GERADOR PROGRAMÁVEL DE SEQUÊNCIAS DE IMAGENS E SEU PROCESSAMENTO

Luís Sobral, Alberto Proença
Departamento de Informática, Universidade do Minho, 4700 BRAGA CODEX
email: aproenca@ci.uminho.pt

Palavras chave
Processamento de Imagem, Khoros, Transputers, Windows, Gerador de Interfaces Gráficos

Sumário

Ferramentas gráficas são normalmente utilizadas no desenvolvimento de aplicações de visão por computador; a visão por computador em tempo real com imagens contínuas - imagens vídeo - requer no entanto sistemas computacionais com grande poder de processamento e com suporte a periféricos dedicados à captação, visualização e armazenamento de sequências de imagens.

A identificação de limitações impostas pela utilização da biblioteca de processamento de imagem do sistema Khoros e do seu interface gráfico **cantata**, como ferramenta de desenvolvimento dessas aplicações, sugeriu o porte da biblioteca para um sistema paralelo baseado em transputers e a sua integração automática num interface gráfico para visão. Analisadas as diversas alternativas de implementação de interfaces gráficos em transputers, construiu-se um protótipo de um gerador programável de sequências de imagens e seu processamento paralelo usando a ferramenta WFS (Windows File Server) em MS.DOS. O protótipo desenvolvido inclui também um processo de integração automática de funções no interface gráfico.

1. Introdução

O domínio de visão por computador requer normalmente vários níveis de processamento de informação aplicada às imagens captadas. Este processamento de informação é normalmente constituído por três níveis de processamento: ao mais baixo nível, designado normalmente por processamento de imagem, contendo operações orientadas ao pixel, e produzindo como resultado outra imagem; o processamento de imagem a um nível intermédio, normalmente designado também por análise de imagem, onde as funções são mais complexas produzindo como resultado descrições da imagem; ao nível superior, da compreensão da imagem (*image understanding*), onde as imagens são interpretadas e classificadas, funcionando como apoio a modelos de decisão.

O domínio de processamento de imagem é o que se encontra mais desenvolvido e estável, sendo a preocupação maior em alguns domínios de aplicação, a sua execução em tempo real ou *vídeo rate*. Diversas arquitecturas para visão por computador têm sido propostas[1,2,3], e a paralelização dos algoritmos de visão é crucial para um eficaz aproveitamento das arquitecturas de processamento múltiplo. Esta paralelização é no entanto bastante dependente do modelo de arquitectura paralela adoptada, quer seja homogénea (*array processor*, multiprocessador de memória partilhada ou rede de processadores com memória distribuída), quer heterogénea (combinação das anteriores). Para o desenvolvimento de ambientes de visão em tempo real em arquitecturas heterogéneas são necessárias criar ferramentas adequadas ao teste das diversas alternativas de paralelização.

O processamento de imagem em tempo real é condição quase indispensável ao processamento de sequências de imagens, em particular de sequências de imagens vídeo em captação contínua de uma câmara ou gravador vídeo. A disponibilização de ferramentas que permitam em tempo real (ou tão próximo quanto possível) seleccionar a imagem ou sequência de imagens a captar, executar uma lista de funções de processamento a aplicar a cada imagem, e visualizar/armazenar a(s) imagem(ns) resultante(s), foi um dos objectivos deste projecto[4,5,6]. Ele inclui-se num projecto mais lato a decorrer num consórcio de Universidades Europeias, no desenvolvimento duma arquitectura heterogénea para visão por computador, constituída por uma rede de transputers controlando um *array* de processadores associativos[7].

A ferramenta a construir tinha como objectivo imediato o desenvolvimento de um gerador programável de sequências de imagens e seu processamento paralelo. A sua especificação inicial continha os seguintes pontos:

- portar para a rede de transputers uma biblioteca de processamento de imagem do domínio público (do sistema Khoros[8,9], desenvolvido na Universidade do Novo México, foi a seleccionada), de modo a se estudar estratégias de paralelização de aplicações de visão por computador;
- testar o desenvolvimento de aplicações em transputers que usem o monitor do sistema hospedeiro para interface gráfico com o utilizador (em ambiente de janelas), com recurso a ferramenta existente para ambiente MS.DOS (Windows File Server, WFS[10]);
- aumentar a funcionalidade do Khoros e respectivo interface gráfico de modo a permitir a utilização de periféricos ligados a transputers, dedicados à visão (câmara e monitor) e ao armazenamento de imagens/sequências de imagens (disco);
- construção de ferramentas automáticas para integração de novas funções de processamento de imagem em arquitecturas heterogéneas.

