Contextual Interaction Support in 3D Worlds

Norman Murray
Centre for Virtual Environments and Future Media
University of Salford
Salford, UK
n.murray@salford.ac.uk

***Abstract*—With the increasing use of 3D displays and input devices we need to be sure that when 3D worlds are created that their users can easily learn how to operate within these 3D worlds. To do this we can provide the user with contextual interaction support within the environment. Within virtual worlds where you are free to move around and especially when you are immersed, trying to refer to a manual to ascertain your next course of action within the world would not be best received by the user. Instead of manuals separate from the computer system, the computer system should be able to interrogate itself to provide the user with information on what the system can do. For computer systems to be able to do this we need to move away from defining interaction using an event-based model to formally define the interaction dialogue. We have shown how by using Augmented Transition Networks (ATNs) you can allow the user to ask what they can do within the current context. The user can also query the system to see how they can perform a specific task. The help provided can also identify to the user the components within the environment that they need to interact with. Further work has begun to examine how the user could adapt the interaction within the system by visualising the ATNs.**

***Keywords*-Contextual interaction; virtual worlds;**

I. Introduction

3D user interfaces have been used for many years in several specific application domains such as virtual and augmented environments, computer aided design and visualisation. Social worlds such as Second Life and Active Worlds allow people to meet and socialise in computer generated 3D worlds. Gamers have begun to embrace 3D input as well, via Nintendo's Wii remote that gained a dominant position in the home console market. The success of the Wii and its 3D input device has led to the development of Kinect by Microsoft for the XBox360 and by Move for the Sony PlayStation 3 that were introduced last year and have gone on to sell many millions. The Nintendo 3DS includes a 3D screen without the need to wear glasses that has launched to large sales across the world. In the past few years, cinema has begin to reutilise 3D, with stereo versions of movies being released to draw in the masses with higher ticket prices increasing revenues. Mobile phones are also making use of 3D user interfaces and the inclusion of projectors within mobile phones could further stimulate their use for 3D worlds and interaction. People are now ready to embrace computer generated 3D virtual worlds and interact with them and their associated 3D user interfaces and devices.

Within virtual worlds, a user can move through the environment altering their view of the world and the components they can interact with. The increased degrees of freedom, interaction devices, and interaction styles available in virtual worlds compared to desktop interfaces increases the necessity for providing assistance to the user. Two ways that we can assist the user in the environment are by providing context aware assistance within the virtual world and by allowing the user to adapt the interactive behaviour of the world to suit them. To accomplish this we propose the development of a formal specification of the 3D interaction dialogue, that will be able to provide the user with contextual interaction support within the 3D world.

II. Related Work

Virtual reality offer users the possibility of interacting with a computer-simulated environment, real or imagined, in an intuitive and natural way. With the advances in hardware, we are now at a turning point in virtual environments as they become more commonplace, with large screen projection systems coming down in price and size, more accurate and wireless tracking systems and advanced input devices. The same is also true of desktop VE with the increasing screen sizes and 3D capabilities of personal computers, home consoles and mobile phones, leading to more people being able to access 3D content and virtual environments, whether this be for work, social interaction or pleasure. To enable the rapid uptake of these forms of virtual worlds we need to make sure that we provide users with easy to learn and use virtual worlds. We can do this by providing them with context aware environments.

A contextual aware environment should be able to respond to questions the user has about the current state of the human computer interaction. The user may ask questions such as what functionality is available to them in the current context. They may ask why they cannot perform some functions in the current state. They may also ask how they can move from their current state to being able to perform a given task. Such a system requires knowledge of the current context of the system and be able to determine how they can progress from the user's current state to the state the user wishes to be in to perform their task. Within desktop interfaces Palanque [1] has shown how context sensitive help can be achieved with interfaces where the interaction dialogue has been formally specified for WIMP (windows, icons, menus, pointer) style interfaces. Within virtual worlds, a user can move through the environment altering their view of the world and the components they can interact with. The increased degrees of freedom, interaction devices, and interaction styles available in virtual worlds compared to desktop interfaces increases the necessity for contextual help being provided to the user. A formal specification of the dialogue component of the virtual world would allow for the development of a contextually aware interface. Some initial work we undertook made use of augmented transition networks and task modelling to provide assistance in the programming and development of virtual worlds[2].

