrators partition their data to decide, in which granularity/level of detail the data is handled, e.g., which parts should be treated as a unit when transforming them to a different representation. **[concrete example?]**

as such this stage partitions the raw unstructured data into subsets according to specified criteria. these subsets are the building blocks that the final representation is constructed from.

the criteria for structuring them are specified by the respective operators of this stage.

the operators of this stage can be called hierarchically to create a scenegraph of the data. a user could at first partition the scene spatially and then partition the resulting subsets by molecule type.

as such, operators implemented for this stage will receive a list of entities or list of subsets as input. the operator will then group the elements of the supplied list according to specified criteria, such as the elements' molecule type, position, or spatial proximity to a defined point. more precisely this means, that a number of subsets is created, e.g, as many as there are molecule types or defined spatial quadrants, and each entity that fits the respective subset is assigned to it as its child. each child entity also stores the id of its designated parent. the output of an operator will therefore be a list of the newly created subsets with the respective child / parent assignments.

4.2.2 Blending/Morphing

illustrators can use the operators of this stage to change the visual representation of parts or all of their data. this can serve multiple purposes / the three high level tasks of simplification, abstraction, and occlusion handling.

entities or subsets change their individual or collective appearance, in order to convey specific information by revealing, simplifying or transforming the molecular data. general: individual or subsets can change their representation: blend into each other into a new collective shape, change individual shape, blend into texture, change individual or collective render mode

concrete examples: simplification: specific molecules could be blended to a 2D texture, that represents the molecule's kekule diagram. unnecessary details can be reduced by blurring the complex noisy distribution of molecules within a compartment to a uniform color. occlusion handling: by changing visual properties, such as transparency, of certain parts of the data, other parts can be revealed. abstraction/transformation/?: the molecules can be morphed into tetris-like blocks that the layouting stage then combines to form, e.g., a histogram bar, encoding the total volume of the respecitve molecule type.

operators thus can work on arbitrary levels of the scene hierarchy that was created in the previous pipeline stage: on the leafes (the individual molecules): on the subsets (the grouped molecules): histogram, giant cubes

depending on their implementation, operators on this stage can operate on the any part of the shader pipeline, i.e., move around individual vertices that are the atoms of a molecule or do post processing on the rendered image in a separate rendering pass. **[need some feedback for improvement of the input/output description]**

4.2.3 Layouting

this is the tool that illustrators use to spatially re-arrange the data that they partitioned and transformed/blended in the previous stages. this spatial re-arrangement of the molecular data can serve various purposes. layouts can be used for occlusion handling, e.g., by exploding the dense data to reveal the inner workings of a cell. they can be used to transform the data into an altogether different visualization space, such as a line graph or a network in order to convey some more abstract points. or they can use them to simplify the complex and dense data in order to steer the viewer's attention towards some specific anatomical details that would otherwise remain hidden.

as such, the input for a layout operator is always a node in the scene hierarchy. for each child of the supplied node, the operator then calculates a new position and rotation - depending on what kind of layout the respective operator is implementing. the output of a layout operator is therefore a new position and rotation for each child of the supplied node. each entity has a vector of control points that stores positions and rotations and is initialized with the element's original position and rotation. the new values calculated by the layout operators are added to this vector.

layouts can be stacked upon each other - meaning they are called multiple times for the same node - this creates additional control points that can be used to transition the data into intermediate representations before arriving at the final state. the final representation is determined by the last layout that has been called. a user could, for instance, first call an explosion layout, to reveal the inner parts of the dense data set, before rearranging these parts into their final form.

layouts can be called for each level in the scene hierarchy. depending on the intended target representation. lowest level nodes contain leaves (molecule instances). here the layout works directly on repositioning the individual molecules, e.g., it could put them into an abstract collective shape like a histogram bar. the next highest level node would then contain a list of children where each is layouted as a bar. the layout operator could then position these subsets (bars) with respect to each other.

4.3 Story Helpers

[actually conceptually, these two stages are parallel to the data partitioning stage - as they create two additional data types (besides the molecular data): annotations and camera control points - that are also handled by the subsequent pipeline stages.]

4.3.1 Annotations

This stage allows the illustrator to add various forms of annotations to the scene.