O sistema Khoros encontra-se disponível no domínio público, incluindo o seu código fonte. Esta disponibilidade permite o porte do sistema para outras arquitecturas e facilita o estudo da paralelização dos algoritmos utilizados. Embora o código não esteja em ANSI C, foi possível com algumas modificações portar a biblioteca de processamento de imagem do Khoros para código sequencial em transputers. Encontra-se ainda em estudo o desenvolvimento paralelo de aplicações - quer através da paralelização algorítmica, quer através da paralelização geométrica, quer ainda do uso encadeado e/ou paralelo de funções sobre partes de imagens - bem como o desenvolvimento de ferramentas automáticas que contemplem a componente heterogéneas do sistema na integração de funções de processamento de imagem.

O restante desta comunicação contém uma breve descrição do sistema Khoros em processamento de imagem, com referência às suas limitações no interface com periféricos dedicados a visão (secção 2), a apresentação do ambiente de desenvolvimento de interfaces gráficos para transputers, WFS (secção 3) e o interface desenvolvido, incluindo a ferramenta construída para a inserção de novas funções (secção 4).

2. O sistema Khoros em processamento de imagem

O sistema Khoros é uma colecção de funções e ferramentas para processamento e visualização de imagens/dados. O sistema inclui múltiplos interfaces com o utilizador, geradores de código, vários processamentos de informação e visualização dos dados. Todas as ferramentas do sistema, bem como o sistema em si, podem ser modificados garantindo a sua extensibilidade e portabilidade. A estas características acresce a disponibilidade em domínio público de todas as fontes do sistema.

Da grande variedade de componentes do sistema, no âmbito deste projecto são de destacar os seguintes:

- a biblioteca de processamento de dados, em especial as funções de processamento de imagem;
- a linguagem de programação visual abstracta, **cantata**, para a utilização do sistema Khoros através do interface gráfico X.Windows;
- o utilitário **animate** que permite a visualização de sequências de imagens;
- os interpretadores e geradores de código que permitem a extensão do sistema com novas ferramentas ou com novas funções de processamento de dados (**preview**, **composer**, **ghostwriter** e **conductor**).

A biblioteca de processamento de dados contempla principalmente o processamento de imagem, embora também inclua funções para processamento de sinal. A biblioteca de

processamento de imagem contém as funções mais comuns no nível mais baixo de visão por computador. De entre a grande variedade de funções destacam-se as funções geométricas (rotação, translação, redução, ...), aritméticas (soma, subtracção, divisão, inversão, ...), lógicas (ou, e, ou exclusivo, ...), filtros espaciais 2D (convolução, mediana, ...), filtros de frequências 2D (passa-banda, passa-baixo, passa-alto, ...), histograma (equalização, histograma, ...) e segmentação (binarização dinâmica, determinação de eixos, fecho de contornos, ...). Existem ainda outros grupos de funções, tais como conversões de dados, transformadas e funções baseadas em máscaras. Praticamente todas estas funções processam uma imagem e originam uma imagem como resultado do processamento. O formato de imagem (dados) usado pelo Khoros designa-se por VIFF (Visualization/Image File Format). O formato contém um cabeçalho seguido dos dados, podendo estes ser de vários tipos; os mais frequentes são: bits, bytes, inteiros e real. Também é possível guardar informação relativa à cor e informação dispersa (pares [x,y]).

O controlo das acções a executar pelo Khoros é efectuado através de linhas de comando textuais. O sistema dispõe no entanto de uma linguagem de programação visual, **cantata**, o qual através de uma janela X.Windows converte as instruções icónicas do utilizador em linhas de comandos. A programação visual representa por uma caixa cada função