As interaction within virtual environments is becoming more complex, adaptable interfaces have been proposed to address this issue. Studies with WIMP style interfaces have shown that user performance is optimised and that participants greatly prefer adaptable interfaces[3], [4] over static or adaptive (self-adapting interfaces where their appearances change according to some algorithm). Current virtual environments offer very little support for adaptation. Adaptable interaction will allow the user to alter the interactive behaviour of the environment to suit them. By providing the user of the virtual environment with the ability to configure the style of 3D interactions to suit their own criteria, users' performance and satisfaction could be increased. Within an adaptable interface, the user will be able to choose between different styles of 3D interaction but they will also be able to manipulate and alter the interaction styles themselves to alter the method of operation of the interaction technique. To provide adaptability within the virtual world, we will visualise the formal specification of the interaction dialogue. Through this visualisation we will examine how we can support user adaptability and evaluate the benefits of providing adaptable interaction within virtual worlds.

Current systems tend to have been developed using an event-based model that makes altering the techniques difficult. Using an event-based model requires considerable ad hoc low level programming that makes it difficult to develop such systems and so makes it hard to share and reuse[5].

If we look back in history we can find that there was a very high cost associated with developing and maintaining WIMP interactive software. In 1992 it was found that the user interface composed 50% to 80% of the code of an application [6]. One explanation for this is that interactive systems are inherently more powerful than algorithmic systems, yet most interactive systems are implemented with algorithmic programming languages[7]. With the production of GUI toolkits for the generation of WIMP styles interfaces there has been a massive decrease in the cost of developing and maintaining such interfaces. Research into WIMP style user interface toolkits has been limited recently, but some recent work has highlighted the problems with event-based model that toolkits are based on [8]. Events are read in and dispatched by a top level loop and sent to widgets via a callback mechanism. This form of programming breaks the interaction into a set of smaller pieces called by the main program loop, independently of each other [9]. Programmers resort to using global variables to share states between the separately invoked pieces of code. This results in code that is hard to debug and maintain.

What is required is a move away from coding these techniques at a low level to being able to define the interaction at a higher, more abstract level using a formal specification. This will promote further research into interaction techniques, as the formal specification will support the development of new techniques from the components already present within the system and by allowing the system to be extended. We have previously developed a task based system[2] that has been used in the development of VEs for building construction[10] and engineering maintenance [11] that overcame the limitations of event-based systems and allows for easier modification of the 3D interaction dialogue.

The development of a visual representation of the formal specification of the 3D interaction dialogue will hopefully ease the development of the 3D user interface. It will allow developers to easily and rapidly prototype the interface so that it can be evaluated and remove the large burden of code development.

III. Formal Description of the Interaction Dialogue

The interaction between the user and the system takes place via a set of input devices that form the medium of communication. As the user interacts with the input devices, this interaction is transformed into input that the system can process. Interaction devices take the form of keyboards, mice, track-balls, touch screens, 3D trackers, etc.. The users perform tasks by interacting with the input devices to achieve a specific goal. The performing of a task usually involves a sequence of interaction cycles or subtasks, with each cycle contributing towards the completion of the task and so satisfying the users goal.

It is these tasks and how they break down into interaction cycles or subtasks that need to be modelled within the system. ATNs were chosen to store the task information for the underlying interaction architecture of the virtual environment. ATNs are built on the foundations of the Finite State Machine (FSA). An ATN consists of a collection of labelled states, with recognised events and transition arcs. Seen as an FSA, a program is a network of states or contexts. When an event occurs, the application processes the event, takes a transition to another state and waits for the next event. FSAs can be used to describe simple systems but as the complexity of the system increases this creates a combinatorial explosion in the number of states in the FSA. ATNs add the option of having a test/condition that must be satisfied before the transition can be taken. Also, during the transition, actions can be executed to achieve certain desired functionality. The events in the ATN correspond to the input device the user is using and the concept that was selected. This will result in the execution of some behaviour, an intermediate result and a new state.

Figure 1 shows a small part of an ATN for a filter flow Boolean query interface shown in Figure 2. This interface allows for the creation of filter flow queries [13]. The query in Figure 2 represents a query to find information on New York or California and driving but not RUVs. When the user selects to view the results, the query is passed to Google with the information being displayed in a web browser. The task shown in the ATN of Figure 1 involves the user editing one of the query text boxes as seen in Figure 2. The ATN begins in the *program start* state. To move to the *edit text node* state, the user would use the left mouse button to select a query text box. Once a box is selected they can press any key to edit the text in the box. This arc leads back to the same state: *edit text node*. To finish editing the query text they press return and are then moved back to the *program start* node of the ATN. This example will be used throughout the paper.

