Canvas 3D: Um novo componente de interface no *XView*

Perla Velasquez Alegre
Shin-Ting Wu

Grupo de Computação de Imagens
Departamento de Engenharia de Computação e Automação Industrial
Faculdade de Engenharia Elétrica
C.P. 6101, Unicamp
13081 - Campinas - SP
perla@dca.fee.unicamp.br
ting@dca.fee.unicamp.br

Abstract. This paper presents an strategy to extend the resources of the *XView Canvas* widget to support 3D-interactions and visualization.

1 Introdução

A diversidade de aplicações, por um lado, e de recursos computacionais, por outro, torna a elaboração de uma boa interface gráfica uma tarefa árdua e altamente complexa. Em decorrência disso, os sistemas de gerenciamento de janelas como *X Window* [**?**] e as ferramentas de interações (*toolkits*), baseados nos padrões de construção de interfaces MOTIF [**?**] e OPEN LOOK [**?**], estão recebendo grande aceitação em ambientes comerciais e de pesquisa, porque eles proporcionam uma plataforma mais simples para o desenvolvimento de uma interface gráfica aplicativa. Esta plataforma é constituída de um conjunto de componentes de interface denominados *widget* [1][**?**].

XView [2][**?**] é uma ferramenta de interações utilizada no desenvolvimento das interfaces de inúmeros projetos no meio universitário. Como ela não oferece nenhum recurso de interações e de visualização 3D, a maioria dos programas gráficos interativos necessita desenvolver adicionalmente as suas próprias rotinas de interação e de visualização. O fato dessas rotinas serem funcionalmente similares nos motivou a desenvolver um novo componente de interface que encapsule as principais **técnicas de interações e de visualização 3D**. Ele é denominado **Canvas-3D**.

Para garantir a modularidade e a portabilidade do novo componente de interface, as seguintes estratégias foram adotadas:

- modularização das técnicas de interações 2D e 3D;

- interface clara entre o conjunto de rotinas dependentes e independentes das ferramentas de interações;

- máxima utilização de convenções padronizadas e funções de domínio público.

A seção 2 apresenta os principais requisitos do **Canvas-3D**. Considerando esses requisitos foi concebido o módulo IQL (*Interactive Quick Line*)[3][**?**] para a ferramenta de interações PRODIA [**?**]. Na seção 3 será explicada a estratégia de integração do IQL no *XView*. Uma comparação entre o *Canvas* e o **Canvas-3D** é mostrada na seção 4. Na seção 5 será discutido o aspecto implementacional. E na seção 6 será traçada a linha de continuação deste trabalho.

2 Requisitos para o Canvas-3D

Em muitas aplicações a criação e a manipulação de modelos 3D desempenham um papel crucial. Para aumentar a eficiência dos usuários nestas tarefas numa tela de visualização (*screen*)[4], são amplamente utilizados mecanismos como grades (*grid*), atrações (*snap*) [**?**, **?**] e mostradores digitais. Na figura **??** são mostrados alguns tipos de grades que poderiam guiar melhor os usuários na criação de algumas entidades gráficas 2D. No caso de modelos 3D, uma das maiores preocupações é prover a noção de profundidade. Hagen salienta em [**?**] a necessidade de criarmos ambientes 3D visualmente "reais" para facilitar a sua interpretação. A percepção de ambientes 3D pode ser reforçada, através de:

- grades tridimensionais parametrizáveis[5];

- deslocamento de entidades gráficas na direção da linha de visão do observador (ou câmera);

[1] Elemento controlador ou exibidor (da informação dos objetos da aplicação) que encapsula geometria e comportamento.

[2] *X Window System based Visual/Integrated Environment for Workstations.*

[3] No Instituto de Computação Gráfica (IGD) da Sociedade de Fraunhofer Gesellschaft (FhG), na Alemanha.

[4] considerado aqui como um ambiente 3D fictício.

[5] Entenda-se ajustáveis.

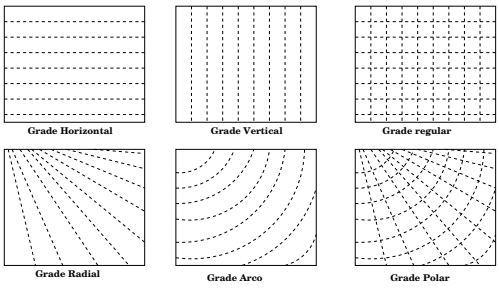

Figura 1: Alguns tipos de Grades

- rotação de entidades gráficas em torno dos eixos, que não sejam perpendiculares à tela de visualização;

- recursos de *rendering* sofisticados.