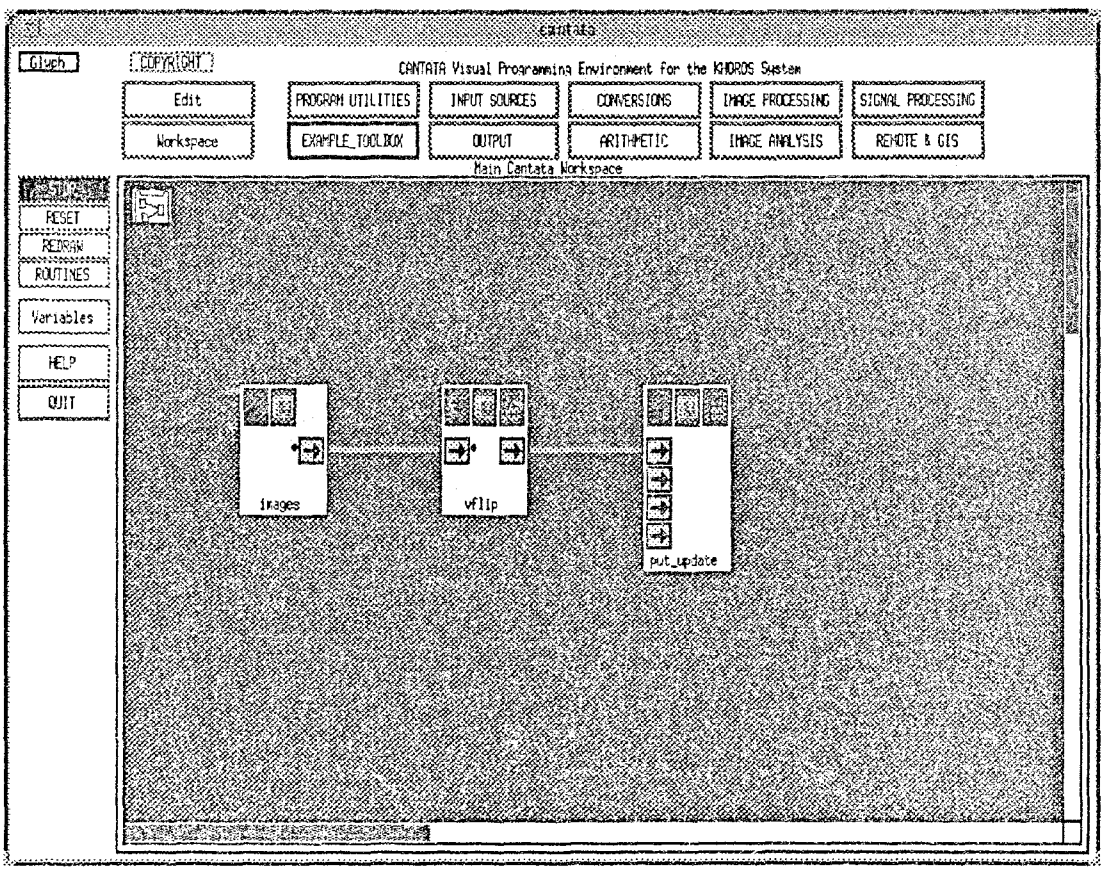

Fig.1 - Cantata

seleccionada num menu, como um utilitário de desenho. O encadeamento de várias operações para obter um dado processamento é efectuado no **cantata** ligando as caixas que representam cada operação. Como algumas funções possuem várias entradas ou várias saídas é possível múltiplas ligações entre diversas caixas. Podem existir entradas ou saídas de uso facultativo. Através da linguagem visual é extremamente fácil construir protótipos de processamento. A figura 1 mostra uma janela do **cantata** com o exemplo de utilização da função *vflip* para processar uma imagem proveniente do disco e visualizar a imagem resultante no ecrã.

A visualização interactiva de sequências de imagens é possível através da função **animate.** As sequências são especificadas através de imagens com várias bandas ou através de ficheiros descrevendo um conjunto de imagens. Esta ferramenta permite a visualização da sequência, imagem a imagem, para a frente ou para trás. A ferramenta dispõe ainda de vários mecanismos de sequenciação (saltos e ciclos).

O sistema Khoros está vocacionado para ser executado num ambiente de computação genérico tipificado por uma *Workstation* Unix sem periféricos específicos. Num ambiente de visão por computador a utilização de equipamentos acessórios dedicados à visão - i.e., à captação de sequências de imagens (câmara), sua visualização contínua (monitor vídeo) e ao armazenamento parcial/integral de sequências de imagens - tem particular relevância no desenvolvimento de aplicações em tempo real, não sendo contudo contemplada no Khoros.

A extensão da funcionalidade do sistema Khoros é obtida através da utilização de ferramentas de desenvolvimento de interfaces com o utilizador, a partir de ficheiros com especificações de alto nível (designadas por UIS - *User Interface Specification*). As UIS podem ainda ser utilizadas para especificar ferramentas do Khoros. As ferramentas mais relevantes incluem interpretadores e geradores de código para criação dos interfaces para o ambiente gráfico:

- **composer**: permite a edição interactiva da UIS na integração de novas funções no interface do **cantata;**
- **preview**: permite visualizar o aspecto gráfico do interface especificado na UIS;
- **ghostwriter**: é o gerador de código para acesso à biblioteca de funções;
- **conductor**: é o gerador do código necessário à produção do interface gráfico X.Windows.

3. O ambiente de interface gráfico para transputers

As arquitecturas baseadas em redes de transputers utilizam normalmente um sistema hospedeiro Unix ou MS.DOS como interface com o utilizador, através duma placa com um transputer que serve de ligação entre o hospedeiro e a rede. O desenvolvimento de aplicações com interfaces gráficos não é ainda directamente suportado pelos ambientes de

desenvolvimento da INMOS para Unix e MS.DOS, sendo apenas fornecido um servidor de I/O no sistema hospedeiro. Este servidor permite o acesso por parte dos transputers ao disco, ao teclado e à impressão de caracteres no ecrã.