IV. Context Aware Support

With ATNs chosen as the method for formally specifying the interaction dialogue, a library was written to manage the tasks, subtasks and events. The library can be used to form an ATN for describing the interaction that occurs between the user and the application. Once the interaction has been specified using an ATN, context aware support can be added to the interface.

ATNs only have one active state. This makes it cumbersome to handle events that can be triggered at any time such as an *abort* or *help* action. To manage this we have a *help* ATN separate from the main program's ATN to manage actions such as requests for contextual help.

A. Textual Help

ATNs can handle a *help* event by analysing the current state of the ATN, and traverse the arcs of the ATN, moving from one state to another to produce appropriate responses to the user's current situation. Our initial help system displays what actions are available at the current state. A virtual world can be expansive and surrounds the user. With a WIMP display the user is presented with a screen that attempts to reflect the internal state of the application. Within a virtual world, the user can have a large world to navigate, what they can do in the world can depend on their location within it, the direction they are facing and the objects that are within their grasp. There needs to be clearer assistance to help them complete their task. The initial help system seen in Figure 3 relays to the user the current tasks that they can perform. It shows that the user is in the *edit text node* that was shown in Figure 1.

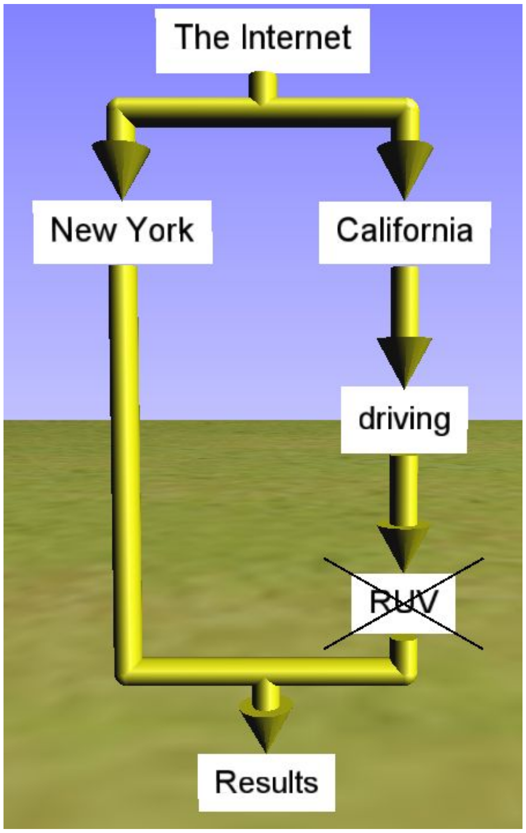
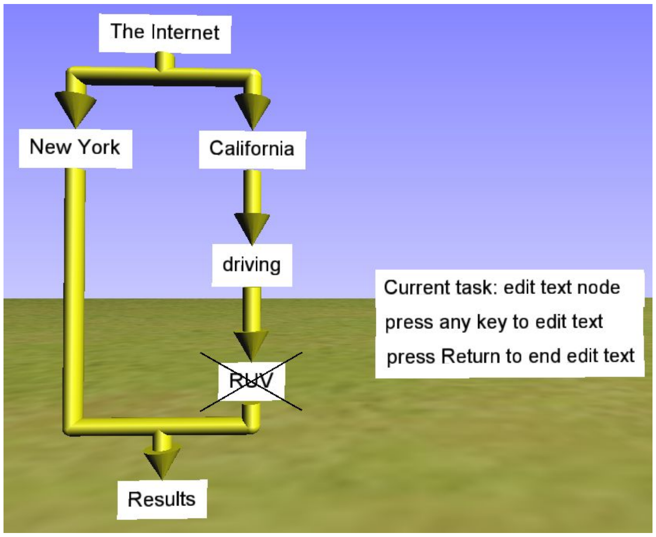


Figure 2. A filter flow Boolean query.



Current task: edit text node
press any key to edit text
press Return to end edit text

Figure 3. Help shown when editing a node.