Paralelamente à percepção de profundidade, é desejável que as entidades gráficas sejam tridimensionais e semanticamente manipuláveis. Nos defrontamos aqui com dois problemas:

- estabelecimento de uma estratégia para diferenciar um número infinito de pontos projetados num ponto da tela de visualização, de forma a garantir uma interpretação não-ambígua das posições captadas pelos dispositivos de entrada;

- modelagem das relações entre as entidades gráficas.

Sob este aspecto pode-se classificar as rotinas que devem ser encapsuladas no componente **Canvas-3D** em:

- funções de suporte à percepção 3D;
- funções de suporte às interações 3D.

2.1 Módulo de Percepção 3D

O módulo procura simular um ambiente 3D. Ele reúne recursos que permitem a realização dessas funcionalidades (figura **??**):

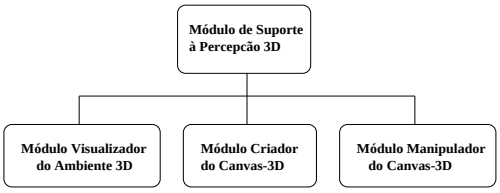

Figura 2: Módulo de Suporte à Percepção 3D

- Módulo Visualizador do Ambiente 3D;
- Módulo Criador do **Canvas-3D**;
- Módulo Manipulador do **Canvas-3D**.

O módulo visualizador é responsável pelas transformações, direta e inversa, dos modelos 2D ou 3D para as entidades gráficas visualizáveis. Quando os modelos forem 3D, as funções do módulo câmera são ativadas para projetar os objetos na tela de acordo com a posição do observador e com o tipo de projeção. O módulo também suporta funções que melhoram a percepção visual: grades e eixos 2D e 3D (figura **??**).

O módulo criador do **Canvas-3D** é responsável pela inicialização dos recursos associados a ele e o módulo manipulador pela sua edição. Entre os recursos associados ao **Canvas-3D** temos a câmera, os grades, os eixos e as entidades gráficas 3D.

2.2 Módulo de Interações 3D

Segundo a concepção do *XView*, o *Canvas* é uma área de desenho, que capta as informações de entrada do *mouse*. As entidades gráficas suportadas pelo *Xlib*[6] [**?**] podem ser visualizadas e as posições do *cursor* podem ser captadas. Visando a oferecer um ambiente de interação 3D mais flexível e considerando que as entidades gráficas 3D são na realidade projeções, precisaremos organizar as entidades visualizáveis num modelo geométrico mais complexo [**?**], de forma que:

- os relacionamentos entre as entidades visualizáveis, **pontos**, **curvas** e **superfícies**, possam ser agrupadas em estruturas mais complexas e manipuladas como um todo;

- as entidades gráficas projetadas sobre um mesmo ponto da tela de visualização possam ser distinguidas.

Outro problema relativo às interações 3D é estabelecer a correspondência entre todos os pontos na tela de visualização e o espaço 3D. Como soluções parciais, pode-se usar grades 3D para orientar melhor os usuários ou então dividir a tela em 6 regiões que correspondem aos seis semi-eixos ($+x, -x, +y, -y, +z$ e $-z$) [**?**, **?**]. Como o **Canvas-3D** deve incluir um recurso para dar suporte à estruturação das entidades gráficas, as funções organizadas nos seguintes

[6] *X library*, interface de programação do *X Window System*.

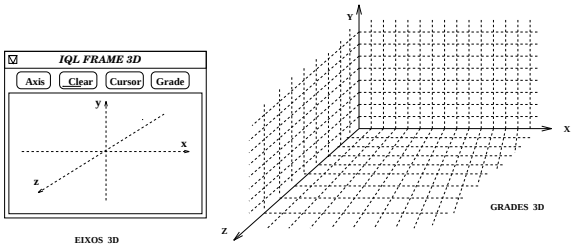

Figura 3: Eixos e Grades tridimensionais no **Canvas-3D**

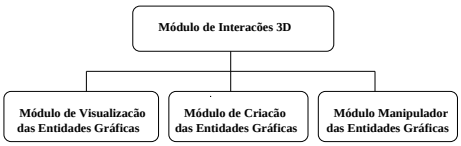

Figura 4: Módulo de Suporte às Interações 3D

módulos devem ser implementadas (figura **??**):

- Módulo de Visualização das Entidades Gráficas;
- Módulo de Criação das Entidades Gráficas;
- Módulo de Manipulação das Entidades Gráficas.