Na criação de aplicações gráficas em transputers, o controlo do diálogo pode ser colocado no sistema hospedeiro ou num transputer da rede. Em ambos dos casos, para uma aplicação em transputers usar os recursos do X.Windows ou do MS.Windows é necessária uma extensão do servidor de I/O fornecido, de modo a que este permita a comunicação entre uma aplicação do sistema hospedeiro e uma aplicação da rede, ou de modo a que este disponibilize os recursos gráficos do sistema hospedeiro na rede de transputers. Para o MS.Windows existe uma extensão ao servidor de I/O, o Windows File Server (WFS), o qual permite o controlo dos recursos gráficos por parte das aplicações na rede de transputers, ou seja, o controlo do diálogo é efectuado pela rede de transputers. A grande vantagem do WFS é a de permitir desenvolver a aplicação integralmente na rede de transputers, sem necessidade de gerar código para o sistema hospedeiro.

O WFS é um produto complementar ao ambiente de desenvolvimento da INMOS, permitindo executar aplicações numa rede de transputers que utilizem o interface MS.Windows. O WFS é constituído por uma aplicação MS.Windows e por uma biblioteca de funções gráficas para serem usadas pelas aplicações na rede de transputers. As funções da biblioteca efectuam os pedidos à aplicação MS.Windows - por ex., criar uma janela -, encarregando-se esta de executar esses pedidos. A aplicação MS.Windows gere ela própria os recursos do MS.Windows, incluindo os sinais enviados pelo *kernel* do MS.Windows às janelas da aplicação. Um dos pontos mais favoráveis do WFS é a abstracção da complexidade intrínseca à gestão dum sistema de janelas, dado que o WFS já inclui os conceito de janela e botão. Esta característica facilita o desenvolvimento de aplicações, tornando-se, no entanto, um obstáculo quando as aplicações possuem maior complexidade. Refira-se ainda o facto deste produto possuir alguns *bugs* até à data não resolvidos - gestão instável de janelas múltiplas, devido a violação de mecanismos de protecção de memória do Windows, ... - e de, impôr algumas restrições sérias - disponibilização de alguns recursos gráficos apenas às aplicações que funcionam no transputer raiz, formatos limitados de menus e botões, ...

4. O interface gráfico IPM e o gerador de código

O interface gráfico desenvolvido (IPM - *Image Processing Management*) contempla duas situações distintas: a construção e manuseamento de sequências de imagens, e o processamento de imagem(ns). O interface de sequências é descrito na secção 4.1, e o interface de processamento de imagem é descrito na secção 4.2.

O interface de processamento de imagem pode ser automaticamente estendido com novas funções (genéricas ou de processamento de imagem) através de um gerador de código. O processo automático consiste essencialmente num analisador sintáctico e num gerador de código baseado em *templates* gráficos para recolha de parâmetros de funções de processamento de imagem.

O processo automático de integração de funções (Fig.2) começa no ficheiro de descrição da função (do tipo texto e designado por *pane*), o qual contém toda a informação gráfica é funcional necessária para integrar uma nova função no interface de processamento de imagem. A informação de nível gráfico é usada para gerar o interface gráfico de cada função de processamento de imagem; a informação de nível funcional é usada para determinar os argumentos necessários para invocar a função de processamento de imagem existente na biblioteca. Embora o conteúdo do ficheiro *pane* seja semelhante ao UIS do Khoros (mais detalhes no apêndice A), o gerador de código desenvolvido vai além do disponível no Khoros: permite a integração automática de funções, enquanto que o Khoros apenas possui ferramentas para uma integração assistida, dado que existem partes codificadas pelo utilizador. O gerador de código é descrito na secção 4.3.