Illustrators can use these annotations for multiple purposes: Simple text labels could assist story telling by describing what is currently happening in the scene. Icons that point to specific events can be used to steer the viewer's attention. These elements can also be combined to create frames of reference, such as coordinate axes, scales, and legends, that can improve the viewers' orientation within and their understanding of the presented information.

Operators of this stage should therefore, in the simplest case, accept text input and should be able to create or load icons and other graphical representations. Annotations can be logically attached to components or can be free-floating.

[I guess the positioning & lifetime of these annotations should be handled (conceptually) by the layouting & timing stages? should we then move this stage to before the layouting stage?]

4.3.2 Guided Navigation

The guided navigation stage enables control over camera parameters.

Illustrators can use this stage for two purposes. They can steer attention towards specific events or details in the scene, in a guided fly through manner. And they can use it to follow the transition, if it leaves the current view frustum.

Operators in this stage modify the camera position & look-at vector. As such, they create a list of control points for these two values. A timing and trajectory is then also defined in the respective stages for the interpolation between these control points.

4.4 Transition

These stages are responsible for defining the transition to the target representation, i.e., how each entity reaches its target position and when it reaches it's target form.

4.4.1 Trajectory

This stage describes for each molecule, how it reaches the target position and the control points in-between that have been defined in the layouting stage. This is in the simplest case a linear interpolation between a molecule's control points. However, if applied wisely, illustrators can use this stage for multiple purposes: simplification: they can make the transition more readable and visually engaging, by bringing structure into its elements' motion paths, e.g., by creating a well positioned point in space for each molecule type through which it has to "travel" to reach its target position. this structuring can serve occlusion handling, e.g., by placing these transit points in a way that the molecules flying through them reveal essential parts of the scene. information generation: they can also encode topological information that was not present in the original data, e.g., in the form of meaningful data highways that transport molecules from one state to another, or even to simulate the flow of blood between two organs.

to achieve this structuring, operators in this stage can implement techniques such as edge bundling. this means that an operator can introduce new controlpoints into the motionpath of a molecule, e.g., to re-direct it so that it does not occlude the existing scene.

as such, this stage, takes a subset from the scene hierarchy as input. the child elements of this subset are analyzed in terms of their controlpoints (at the very least original and target position) and their relation to the controlpoints (original and target position) of the remaining elements of this subset (or even the entire scene). depending on the operator type, new controlpoints that fulfill one of the above described functions are created as the output of the stage.

4.4.2 Timing

the timing stage is responsible for the temporal coordination of the transition by determining the order and speed at which entities or subsets reach their target position and visual form.

as such, this stage is the illustrators' canvas in the temporal dimension.

while the inherent purpose of this stage is to encode chronological information, i.e., the sequence of events, illustrators can also use it for additional purposes: they can make the transition more readable & visually engaging by reducing the number of simultaneous visual stimuli, i.e., by starting the transition of indicisual molecules / subsets with a time offset to each other, thus creating staged transitions/animations [cite heer?]. timing can also be used for occlusion handling, i.e., by first starting the transition of elements that occlude important structures or that are closer to the viewing position, while other elements remain in their positions until they are revealed.

further, timing can also be used for controlling visual stimuli. the transition can be timed in a way that the attention is steered to a currently important structure or event by animating what should be in focus while keeping the remaining scene still.

as such, operators of this stage, depending on the intent that they implement, create a time curve for each entity that determines the start time and speed of the spatial and visual transition. optionally they can also create an end time, i.e., determine, when an element should cease to exist but also, when the transition should stop, i.e., whether an element should reach its final destination as well as how close it should get. this can be used to create a target representation that looks like it stopped in mid-motion.

4.4.3 Sampling

This final stage determines the sampling strategy, i.e., at which rate the transition is sampled.

The operator, that the illustrator chooses, therefore determines, whether the transition will be continuous or if sparser sampling will be applied, e.g., to extract a narrative sequence like in a comic strip. The lowest rate would therefore just output the final image of the target representation.

While the previous stages of the pipeline can be called multiple times in order to create more complex animations/transitions/stories, the sampling stage is final, as it ties together the information from the previous stages to sample the visual result.

Before the sampling starts, the scene hierarchy is flattened again - which means that all information from higher level nodes is projected into the leaves, i.e., the individual molecules. The transformation matrices of higher level nodes are multiplied recursively with all their lower level children for each controlpoint - so that each molecule knows about its global position for each control point.