3 IQL

O módulo IQL da ferramenta de interações PRODIA foi projetado, com o objetivo de atender parcialmente os requisitos discutidos na seção 2. Muitas funções já implementadas podem ser aproveitadas na implementação do **Canvas-3D**.

Entre os recursos disponíveis no módulo IQL destacam-se:

- Um modelo de entidades gráficas 3D, denominado SST, com as primitivas:**vértices**, **arestas**, **estruturas ou segmentos**.
- Rotinas para gerenciamento de transformações locais e globais de uma entidade ou de um conjunto de entidades gráficas;
- Rotinas para coordenação da visualização e manipulação de um mesmo conjunto de entidades sob diferentes ângulos de visualização;
- Rotinas para transformações de visualização com perspectiva paralela.

Funcionalmente, IQL é um widget do PRODIA, e os seus códigos estão fortemente interdependentes. Com o intuito de reaproveitar a maior parte dos códigos do IQL para o nosso objetivo, foi necessário desacoplá-los e definir uma interface de comunicação bem clara entre eles. Uma vez desacoplado o IQL do PRODIA, procedeu-se a sua integração no *XView*.

4 Integração do módulo IQL no *XView*

Foram experimentadas duas estratégias para integrar o módulo IQL no *XView*:

1. criar um novo componente de interface, utilizando os recursos oferecidos pelo próprio *XView*, de tal forma que as funções implementadas no IQL fossem encapsuladas neste novo componente;
2. estender os campos de atributos do *Canvas* através do uso de um campo extensível, de tal forma que todos os dados necessários ao módulo IQL pudessem ser armazenados e acessados por ele.

XView dispõe de um conjunto de métodos que permitem criar novos componentes de interface (*X-View Package*)[**?**]. Estes novos componentes podem modificar a aparência ou estender as funcionalidades dos já existentes. O uso destes métodos exige a existência de uma classe-pai, da qual o novo componente herdará as características predominantes. No caso de um novo componente que requer o uso de janelas, só o componente *Window* pode ser utilizado como sua classe-pai. Isso implica que todas as facilidades oferecidas pelo componente *Canvas* [7] deverão ser adicionalmente implementadas. Para evitar este esforço computacional adicional, optamos pela segunda estratégia.

Cada *widget* no *XView* é provido internamente de campos extensíveis (*Key Data*)[**?**] que permitem a associação de um objeto a um outro. A vantagem no uso deste segundo artifício para a implementação do **Canvas-3D** está no uso de todas as funcionalidades disponíveis no *Canvas*. Sob o ponto de vista organizacional do *XView*, **Canvas-3D** é um *Canvas* com um campo estendido, contendo informações necessárias para a execução correta de rotinas do módulo IQL. Logicamente, o **Canvas-3D** é um novo componente de interface que **apresenta todas as propriedades do** *Canvas* e **suporta a execução das funções do módulo IQL**.

5 Comparação entre *Canvas* e Canvas-3D

Para enfatizar as características do **Canvas-3D**, compararemos ambos componentes em função de:

Suporte à Percepção: o *Canvas* não oferece nenhum apoio ao melhoramento do ambiente de trabalho dos usuários, enquanto o **Canvas-3D** dispõe de grades e eixos para aumentar a percepção de profundidade.

Suporte à Visualização: as técnicas de transformação de cenas 3D precisam ser implementadas adicionalmente no *Canvas*. Já o **Canvas 3D** oferece recursos de visualização, tais como as técnicas de transformação da cena (projeção paralela e perspectiva).

Suporte à Estruturação de Entidades: o *Canvas* do *XView* exibe suas primitivas gráficas através das funções do *Xlib*. O **Canvas-3D** possui um modelo de dados que possibilita definições e manipulações mais intuitivas de um objeto.

Suporte à Manipulação: para manipular interativamente as primitivas gráficas visualizáveis tanto no *Canvas* como no **Canvas-3D**, algoritmos de

[7] *Canvas* é uma subclasse de *Window*.

seleção (*picking*) e realimentação visual precisam ser implementados. Estes recursos não são explicitamente disponíveis no *Canvas*, enquanto no **Canvas-3D** as técnicas de seleção hierárquica são encapsuladas como seus recursos.