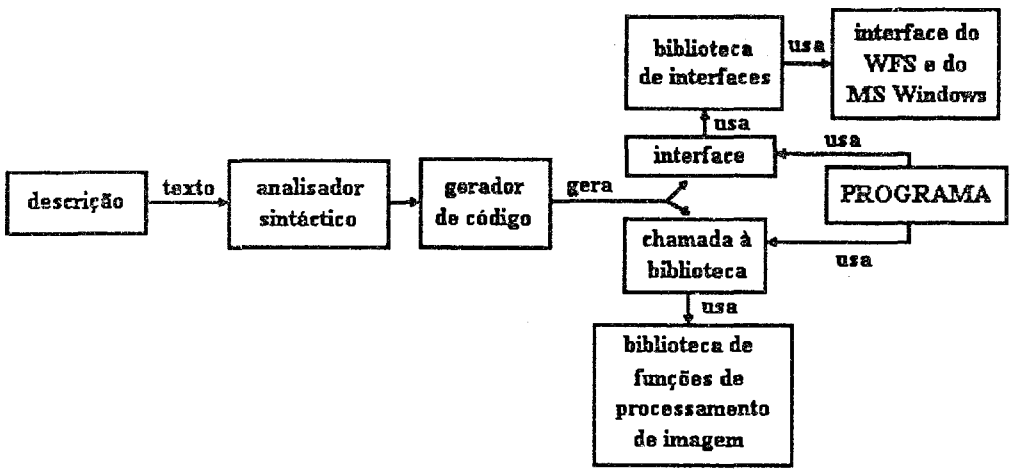

Fig.2 - Processo automático de integração de funções

O processo automático aqui descrito foi testado na criação da biblioteca de funções de processamento de imagem, sendo a grande maioria portada directamente do Khoros. O interface das funções de processamento de imagem é baseado numa biblioteca construída, detalhada no apêndice B.

4.1. Construção de sequências de imagens

Em visão por computador em tempo real torna-se necessário lidar com sequências de imagens. Com esse intuito foi desenvolvido um interface que permite visualizar sequências de

imagens, imagem a imagem e alterar sequências removendo ou adicionando imagens. Quando a aplicação é executada numa rede de transputers o armazenamento/pesquisa de imagens em memória secundária exige um acesso de grande velocidade, normalmente incompatível com a utilização do unidade de disco do sistema hospedeiro. O interface desenvolvido prevê a utilização de um disco da rede dos transputers para além do disco do sistema hospedeiro. O aspecto visual do interface encontra-se na figura 3.

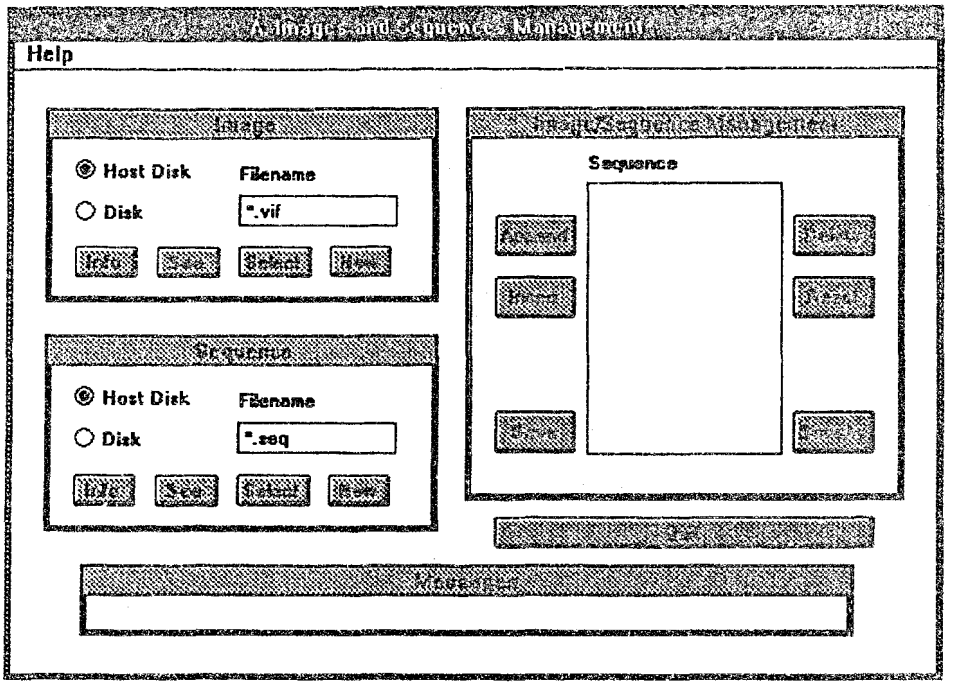

Fig.3 - Construção de sequências de imagens

4.2 Processamento de imagem

O interface desenvolvido para processamento de imagem é particularmente adequado em situações que exijam processamento em tempo real. Em todo o processamento de imagem existe sempre uma origem das imagens a processar e um destino das imagens processadas. Neste caso o interface desenvolvido inclui também como origeme da imagem a processar uma câmara ou um ficheiro do disco da rede; o destino poderá também ser um monitor vídeo ou um disco da rede (Fig.4). Os periféricos para captação, armazenamento directo e visualização de sequências de imagens (vídeo) são recursos exclusivos da rede de transputers, vocacionados para funcionarem em paralelo e em tempo real . Deste modo, o interface com o utilizador pode ser utilizado apenas para activar e interromper o processamento, sem interferência do sistema hospedeiro (desde que não se usem recursos seus, tais como o disco e o ecrã). A principal razão para esta opção é o facto da passagem de informação entre o sistema hospedeiro e a rede de transputers ser demasiado lenta para permitir processamento em tempo real.