The sampling stage then uses the output of the timing stage to determine at which rate/speed to interpolate between control points for each molecule. And it uses the output of the trajectory stage to determine how to interpolate between the supplied control points, e.g., linear, cubic,..

The sampling stage can also determine if the transition is pre-baked or sampled on the fly in real-time.

As such, this makes it possible to use a continuous pre-baked transition to create, for instance, a single result image that encodes the missing transition information within glyphs. These glyphs can be calculated from the continuous pre-baked information.

The continuously sampled information could also be exploited to create a hybrid visualization that displays a discretely sampled narrative sequence of small multiples. Upon clicking one of these small multiples, it could show the continuous transition to the next discrete element in the narrative sequence.

5 Proof Of Concept Implementation

our PoC implementation allows illustrators to generate illustrative representations of their data, as well as coordinating/authoring the transitions to these altered representations.

we designed two types of representations and created the components that our pipeline requires to produce them for arbitrary molecular data sets.

the components for each stage can be parametrized and queued based on a simple scripting interface.

we will first discuss the representations that we designed and then go into detail concerning the components that are involved in creating them.

5.1 Supported Representations

5.1.1 Explosion of Molecular Structures

5.1.2 Schematization of Molecular Structures

This representation was motivated by illustrative drawing, like the ones that can be found in medical/??? books.

simplification (schematization) : simplify data to create understanding. transformation within the same visualization space. resembles the illustrations that we see in biomedical textbooks. immediate transition = more intuitive to follow. tries to preserve anatomical attributes but in a simplified / schematic way. tries to give overview of elements in the data and possibly their interrelations.

preserves: shape/size of compartments, maybe individual elements
sacrifices: low level detail for high level information

5.1.3 Representation of Volumetric Relations

This representation was inspired by **[cite graham video (he actually mentioned a video where something like this is happening)]**. XXX created this animation to give a better understanding of what kind of proteins are represented in the different compartments of a cell, as well as how their collective mass relates to each other.

what is important in these representations (preserve vs sacrifice vs put into the foreground)

derivation (infographic) : transformation to another visualization space. tries to convey information that an anatomic representation cannot. derive data to convey hidden insights. immediate transition = harder to follow. sacrifices high level shapes (e.g. of compartments or other structures) for low level information (details on the data such as count of instances). insight into hidden relations in the data.

does not care about these aspects in the data (preserve): todo
cares about these aspects in the data (sacrifice): todo

5.2 PoC Pipeline

[contains specifics about our pipeline implementation] implemented in cellVIEW: give some details about cellVIEW): unity, supports loading &rendering of cellPACK data, renders molecules as point clouds

desc was jeder component beisteuert um die beiden reps zu erzielen

5.3 PoC components

[describes PoC components that we implemented per pipeline stage] [should we describe beforehand what (which representations) we try to achieve with these implementations?]

5.3.1 Partitioning

spatial subselection by clipping plane/object,
splitting into types,
splitting into type ranges
cloning
—not showcased but implemented: *spatial splitting operations into equal parts (pie splitter),

5.3.2 Annotations

text labels, icons (textures), maybe: data representatives (for each type), optional addon to each operator: connectors between annotation & assigned subset

5.3.3 Blending / Morphing

blur into collective shape, morph,
—not showcased but implemented: blending to texture
—other not implemented & not used examples: unfolding

5.3.4 Layouting

bar, line, translate, rotate, scale, schematization layout = combi of scale & rotate but selection of elements where what op is applied is very specific
—not showcased but implemented: slice, sphere, circle

5.3.5 Camera / Guided Navi

[whole-data-in-view-keeper] update the camera pos & lookat to keep all molecules in the view frustum.
other examples:

5.3.6 Trajectory

linear: from A straight to B, bundling/transit points
—other not implemented & not used examples: curved

5.3.7 Timing

offset function: delay timing by distance to plane & staging by type ... anything else?
—other not implemented & not used examples: offset according to: inherent spatial parameters (position, distance to other entities, density)

5.3.8 Sampling

continuous sampling,
—other not implemented & not used examples: sparse (comic strip / narrative sequence),
post-processing operators: motion blur,

6 Results

in this Section, we discuss the outcome of our proof of concept implementations

6.1 Representation of Volumetric Relations

6.2 Schematization of Molecular Structures

things that should be preserved: depending on the usecase / representation - not all properties are essential for each representation form.

