Uma ferramenta 3D aplicada ao Desenvolvimento de Algoritmos Perceptuais em Humanóides

Schubert R. Carvalho[1], Claus C. Aranha[1], Luiz M. G. Gonçalves[2]

[1]Universidade Estadual de Campinas
Caixa Postal 6176, 13083-970, Campinas, SP, Brasil

[2]Universidade Federal do Rio Grande do Norte
DCA-Centro de Tecnologia-UFRN, 59.072-970 – Natal, RN, Brasil
{schubert.carvalho,claus.aranha}@ic.unicamp.br, lmarcos@dca.ufrn.br

Abstract: This work intends to build a simulator to be used in the research and development of control and perceptual algorithms for humanoid robots. It evolves the tools presented in [04, 07], so that they can be applied to a realistic three-dimensional virtual environment. In this work, the virtual agent's behavior is controlled by a visual perception platform based in sensorial information obtained from a simplified model of the environment. The algorithms developed here are intended for use in a real humanoid robot.

Resumo: Este trabalho, visa a construção de um simulador que será utilizado como auxílio a pesquisa no desenvolvimento de algoritmos perceptuais e de controle para humanóides. Tal aplicativo é uma evolução das ferramentas apresentadas em [04,07], para que sua aplicação seja possível em um ambiente virtual realístico tridimensional. No presente trabalho, o comportamento do agente virtual é controlado por uma plataforma de percepção visual, baseada em informações sensoriais, extraídas de um modelo simplificado do ambiente. Os algoritmos desenvolvidos neste simulador têm como um de seus objetivos, serem incorporados a um robô humanóide real.

1 Introdução

No campo da Realidade Virtual, evidencia-se praticamente que “tudo” possa acontecer. A mesma cria a ilusão de um mundo tridimensional gerado pelo computador a partir dos dados do usuário, permitindo que o mesmo possa realizar imersão, navegação e interação em tempo real, sem riscos de acidentes. Situações que são impraticáveis na vida real podem ser modeladas artificialmente [18].

Entretanto, quando se pretende aplicar os algoritmos desenvolvidos em um ambiente simulado a um robô real, certamente ocorrerão problemas. Devido a impossibilidade de se modelar um ambiente real por completo no computador, em decorrência das complexidades físicas associadas ao mesmo. Por exemplo, o algoritmo desenvolvido no simulador pode simplesmente não funcionar na prática, em virtude de fatores que não foram levados em consideração na simulação, como o grau de discretização a ser considerado na definição de um modelo geométrico (um problema numérico de representação). Até que ponto esta “falsa realidade” é permitida, para que as soluções conseguidas no ambiente simulado possam ser utilizadas em um modelo real? Os algoritmos que funcionam eficientemente no ambiente de simulação, funcionarão com a mesma eficiência em uma plataforma real? Estas e outras perguntas

devem ser feitas quando se pretende buscar soluções em modelos simulados e aplicá-las em robôs que irão desempenhar tarefas em ambientes reais.

Dois aspectos importantes devem ser levados em consideração quanto, a construção de modelos simulados que serão aplicados em robôs humanóides (simulados ou reais). O primeiro diz respeito ao ambiente que será modelado. O mesmo deve ser uma cópia fiel do ambiente real ou um modelo bem aproximado? Ou seja, os conceitos físicos que regem o ambiente, como: gravidade, atrito, inércia, etc. devem ser levadas em consideração. Esta necessidade é para que não se tenham surpresam com o comportamento esperado da criatura, ou para que não seja necessário se fazer diversas modificações nos algoritmos desenvolvidos. O segundo trata da metodologia adotada na construção do humanóide, neste ponto alguns fatores devem ser levados em consideração: a arquitetura deve propor um modelo computacional realístico, no sentido de que os algoritmos desenvolvidos no ambiente simulado possam ser migrados, sem muita adaptabilidade, para o modelo real. Outro fator aborda a tarefa que será realizada pelo robô. Isto é importante devido a interação que o mesmo terá com o ambiente, ou seja, quais os tipos de sensores que ele deverá utilizar para executar seu objetivo da forma desejada e como será a informação que ele receberá do ambiente. Visto que, tal informação, na maioria dos casos, provocará uma reação sensor-motor no mesmo.