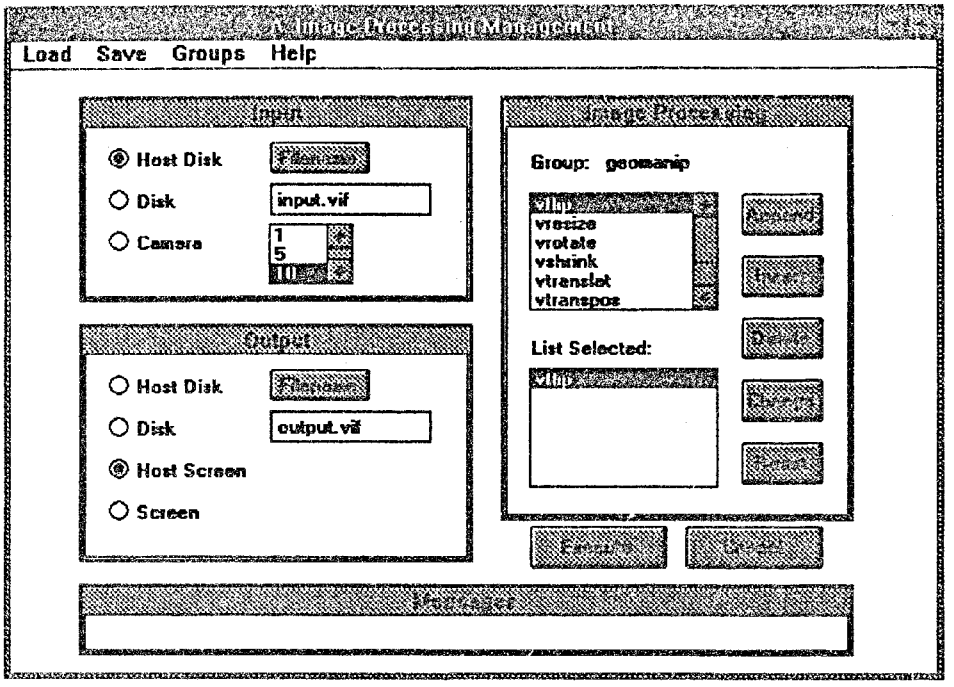

Fig.4 - Janela de processamento de imagem

O processamento é representado por uma fila de funções a aplicar à(s) imagem(ns) de entrada, correspondendo à situação mais comum em processamento de imagem. A fila de funções é uma *pipeline*, a entrada de uma função é a saída da função anterior na fila. Cada função de processamento de imagem possui parâmetros próprios. Na figura 5 apresenta-se a janela dos parâmetros da função *vflip*. Alterando a palavra *PIPELINE* pode-se especificar origens ou destinos de imagens em funções intermédias.

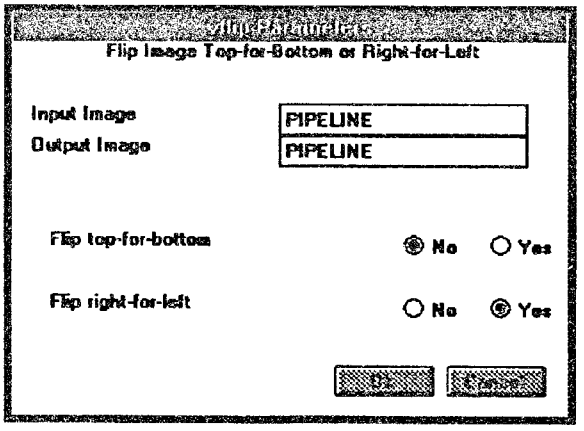

Fig.5 - Parâmetros da função *vflip*

Uma situação particularmente interessante duma lista de processamento, consiste na detecção de movimento em imagens provenientes da câmara, por forma a que sempre que este é detectado, a informação relevante seja gravada em disco.

4.4. Gerador de código

O gerador de código altera todos os ficheiros necessários para integrar a função no ambiente gráfico. Apenas é necessário juntar o código da função ao programa. O gerador de código efectua as seguintes operações:

- cria um novo grupo de funções se necessário (existem grupos de funções geométricas, aritméticas, lógicas, transformadas, filtros espaciais, ...);
- adiciona a função à tabela de funções;
- modifica os ficheiros para incluírem as declarações das novas funções;
- gera a estrutura dos parâmetros da função;
- cria a função gráfica para obter os parâmetros da função;
- cria a chamada a função da biblioteca.