6 Implementação e Resultados

As principais funções do módulo IQL, implementadas em linguagem C, já se encontram desacopladas da ferramenta de interações PRODIA. Entre elas,

- os módulos criadores e manipuladores dos componentes de interface **Canvas-3D**;

- o módulo para visualização do ambiente 3D;

- as funções para definição das entidades gráficas 3D;

- o módulo de visualização em perspectiva paralela das entidades gráficas.

A interface entre IQL e *XView* é escrita também em C. O *XView* estendido executa no sistema UNIX das estações SPARCstation. A integração foi concebida de tal forma que, para os programas aplicativos, o **Canvas-3D** é inicializado de maneira similar a um *Canvas*, como mostrado nos códigos seguintes:

```
/* Inclusao do widget Canvas do XView */
    Frame          frame;
    Canvas         canvas;

    xv-init(XV-INIT-ARGC-PTR-ARGV,&argc,argv,NULL);
    frame = (Frame)xv-create(NULL, FRAME, NULL);
    /* cria um Canvas */
    canvas = (Canvas)xv-create(frame, CANVAS, NULL);

/* Inclusao do widget Canvas-3D */
    Frame       frame;
    Canvas      frame-id;
    char        *desc;

    xv-init(XV-INIT-ARGC-PTR-ARGV, &argc, argv, NULL);
    frame = (Frame)  xv-create(NULL, FRAME, NULL);
    /* cria um Canvas-3D */
    iql-createframe (frame,  &frame-id,  &desc);
```

A única diferença entre os dois códigos está na chamada às funções: *xv_create()* e *iql_createframe()*. Entretanto, sob o pronto de vista de recursos, a chamada de *iql_createframe()* inicializa todos os recursos, tais como grades, eixos, câmera e entidades gráficas 3D, e torna-os disponíveis aos programas aplicativos. Outro resultado importante obtido é a portabilidade e independência das funções do módulo IQL. Assim, acreditamos que o módulo possa ser incorporado facilmente em outras ferramentas de interações além do PRODIA e *XView*.

7 Trabalhos Futuros

Neste trabalho foi proposta uma estratégia para a integração do módulo IQL no *XView* para estender as suas técnicas de interações. Devido à sua modularidade, pressupõe-se que o módulo IQL é integrável a outras ferramentas de interações. Portanto, pretendemos continuar o nosso trabalho na direção do enriquecimento das funcionalidades do IQL. Mais especificamente, novos recursos para aumentar a percepção 3D e outras técnicas de manipulação 3D [**?**].

8 Agradecimentos

Agradecemos as valiosas sugestões e comentários construtivos dados pelos revisores do Sibgrapi.

Referências

[1] R. Scheifler, J. Gettys, "The X Window System", ACM TOG, 5(2), April 1986, 79-109.

[2] Open Software Foundation, *OSF/MOTIF Manual*, Open Software Foundation, Cambridge, MA, 1989.

[3] Sun Microsystems, OPEN LOOK Graphical User Interface, Sun Microsystems, Montain View, CA, 1989.

[4] D. B. Conner, S. S.Sinibbe and K. P. Herndon , "Three-Dimensional Widgets" *ACM SIGGRAPH 1992*, 183-188.

[5] D. Heller, "XView Programming Manual", Volume Seven, *O'Reilly & Associates,Inc.*, 1990.

[6] F. Loseries, "Functional Interface of the IQL frame type of PRODIA/11", Relatório interno, FhG-IGD, 1990.

[7] D. Krömker, H. Steusloff, H.-P. Steubel, "PRODIA und PRODAT, Dialog- und Datenbankschnittstellen für Systemenentwurfswerkzeuge", Springer-Verlag, Heidelberg, 1990.

[8] M. A. Hagen,"How to Make a Visually Realistic 3D Display", *Computer Graphics* 25(2), April 1991, 76-81.

[9] A. Nye, "Xlib Programming Manual", Volume One, *O'Reilly & Associates, Inc.*, April 1990.

[10] Foley, van Dam, Feiner and Hughes "Computer Graphics: Principles and Practice" , *Addison–Wesley*, 2nd. ed., 1990.

[11] E. A. Bier, "Snap-Dragging in Tree Dimensions", *ACM SIGGRAPH 1990*, 193-204.

[12] G. M. Nielson, D. R. Jr. Olsen, "Direct Manipulation Thecniques for 3D Objects Using 2D Locator Devices", *Proceedings 1986 Workshop on Interactive 3D Graphics* (Chapel Hill, North Carolina October 1986), 175-182.