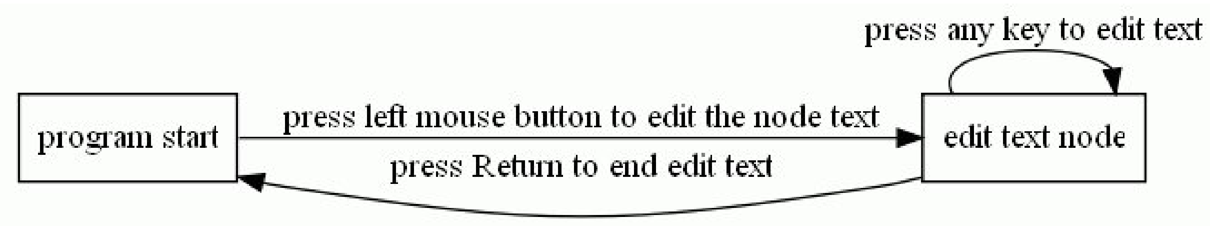

press any key to edit text
program start
press left mouse button to edit the node text
edit text node
press Return to end edit text

Figure 1. A subsection of the ATN for the filter flow Boolean query interface (created using graphviz[12].)

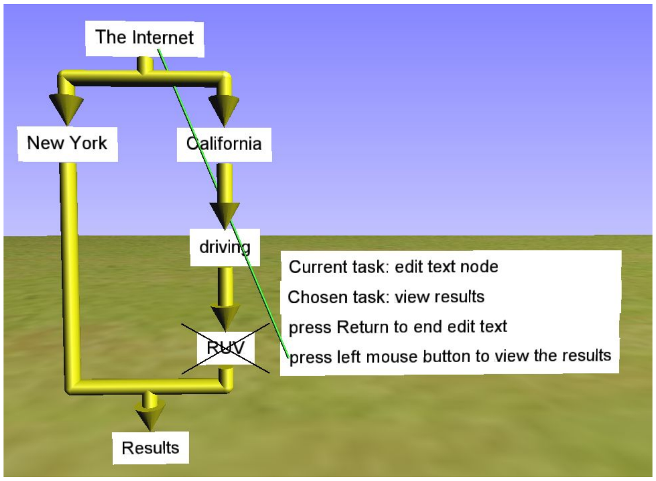
The text shown in the figure shows that there are only two actions that the user can perform. These are to press return to complete editing the text or to press any other keys to alter the text. This textual contextual information answers the user's question about what functionality is available to them in the current context. The two interaction sequences shown use the keyboard and does not require the user to select an object in the 3D world. When the interaction does require interaction with an object in the environment, this can be highlighted. Figure 4 shows a task that requires interaction with a component and to aid the user a line is drawn from the help to the component in the environment that they would need to interact with. This helps to link the contextual support directly with the components in the environment.

B. How do I?

Aside from offering contextual assistance within the user's current context, they may also wish to know what tasks they can perform and how do they get to a state so that they can perform these tasks. To do this we list the tasks that are available at the system. The user can then view the list of tasks. They can select a task from this list and obtain information on how they could reach the state within the program so that they can perform this task. To perform this function, a breadth first search is performed within the ATN to obtain how the user can move from their current state to the user's selected state. Once the route through the ATN from the current state to the user's selected state is calculated, we can traverse through the ATN and display the interactions required to perform the action on the screen. Figure 4 shows how the user can move from their current state of editing a text node to viewing the results of the query. The contextual information also highlights the components in the 3D world that the user needs to interact with to move to a state where they can perform their selected task.

V. Adaptable Interaction

The previous section looked at how we can provide contextual interaction support to the user within a 3D world. Another form of assistance that we could provide the user is to provide adaptable interaction. One way we could provide this is to visualise the ATN that is used to define the interaction dialogue. We could then allow the user to modify the ATN directly to provide adaptable interaction. Figure 5 shows part of the visualisation of the interaction dialogue of the filer flow query interface, being displayed within the virtual world.



Current task: edit text node
Chosen task: view results
press Return to end edit text
press left mouse button to view the results

Figure 4. Realtime contextual assistance.

Initial work has looked at allowing the users to modify specific keys that are used to perform tasks. Further work needs to be done to examine the switching in and out of larger tasks such as specific forms of navigation being altered within the ATN aswell as being able to alter the specific keys that are used to perform functions. The development of this is on going.