A priori, será tratada a informação visual, visto a grande importância que a visão humana possui para ajudar as pessoas a reagirem em um ambiente conhecido ou não no desenvolvimento de uma dada tarefa. Muito esforço tem sido feito no desenvolvimento de novas técnicas ou no melhoramento de técnicas já conhecidas, para a extração de características obtidas a partir de dados visuais(imagens), capturadas por plataformas robóticas em ambientes restritos ou abertos. As principais abordagens são: que tipo de características devem ser extraídas do ambiente, como obter estas características, como definir característica, como processar as imagens capturadas, qual o tempo necessário que o sistema deve ficar olhando para uma dada região para adquirir a informação(característica) correta e como fazer isto tudo de forma eficiente e automática?

A pesquisa em visão computacional aplicada a robótica requer máquinas caras, e que podem ser danificadas no decorrer de uma certa tarefa, para o desenvolvimento de experimentos. Dentre estes equipamentos, são encontrados câmeras, pequenos robôs, cabeças estéreos, ou seja, cabeças robóticas equipadas com duas câmeras afastadas uma certa distância umas das outras. Deixando de lado o custo destas arquiteturas dedicadas, o uso de robôs reais restringem o desenvolvimento de algoritmos e experimentos a características específicas de um robô em particular (número de graus de liberdade, dispositivo sensor-motor, etc.). Além de não ser recomendável executar qualquer tipo de experimentos em um robô real; pois uma tarefa perigosa pode comprometer sua caríssima estrutura.

Com o que foi exposto acima, este trabalho apresenta um ambiente simulado para a pesquisa em visão robótica e humanoides denominado afetivamente de “Câmbio”. Com a utilização de um robô humanóide simulado, a pesquisa nestas áreas pode ser feita a baixo custo, requerendo somente um computador e um software gratuito (OpenGL) para a interface gráfica. Como tudo é feito em software, pode-se fazer inúmeras simulações e utilizar várias estruturas robóticas sem a necessidade de se utilizar o modelo real. Pode-se testar também algoritmos e situações que podem colocar

o robô no seu limite, sem a preocupação de danificá-lo. Com a utilização de ferramentas de simulação eficientes, pode-se trabalhar em vários campos da robótica, como: cognição, atenção, desenvolvimento sensor-motor, visão computacional, visão robótica entre outros, sem a necessidade imediata do robô real. Modelos matemáticos para estas áreas podem ser criados e desenvolvidos nestas ferramentas de simulação, até que testes exaustivos tenham sido feitos para que tais algoritmos possam ser testados em plataformas reais.

Neste trabalho, o simulador esta voltado ao domínio da visão, a arquitetura completa de um humanóide ainda está sendo desenvolvida, o que significa que no presente momento é proposto uma arquitetura de visão para a aquisição da informação visual. Dessa forma, são apresentadas algumas técnicas de computação visual como: operadores Gaussianos para a extração de características, utilização do algoritmo de vergência para o posicionamento da atenção visual, características de movimento "*motion"* para acompanhar objetos que se movem e são utilizados dois braços para completar a informação visual.

(a) (b)

Figura 1: Interface do Roger (a) e interface do Câmbio (b).

2 Trabalhos Relacionados

O simulador tridimensional proposto neste trabalho, Câmbio, teve sua inspiração vinda de outro projeto, "*Roger-the-Crab*" [04, 07]. O item (a) da Figura 1, mostra a interface antiga do Roger (2D) e o item (b) mostra a interface atual do Câmbio (3D). Ambas possuem basicamente os mesmos recursos: dois olhos e dois braços.

O *Roger* foi originalmente proposto em 1991, para ser usado como ferramenta de trabalho para a pesquisa em visão computacional de baixo nível. Desde então, ele vem sendo usado como auxílio ao desenvolvimento de ferramentas computacionais e algoritmos em vários campos de pesquisa, como o controle da atenção [02]. No estudo de meta-heurísticas feitas com algoritmos de aprendizado de máquina (*machine learning*), para categorização de padrões [04], no desenvolvimento de novos modelos para animação por computador [01], etc.