- instance level: position of instances
- type level: shape of types, relative size of types, color of types (?), number relation between types
- compartment level: shape/pos/size/color of compartments / structures
- global level: volumetric dimensions of the data
- todo: what else?

7 Discussion

7.1 additional outcomes

side products of our pipeline - free of charge / easily achievable with minimal additional effort: sequence: comic strips, static image with glyph

7.2 generalization

compatible data formats: point cloud, volume, polygonal?

compatible data types: dense hierarchical data with lots of repeating entities of a few dozen types

7.3 general discussion

tversky: congruence principle: the structure and content of an external representation should match the structure and content of the desired mental representation.

apprehension principle: the structure and content of the external representation should be readily and accurately perceived and comprehended.

7.4 occlusion

is of course a central factor that could hinder a user's understanding of a transition. can be dealt with at various stages of the pipeline. grouping: choose groups so that they dont interfere with each other (depending on how you plan to transform the data) TODO: come up with examples that describe which kind of grouping would be beneficial for which kind of target state e.g., group the data first into layers parallel to the viewing plane that basically peel the data layouts inner bar: within the bar occlusion does not matter, as long as the bar represents entities of the exact same type todo: other inner layouts & general outer general: the placement of entities/subests should be chosen so that they dont occlude each other from the viewing angle. maybe we should not distinguish between inner and outer layout since this kind of restricts us to having two layouts instead of n.

then again, we could argue that there can be n-outer layouts, and one inner layout morphing: same as inner layout: within the shape occlusion is not an issue

trajectory: the trajectories offer two of techniques for avoiding occlusion.

edge bundling: setting control points so that paths don't occlude each other todo: anything else?

timing: coordination of timing and speed of elements, e.g., staging

7.5 future work

implementation of comic strip / animation hybrid

8 Conclusion

Acknowledgments

The authors wish to thank A, B, C. This work was supported in part by a grant from XYZ.

References

[1] C. Basch. Animated transitions across multiple dimensions for volumetric data. Master's thesis, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria, oct 2011.

[2] S. Bruckner and M. E. Gröller. Exploded views for volume data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1077–1084, Sept. 2006.

[3] D. Guilmaine, C. Viau, and M. J. McGuffin. Hierarchically animated transitions in visualizations of tree structures. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 514–521, New York, NY, USA, 2012. ACM.

[4] J. Heer and G. Robertson. Animated transitions in statistical data graphics. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1240–1247, Nov 2007.

[5] S. Huron, R. Vuillemot, and J.-D. Fekete. Visual Sedimentation. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2446–2455, 2013.

[6] O. Karpenko, W. Li, N. Mitra, and M. Agrawala. Exploded view diagrams of mathematical surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1311–1318, Nov. 2010.

[7] R. Kosara and J. Mackinlay. Storytelling: The next step for visualization. *IEEE Computer*, 46:44–50, 2013.

[8] R. Kosara, G. N. Sahling, and H. Hauser. Linking scientific and information visualization with interactive 3d scatterplots. In *Proceedings of the 12th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*, pages 133–140, 2004.

[9] W. Li, M. Agrawala, B. Curless, and D. Salesin. Automated generation of interactive 3d exploded view diagrams. *ACM Transaction on Graphics*, 27(3):101:1–101:7, 2008.

[10] K. L. Ma, I. Liao, J. Frazier, H. Hauser, and H. N. Kostis. Scientific storytelling using visualization. *IEEE Computer Graphics and Applications*, 32(1):12–19, Jan 2012.

[11] G. Robertson, R. Fernandez, D. Fisher, B. Lee, and J. Stasko. Effectiveness of animation in trend visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1325–1332, Nov 2008.

[12] E. Segel and J. Heer. Narrative visualization: Telling stories with data. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1139–1148, Nov 2010.

[13] E. Tufte. *Envisioning Information*. Graphics Press, Cheshire, CT, USA, 1990.

[14] B. Tversky, J. B. Morrison, and M. Betrancourt. Animation: Can it facilitate? *International Journal of Human-Computer Studies*, 57(4):247–262, Oct. 2002.

[15] M. Wohlfart and H. Hauser. Story telling for presentation in volume visualization. In *Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization*, EUROVIS'07, pages 91–98. Eurographics Association, 2007.