No caso de alguma das funções já existir, o seu código é substituído pela nova versão, permitindo assim acrescentar novas versões de funções já existentes na biblioteca.

A chamada à função efectua várias operações:

- obtém as imagem necessárias;
- verifica se o tipo das imagens é aceite pela função;
- chama a função na biblioteca;
- se necessário grava a imagem resultante;
- liberta as imagem intermediárias e adapta a imagem da *pipeline*.

A função na biblioteca de funções tem que obedecer a regras rígidas. Os seus argumentos devem coincidir (no tipo e ordem) com os declarados no ficheiro *pane*. Por exemplo, a função *flip* tem o seguinte protótipo:

```
lvflip( imagemVIFF Img, boleano FlipLR, boleano FlipTB)
```

A verificação do tipo de imagem aceite pela função pode ser bastante complexa, sendo bastante difícil incluir no ficheiro *pane* todo o tipo de informação. A opção tomada foi suportar as mais comuns. A linha do tipo t permite especificar o tipo de imagem aceite. Adicionalmente quando a função aceita várias imagens, verifica-se se elas são do mesmo tamanho e tipo. Em alguns casos esta verificação deve ser retirada.

O principal objectivo do gerador de código será evoluir para uma abordagem em que tire partido da arquitectura. Nesse caso as UIS terão que incluir informação sobre as necessidades computacionais de cada função, e consoante essas necessidades a chamada às funções deverá lidar com a respectiva paralelização.

5. Conclusão

O sistema Khoros é um sistema poderoso, flexível, e o seu interface visual permite uma fácil utilização do produto no domínio genérico da visão por computador. A sua utilização em ambiente de sequências de imagens (vídeo) e em tempo real é no entanto bastante limitada, dado que não está ainda preparado para suportar um sistema de processamento paralelo com periféricos dedicados à visão, nomeadamente à captação/visualização contínua de imagem, e ao armazenamento de sequências de imagens em tempo real.

O desenvolvimento de um novo interface gráfico que ultrapassasse essas limitações foi empreendido usando a ferramenta WFS em MS.DOS para suporte do diálogo entre a aplicação (em transputers) e o utilizador (no sistema MS.DOS). Embora o WFS permita que as aplicações de visão por computador possam ser integralmente desenvolvidas no sistema paralelo, possui contudo limitações e *bugs* que dificultam a sua utilização em situações mais complexas.

Integrar automaticamente as funções de processamento de imagem no interface gráfico reduziu drasticamente o tempo necessário para a integração das funções quer no interface, quer no sistema de processamento, assegurando uma homogeneidade de comportamento de todas as funções. Grande parte deste sucesso deveu-se ao uso das UIS do Khoros. Este integrador poderá facilmente evoluir de modo a tirar partido dos recursos de uma arquitectura heterogénea para visão por computador, baseada em transputers e *arrays* de processadores associativos.

A experiência entretanto obtida com este projecto sugere também a sua evolução no sentido de se tentar modificar o **cantata** de modo a nele se incluir a maioria das características já implementadas no protótipo desenvolvido com o WFS.

Referências

[1] C.Weems, E.Riseman and A.Hanson, "Image Understanding Architecture: exploiting potential parallelism in machine vision", *Computer*, **25**, (2), pp. 65-68, February 1992

[2] V.Cantoni, M.Ferreti, S.Levialdi and R.Stefanelli, "PAPIA: Pyramidal Architecture for Parallel Image Analysis", Proc. 7th IEEE Symposium on Computer Arithmetic, IEEE Comp. Soc. Press, pp. 237-242, 1985

[3] M.Sunwoo and J.Aggarwal, "VisTA for a general purpose computer vision system", Proc. 10th Int. Conf. Pattern Recognition, Vol. 2, pp. 635-641, IEEE Com. Soc. Press, 1990

[4] L.Sobral, "An Image Processing System for Transputers", Relatório de Estágio de Licenciatura em Engª de Sistemas e Informática, Universidade do Minho, 1992

[5] A.Oliveira, "Programmable Source of Processed Images", Relatório de Estágio de Licenciatura em Engª de Sistemas e Informática, Universidade do Minho, 1992

[6] C.Horta, "Ambiente de Desenvolvimento para Processamento Paralelo de Imagem", Relatório de Estágio de Licenciatura em Engª de Sistemas e Informática, Universidade do Minho, 1993

[7] R.Storer, M.Pout, A.Thomson, E.Dagless, A.Duller, A.Marriot and P.Hicks, "An associative processing module for a Heterogeneous Vision Machine", *IEEE Micro*, **12**, (3), pp. 31-41, June 1992

[8] Khoros Group, "Khoros Manual - Volume I - Users Manual", University of New Mexico, USA, September 1991