As duas principais melhorias que estão sendo introduzidas, em relação à versão anterior, são a adição da terceira dimensão para o modelo de percepção do ambiente (visão estéreo 3D), assim como adaptá-lo de tal forma que ele possa ser usado para testes com robôs humanóides. Junto com estas melhorias vem também o aperfeiçoamento da percepção visual. Acreditamos que este modelo possa representar

de forma mais realística o mundo real, para sistemas que serão implementados posteriormente em arquiteturas robóticas reais ou para simuladores de robôs humanóides agindo em ambientes reais (humanos virtuais). A modelagem tridimensional é um requisito importante no desenvolvimento de algoritmos de navegação e para a criação de políticas de navegação para humanóides baseadas em informação visual.

Atenção visual pode ser definida como, a habilidade de selecionar uma região de interesse (um alvo), para a extração de informação, a qual será útil para o desenvolvimento de uma certa tarefa. Um dos desafios encontrados hoje é adicionar, a um humanóide, a habilidade de mudar o foco de atenção de uma região de interesse para outra. Isto devido a mudanças no ambiente ou devido a aquisição de diferentes tipos de dados. Em um trabalho como este, onde a atenção ocorre em um ambiente simulado, as regiões de interesse são objetos virtuais descritos por um modelo matemático. Assim, o controle da atenção visual é uma medida de interação entre o robô e seu ambiente. Alguns dos trabalhos relacionados a problemas de atenção visual [10, 14, 17], definem somente imagens estáticas como objetivo para os seus algoritmos atencionais, não considerando aspectos temporais como o movimento (*motion*), reconstrução estéreo (disparidade) e comportamento. Uma técnica muito comum usada em alguns trabalhos [12, 13], é a utilização de operadores baseados nas derivadas da função gaussiana, para a extração de características de uma certa imagem. Outra linha de pesquisa baseia-se na explicação e imitação do modelo visual humano [12, 15]. Em [12], foi feita uma simulação da fóvea, onde são colocadas as suas zonas de interesse.

Em geral, estes trabalhos foram desenvolvidos para processar imagens reais, ou seja, imagens adquiridas por uma plataforma robótica real e na maioria das vezes com custos elevados. Pesquisas utilizando percepção simulada realística não são muito encontrados na literatura, o que vem mostrar a necessidade em se ter um simulador para testes.

3 Teoria Adotada

Nesta etapa da pesquisa, utilizamos conceitos de aquisição e processamento de imagens, para a arquitetura de percepção visual do humanóide.

3.1 Operador Gaussiano

Na etapa de processamento das imagens para extração de características visuais, utilizamos operadores de convolução bidimensional, usando como máscaras funções definidas por derivadas direcionais, de ordem zero a dois do operador gaussiano; com diferentes valores de escala (σ). A equação 1 representa a fórmula básica para a distribuição gaussiana. A equação 2 representa e equação de convolução gaussiana e seus kernels são dados pelas equações 3, 4 e 5, cada um em suas direções ortogonais. O motivo da utilização de tais equações, é que elas são utilizadas em muitos dos algoritmos de extração de características em imagens, devido as suas propriedades de suavização. Em geral esta é uma fase de pré-processamento antes da extração final das características da imagem. Em um trabalho recente, Ballard [12] usou até a derivada de ordem três da gaussiana como filtro direcional para a extração de características, aplicado a tarefas de atenção e de reconhecimento. No presente trabalho, foi utilizada até a derivada de segunda ordem (Laplaciano da Gaussiana) do operador Gaussiano.

$$g_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-x^2/2\sigma^2} \quad (1)$$

$$G_{d=x,y}^{(k=0,1,2)} = g_d^{(k)} * I_t \quad (2)$$

$$\left.\begin{array}{l} g_x^{(0)}(x,y) = \lambda e^{ax^2} \\ g_y^{(0)}(x,y) = \lambda e^{ay^2} \end{array}\right\} \quad (3)$$

$$\left.\begin{array}{l} g_x^{(1)}(x,y) = 2a\lambda e^{a(x^2+y^2)}x \\ g_y^{(1)}(x,y) = 2a\lambda e^{a(x^2+y^2)}y \end{array}\right\} \quad (4)$$

$$\left.\begin{array}{l} g_x^{(2)}(x,y) = 2a\lambda e^{a(x^2+y^2)}(2ax^2+1) \\ g_y^{(2)}(x,y) = 2a\lambda e^{a(x^2+y^2)}(2ay^2+1) \end{array}\right\} \quad (5)$$

$\forall(x,y) \in ([-s,+s],[-s,+s])$; where $a = \frac{-1}{2\sigma^2}$, $\lambda = \frac{1}{\sigma\sqrt{2\pi}}$, $\sigma = 1.7$, and $s = 3$.

