IGScript: An Interaction Grammar for Scientific Data Presentation

Richen Liu ¹, Min Gao ¹, Shunlong Ye ¹, and Jiang Zhang ²

1) Nanjing Normal University
2) Peking University
Motivation

- Data story animation: reveal the dynamic changes and evolution processes

- Building data story animation is challenging: lack of tools on customizing animation contents, viewpoints, steps, styles of transitions, and shot changes

- Hard to customize set operator-based query expressions like “show (B∩(A∪C ∪D))” to generate a dynamic animations by using GUI or traditional interactions

- Interpretive grammars: applied to customize static visualizations or even dynamic data-story animations flexibly

- We propose **IGScript**, a script-driven (interpretive grammars) and data-driven tool to help users build scientific data presentation animations
Background and Related Work

- Three classification criteria
  - External Grammars (Ext) v.s. Internal Grammars (Int)
  - Textual Grammars (Tex) v.s. Graphical Grammars (Gra)
  - DSVL, DSML, DSEL

- Most existing related work focus on either the computation stage or the rendering stage. Few focus on the stages of interactive data presentations

- IGScript is more like a DSVL work while it focuses on grammars for interactive data presentation animation instead of visualization
Design Rationale

• Design Goals
  – G1: help users define ROIs (region-of-interests) via coarse-grained and fine-grained interactions
  – G2: form a presentation animation by recording visual traversals or visual tracking across ROIs
  – G3: enable users to edit the animation clips of a data story in a semantic space
Overview & ROI Definitions

• Design Overview
  – A linked view: a visualization space view and a coding space view, providing **visual steering/visual feedback** for ROI definitions

• ROI Defs.
  – Place/move an ROI box in the visualization space by coarse-grained tuning (dragging)
  – Adjust the box slightly in the coding space by fine-grained tuning (text edit)
Grammar Design (1/2)

- General-purpose Grammars
  - Define ROIs: `defineROI`
  - Data loading: `load`, like a C++ overloading function to support different data types in run-time
  - Camera lens transformation around the ROIs: `rotate`, `translate`, `scale` (zoom), `parallel`
  - Data story animation: `animate`, `locate`

```plaintext
1 load { // similar to "overloading" function in C++
2   // load a map-based 2-D scalar data
3     data(string dataFile); map(string mapFile);
4 }
5 // or load a 3-D scalar data (volume data)
6 volumeData(string dataFile); rgbaScheme(string tf); // transfer function
7 // or load a vector field data, e.g., "ocean"
8 vectorFieldData(string dataFile);
9 // or load a DTI data, e.g., "dti"
10 dtiData(string dataFile);
11 }

12 defineROI(roiID=int, roiName=string, at(float, float, float), size=(float, float, float));
13 locate {
14   roiArray=string[roiName#1, roiName#2, roiName #3, ...],
15   foreach {
16     duration=float seconds,
17     interval=float seconds
18   }
19 }

1 rotate { axis=string, angle=string, duration=float seconds };
2 rotate { axis=(float, float, float), angle=float degrees, duration=float seconds };
3 translate { to(float, float, float), duration=float seconds };
4 scale { factor=(float, float, float), duration=float seconds };
5 animate { speed=string }; // [low, moderate, high]
6 parallel { // executed concurrently
7   rotate { axis=(float, float, float), angle=float degrees, duration=float seconds };
8   scale { factor=(float, float, float), duration=float seconds };
9   translate { to(float, float, float), duration=float seconds };
10 };
```
Grammar Design (2/2)

- Application-specific Grammars
  - Generate data story animation for vector field data visualization and DTI data visualization
  - Vector field data visualization (OD query)
  - 3-D box queries and their arbitrary logic combinations for DTI fiber visualization

```cpp
1 trace {
2   mode=string, // [destination, origin, realtime]
3   clusteringAlg=[dbscan, kmeans, pca, ...]
4   lifetime=vectorT, // lifetime of traced fieldlines
5   colorOfCenterPathline=[c1, c2, ...]
6 }
7 halfMergeSplit { // overview-to-details
8   timeSlots=[overview|timeUnits, t2, t3, ...]
9 }
10 lineStyle {color=colorL, width=float}
11 tubeStyle {color=colorT, thickness=float}
```