We propose to extend the formal specification developed for contextual help and adaptability within virtual worlds to allow for the visual development of 3D interactions. We will need to deconstruct 3D interaction techniques into reusable concepts, in a similar manner to our previous work on visual interfaces to databases[14]. These reusable concepts will form part of the 3D interaction formal specification that will form the basis of reusabilty.

The current system is implemented in C++ with OpenSG being used for 3D visualisation, and Graphviz being used to define the layout of the ATN for visualisation using OpenSG.

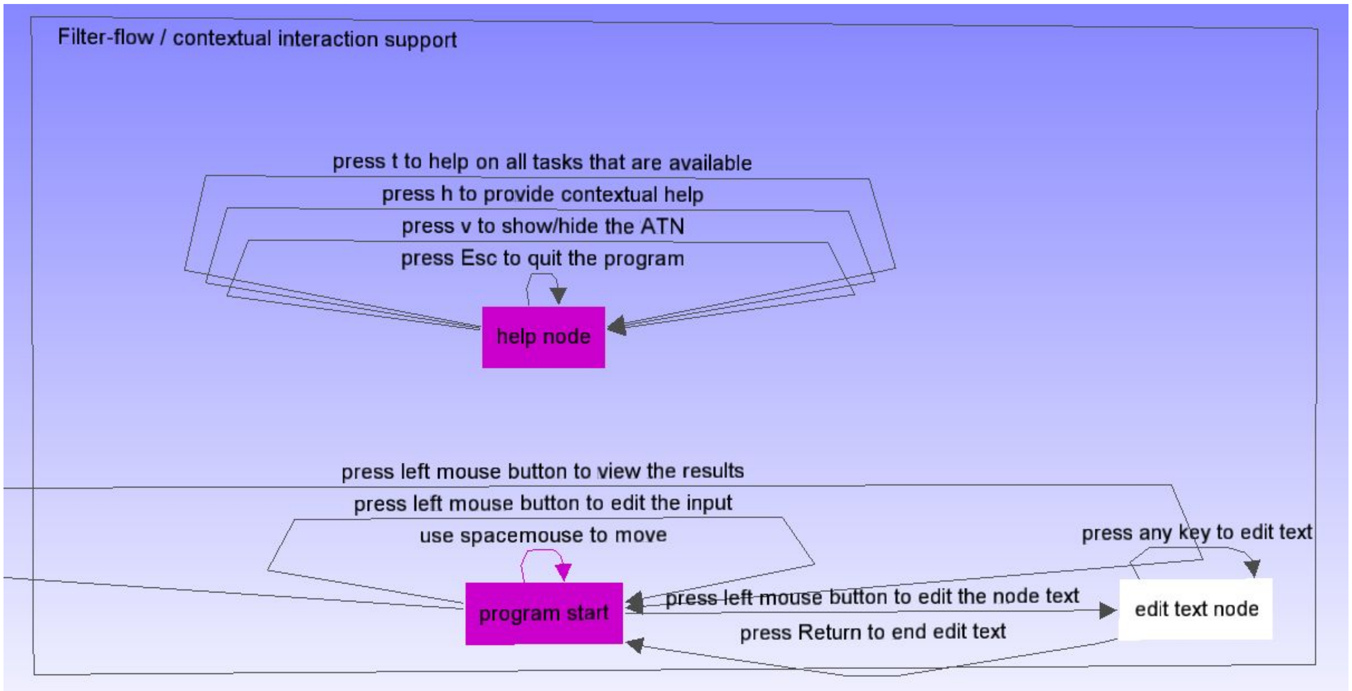

Filter-flow / contextual interaction support
press t to help on all tasks that are available
press h to provide contextual help
press v to show/hide the ATN
press Esc to quit the program
help node
press left mouse button to view the results
press left mouse button to edit the input
use spacemouse to move
press any key to edit text
program start
press left mouse button to edit the node text
edit text node
press Return to end edit text

Figure 5. Visualisation of a subpart of the ATN interaction dialogue for the filter flow query world.

VI. Conclusions

With the increasing use of 3D displays and input devices we need to be sure that when 3D worlds are created that their users can easily learn how to operate within these 3D worlds. To do this we can provide the user with a context aware environment. Many of todays current systems operate an event-based model that can be difficult to develop and cannot provide contextual help with users being forced to refer to textual help that is normally completely separate from the program. Within virtual worlds where you are free to move around and especially when you are immersed, trying to refer to a manual to ascertain your next course of action within the world would not be best received by the user.