3.2 Disparidade Estéreo

Na etapa de extração de características é calculada também a disparidade estéreo, a qual pode ser definida como a diferença percebida entre as posições de um objeto (ou ponto) em duas imagens geradas por duas câmeras (olhos) posicionadas uma certa distância uma da outra. Há vários métodos na literatura para se calcular o mapa de disparidade estéreo [08, 11]. Isto é, o mapa onde é determinado a disparidade de cada píxel em uma imagem com relação ao seu píxel correspondente na outra imagem. Neste trabalho utilizamos uma abordagem simples baseada em cálculo de correlação por área para determinar a disparidade em cada par de imagens, sobre o mapa de características obtido com a derivada de segunda ordem da gaussiana, referida acima. Uma forma simples e prática de se realizar o controle do foco de atenção do humanóide é escolher um ponto no espaço e tentar colocar este ponto no píxel central (fóvea) das imagens fornecidas por ambas as câmeras (olhos). Pode-se determinar um dos olhos como sendo o olho dominante do movimento (que se moverá na direção do ponto no espaço) e fazer o outro olho seguir o dominante. Para isto, basta tentar a cada instante de tempo do movimento, sendo realizado discretamente, minimizar a disparidade estéreo para o ponto central do olho não dominante em relação ao ponto central do olho dominante, usando cálculos de correlação para estes pontos centrais. Isto faz com que o olho não dominante siga o olho dominante, terminando centrado no ponto desejado.

Para que seja possível a extração de características, as quais serão usadas por processos de mais alto nível, é necessário que ambos os olhos estejam “foveados” (centrados), num ponto ou região. Nota: estes processos cognitivos de mais alto nível serão estudados posteriormente e estão fora do escopo do presente artigo, ver [16] para mais detalhes. Para que a região de interesse consiga ficar na fóvea, propomos usar um algoritmo de vergência, genericamente explicado abaixo, para maiores detalhes sobre sua implementação ver seção 4.5.

Um píxel correspondente a um determinado ponto no espaço, vindo de uma das duas imagens, é escolhido para ser o centro de atenção para uma das câmeras denominada olho dominante. Neste momento, alguma política de atenção visual indica que um determinado ponto ou região do espaço deverá estar no ponto central (fóvea), desta câmera. Ela então se move a pequenos deslocamentos em direção ao ponto escolhido, enquanto que o olho não dominante tenta minimizar a disparidade estéreo do píxel central de sua imagem em relação ao píxel central da imagem do olho dominante. Desta forma, através da minimização da disparidade estéreo, consegue-se manter os olhos mais ou menos focados numa determinada região do espaço enquanto o objetivo

ainda não foi atingido pelo olho dominante. Após este chegar ao destino, o olho não dominante rapidamente converge para uma situação de disparidade mínima.

3.3 Movimento (*Motion*)

Uma outra utilidade para o mapa de disparidade, entre duas imagens, é a restauração de características de movimento em cenas dinâmicas. Se o sistema deseja manter seu foco de atenção em uma área que possui objetos se movendo, ele pode capturar uma série de imagens obtidas por uma câmera estática e calcular o mapa de disparidade para as mesmas. Desde que a câmera não se mova, qualquer disparidade (intensidade da disparidade) encontrada devido a movimentação dos objetos na cena, será proporcional a mobilidade dos objetos. Isto é, o movimento pode ser calculado como a disparidade entre uma imagem tirada no instante *t* e a imagem tirada no instante *t* –1. Note que, o campo de movimento [09] poderia ser calculado aqui, mas nós preferimos usar diretamente este tipo de disparidade como uma medida de movimento, uma vez que isto pode também ser usado para resolver problemas referentes ao controle da atenção e outras tarefas de cognição. O movimento pode ser calculado como sendo a diferença entre duas imagens, em [06,05] isto foi suficiente para o controle da atenção. Assim, isto é uma característica a mais introduzida como melhoria neste trabalho.