[9] Khoros Group, "Khoros Manual - Volume II - Programmers Manual", University of New Mexico, USA, September 1991

[10] Nexis, "Windows File Server 3.1", Nexis Ltd, UK, October 1991

Apêndice A. Os ficheiros pane (UIS)

Cada linha do ficheiro especifica em principio um parâmetro de entrada ou de saída da função da biblioteca, embora algumas linhas contenham informação que não está relacionada com os parâmetros da função. As linhas possíveis são: **M** (linha de forma); **P** (linha de subforma); **I** (imagem de entrada); **O** (imagem de saída); **l** (valor lógico); **i** (valor inteiro) e **f** (valor real). Cada linha de parâmetros (**I, O, l, i, f**) contém o comprimento, a posição e o texto do campo na janela de entrada, e o nome da variável a ser usada para esse parâmetro. As linhas do tipo **i** ou **f** contêm também o valor mínimo, máximo e o valor por defeito do parâmetro. Na linha de imagens de entrada existe um campo que indica se a imagem é opcional, e outro que indica se por defeito é usado. As linhas **M** e **P** indicam o grupo da função, o nome interno da função e o tamanho e posição da janela de entrada dos parâmetros. Existem duas linhas especiais: a linha **b** permite introduzir comentários e a linha **t** permite restringir o tipo da imagem de entrada. De seguida apresenta-se o conteúdo do *pane* que originou a figura 5.

```
-M 'Geometric Manipulations' geomanip
-P 80x38+22+2 +5+0 'Flip Image Top-for-Bottom or Right-for-Left' vflip
-I 0 1 50x1+2+2 +0+0 'Input Image ' i
-O 0 1 50x1+2+3 +0+0 'Output Image' o
-l 50x1+4+8 +0+0 1 'Flip right-for-left' 'No ' 'Yes' r
-l 50x1+4+6 +0+0 0 'Flip top-for-bottom' 'No ' 'Yes' t
```

Na linha M, *geomanip* é o grupo da função. Na linha P, *80x38* é o tamanho da janela de entrada, *(22,2)* é a sua posição, *(5,0)* é a posição do título, *vflip* indica o nome da função correspondente na biblioteca.

A maior diferença entre o formato usado e o do Khoros é a ordem das linhas. Neste formato as linhas de parâmetros devem estar ordenadas de acordo com a ordem dos parâmetros da função da biblioteca. Outra diferença é o tamanho dos objectos, principalmente nos campos lógicos é necessário mais espaço, porque a implementação é diferente. As linhas do tipo t não são usadas no Khoros.

Apêndice B. Interface gráfico das funções

Cada função de processamento de imagem necessita de parâmetros (por ex., uma binarização necessita do ponto de binarização). A função também precisa de saber qual a imagem a processar, e qual o destino da imagem processada.

A maior parte das funções de processamento de imagem de baixo e médio nível possuem parâmetros do mesmo tipo. Normalmente estes são números inteiros, reais, booleanos ou imagens. O código para recolher os parâmetros das funções é bastante simplificado se já existirem funções de recolha destes tipos de parâmetros. Esta solução tem o benefício de tornar o código gráfico independente do WFS.

A biblioteca de interface gráfico criada possui funções para criar uma janela de entrada, adicionar campos para recolha de parâmetros inteiros, reais, booleanos, imagens e sequências de opções exclusivas. É necessário indicar a posição de cada campo na janela de entrada. A biblioteca também permite especificar valores iniciais e, no caso de campos inteiros e reais permite especificar valores mínimos e máximos. Depois de criar os campos necessários, existe uma função que espera pelo utilizador e adapta os campos modificados. Como exemplo apresenta-se o código que gera a figura 4 (exemplo simplificado).

```
vflipGetParam(arg) =
  CreateInput("vflip Parameters", 48, 66, 90, 36,
        "Flip Image Top-for-Bottom or Right-for-Left", 15, 0)
  CreateString(arg.InputImage, "Input Image", 3, 6, 63, 6)
  CreateString(arg.OutputImage, "Output Image", 3, 9, 63, 9)
  CreateLogical(arg.FlipRL, "Flip right-for-left", 6, 24, 63, 24, "No ", "Yes")
  CreateLogical(arg.FlipTB, "Flip top-for-bottom", 6, 18, 63, 18, "No ", "Yes")
  WaitForUser()
```

Na segunda linha, *CreateString* cria um campo para recolher uma *string*. O seu texto é *'Input Image'* , a posição do texto é *(3,6)* e a posição do campo é *(63,6)*. A variável *arg.InputImage* contém a *string* inicial (*'PIPELINE'*) e irá conter a *string* modificada depois da chamada a *WaitForUser()*.