Instead of manuals separate from the computer system, the computer system should be able to interrogate itself to provide the user with information on what the system can do. For computer systems to be able to do this we need to move away from defining interaction using an event-based model to formally defining the interaction dialogue and reap the benefits such a system can provide.

ATNs have been chosen to store the task information for the underlying interaction. We have shown how by using ATNs you can allow the user to ask what they can do within the current context. The user can also query the system to see how they can perform a specific task. The user can view the available tasks and from this select one and be told how they can move from their current state to a state where they can perform this task. The help provided can also identify to the user the components within the environment that they need to interact with.

We have also begun work to allow the user to adapt the interaction within the system. This has been performed by visualising the ATN and allowing the user to alter the keys that are used within the interaction.

Further work is going to look at providing more assistance to the user by allowing them to select a component in the environment and see what tasks are associated with that object. This will involve analysing the ATN and seeing where the component can be operated on within the ATN and so drawing up a list of the available tasks that can be performed on that object. We are also going to examine animating the help within the system. Instead of just specifying what actions the user needs to undertake, we can animate and actually perform those actions to further assist the user in learning how to operate within the 3D world.

Beyond providing support for the user operating within the virtual environment, we also aim to make it easier for the developer to create their 3D worlds. By providing developers with a visual interface where they can create, and modify the ATN such as the one visualised in Figure 5, they will be able to create their interactive systems much more easily and also create systems that assist the user in learning the system.

References

[1] P. Palanque, R. Bastide, and L. Dourte, “Contextual Help for Free with Formal Dialogue Design,” in *Proceedings of ”HCI*

International 93", 5th International Conference on Human-Computer Interaction joint with 9th Symposium on Human Interface, Aug. 1993.

[2] N. Murray and T. Fernando, "A Task Manager for Virtual Environments," in *Proceedings of the Workshop on User Centred Design and Implementation of Virtual Environments*, 1999.

[3] L. Findlater and J. McGrenere, "A comparison of static, adaptive, and adaptable menus," in *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI 2004)*, 2004.

[4] W. Stuerzlinger, O. Chapuis, D. Phillips, and N. Roussel, "User interface faades: towards fully adaptable user interfaces," in *Proceedings of the 19th annual ACM symposium on User interface software and technology*, 2006.

[5] R. Jacob, L. Deligiannidis, and S. Morrison, "A Software Model and Specification Language for Non-WIMP User Interfaces," *ACM Transactions on Computer Human Interaction*, vol. 6, no. 1, pp. 1–46, 1999.

[6] B. A. Myers and M. B. Rosson, "Survey on User Interface Programming," in *ACM Conference on Human Factors in Computing Systems, CHI'92*, 1992, pp. 211–220.

[7] P. Wegner, "Why Interaction is More Powerful Than Algorithms," *Communications of the ACM*, vol. 40, no. 5, pp. 80–91, 1997.

[8] M. Beaudouin-Lafon, "Designing Interaction, not Interfaces," in *Advanced Visual Interfaces*, 2004, pp. 15–22.

[9] B. A. Myers, "Separating application code from toolkits: eliminating the spaghetti of call-backs," in *Proceedings of the 4th annual ACM symposium on User interface software and technology*, 1991, pp. 211–220.

[10] N. Murray, T. Fernando, and G. Aouad, "A Virtual Environment for the Design Simulated Construction of Prefabricated Buildings," *Virtual Reality Journal*, vol. 6, no. 4, pp. 244–256, 2003.

[11] N. Murray and T. Fernando, "An Immersive Assembly and Maintenance Simulation Environment," in *The 8th IEEE International Symposium on Distributed Simulation and Real Time Applications*, 2004.

[12] E. R. Gansner and S. C. North, "An open graph visualization system and its applications to software engineering," *SOFTWARE - PRACTICE AND EXPERIENCE*, vol. 30, no. 11, pp. 1203–1233, 2000.

[13] D. Young and B. Shneiderman, "A Graphical Filter/Flow Representation of Boolean Queries: A Prototype Implementation and Evaluation," *Journal of the American Society for Information Science*, vol. 44, no. 6, pp. 327–339, 1993.

[14] N. Murray, C. Goble, and N. Paton, "A Framework for Describing Visual Interfaces to Databases," *Journal of Visual Languages and Computing*, vol. 9, no. 4, pp. 429–456, Aug. 1998.