// overloading "placeANewBox": assigned by an ROI
1 placeANewBox {boxID(string), at(roiName=string),
2   color=c, alpha=float};
3 // overloading "placeANewBox"
4 placeANewBox {boxID(string), at(float, float, float),
5   size=(float, float, float), color=c, alpha=float};
6 moveBox {boxID(string), to(float, float, float),
7   duration=float seconds};
8 scaleBox {boxID(string), factor(float, float, float),
9   duration=float seconds};
10 // code block "with": set local color and opacity
11 with {color=c, alpha=float} {
12   show (queryExpr); // e.g., show(not(A\B)\(C\cap D))
13   pause (float seconds);
14   show (queryExpr); // e.g., show((A\cap C)\(B\cap C))
15 };
```
Design Details: Implementations

- A compiler converts textual grammar codes into a data story animation.
- A code generator (decompiler) to translate the interactive data exploration animations back into the codes.
- IGScript makes the presentation animations editable, e.g., it allows to cut, copy, paste, append, or even delete some animation clips.

Camera tracking for pathline clusters (PCA components)
User Study

• Examine if users with limited programming skills could create their desired data presentation animations by IGScript.
• 14 participants: 8 doctors from different hospitals and 6 non-expert novice users with different majors
• Results
  – Users can define ROI easily (G1)
  – The generated animations are what users want (G2)
  – It is allowed to edit the small clips in an animation (cut, copy, paste, append, delete) (G3)
Case Study 01: 2-D Scalar Field Data

- 2-D scalar field: global carbon emission data

A 36-sec animation by the statement rotate

An animation with transitional scene switching across ROIs
Case Study 02: 3-D Scalar Field Data

- 3-D scalar field data: CT scanning volume data LUNG (top) and HAND (bottom)
- The transitional scene switching follows the shot change style in film photography

Video Figure

Snapshots of overview-to-details animations
Case Study 03: Time-varying Vector Field Data

- Time-varying vector field data: global ocean pathline data
- The camera splitting/merging strategy is customized by **halfMergeSplit**
- OD query animations: origin clustering (top) and the destination clustering (bottom)

Animation snapshots with four time-steps (#01, #10, #20, #25) for the ocean data
Case Study 04: Diffusion Tensor Imaging Data

Easy to reproduce a similar result to the work [14] (left) without changing any rendering codes

Complex expression-based queries for DTI fiber data
Discussion and Future Work

• Supported data currently: four general types of scientific data (“The Visualization Handbook” by Charles Hansen and Chris R. Johnson [28]):
  – 2-D/3D scalar field data (0th order tensors)
  – Vector field data (1st order tensors)
  – Tensor field data including DTI data (higher order tensors)

• Scalability on data types
  – It can be easily adapted to other use cases for these four types of data
  – Regarding a new data type, APIs on data loading and application-specific presentations (if any) should be added to the IGScripts library

• Suggestion by user study participants (future work)
  – Provide more sample codes
  – The identifiers and parameters could be redefined using their technical terms or closer to natural languages
  – Visualize the GUI flow charts for the textual codes
Conclusions

• We design an textual interaction grammar-based tool named *IGScript*
• A linked views, providing visual steering and visual feedback when customizing data presentation animations
• Script codes: look simple and follow the style of natural language-like grammars
  – A special-purpose compiler is designed to convert natural language-like grammar scripts into data presentation animations
  – A code generator (or a decompiler) is developed to translate the presentation steps into script codes.
• The animation clips are easy to be cut, copied, pasted, or even deleted
• Evaluations: a user study, four case studies and performance analysis demonstrate the usability and customizability of IGScript
Acknowledgement

• The reviewers for their valuable comments
• All the user study participants
• Funding
  – National Nature Science Foundation of China (61702271)
  – Postgraduate Research & Practice Innovation Program of Jiangsu Province (SJCX20_0445)
Thank You!