4 Implementação

O Câmbio foi desenvolvido baseado em uma arquitetura modular mostrada na Figura 2. Foi gerada uma aplicação multi-processo, ou seja, cada módulo do Câmbio é executado como um processo diferente. Outra característica utilizada, foi a habilidade da troca de um módulo por outro contendo uma implementação diferente, sem a necessidade de modificar outras partes da arquitetura. Atualmente o Câmbio possui quatro módulos: *World Modeling Module, Sensory Servo Module, Decision Making Module* e *Control Module*. O multi-processamento utiliza o *kernel* do sistema operacional Linux como base de implementação, podendo ser portável a outros sistemas operacionais que possuam implementações do OpenGL.

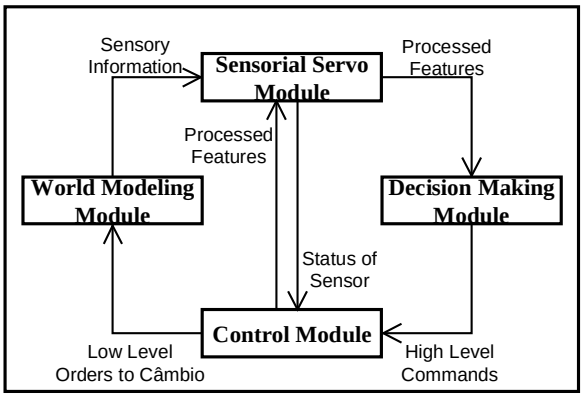


Figura 2: Módulos da implementação do Câmbio.

4.1 Modelo do Ambiente *(World Model Module*-WMM)

Na arquitetura do Câmbio é utilizado um modelo geométrico simplificado para a criação do ambiente. Os objetos são armazenados através de suas características matemáticas. Por exemplo, uma esfera no ambiente simulado é armazenada e indexada por alguns índices chaves, os quais a identificam entre os outros objetos presentes no mundo. Um

índice de tipos identifica os tipos de objetos (esferas, cubos, etc.). Existe o interesse em ir mais adiante com as características dos objetos, ou seja, a quantidade de informação que pode ser extraída dos mesmos através da inclusão de texturas, massa e densidade, daí a utilização desta forma de indexação. Uns tipos especiais de objeto são as próprias partes do Câmbio. O mundo armazena os parâmetros para estes objetos, os quais são alterados de acordo com as requisições do *Control Module*. Atualmente o Câmbio é composto por três tipos de objetos:

1. Olhos: os quais são responsáveis pela aquisição de informação sensorial, possuem dois graus de liberdade – movem-se para direta, esquerda e para cima e para baixo;
2. Pescoço: com um grau de liberdade – rotação para esquerda e direita em torno do seu eixo;
3. Braços: os quais possuem quatro graus de liberdade – dois para os cotovelos e dois entre o braço e o antebraço. O braço possui também um sensor de toque na extremidade dos braços.

Outra função do WMM é produzir informação sensorial a medida que esta seja requisitada pelo *Sensory Servo Module* (SSM). Na interface atual, o SSM pede ao WMM para prover as leituras necessárias para um dos sensores do humanóide. Ele fornece o modelo do mundo com os parâmetros necessários para checar a informação proveniente do ambiente e gerar uma resposta sensorial. Quando o SSM quer trabalhar em uma imagem, ele pede a informação sensorial de um dos olhos. O WMM utilizará a posição e a orientação dos olhos para construir a informação visual e enviá-la ao processo que está sendo executado no SSM.

4.2 Módulo de Percepção (*Sensory Servo Module-SSM*)

O SSM deve processar a informação recebida do WMM, de acordo com as requisições do *Decision Making Module* (DMM). O SSM funciona como um servidor de informação visual, o qual processa o mapa de características construído a partir das imagens para os diferentes algoritmos que serão usados em processamento de mais alto nível para o controle da atenção. A informação sensorial para os braços, atualmente, é transferida diretamente para o DMM sem um processamento mais avançado. Até o momento foi utilizado o operador gaussiano O uso dos mesmos é provado ser útil para a extração de características visuais voltadas tanto para propósitos atencionais [02,03,05] quanto para o de categorização de padrões [02, 06, 12, 13]. Além do comportamento atencional, a disparidade estéreo também será usada no controlador de vergência implementado no *Control Module*, baseado na disparidade entre as imagens adquiridas em ambos os olhos ao mesmo tempo. Para a detecção de movimento, o servidor salva a imagem para vários ciclos consecutivos e envia para o DMM a disparidade calculada entre estas imagens.

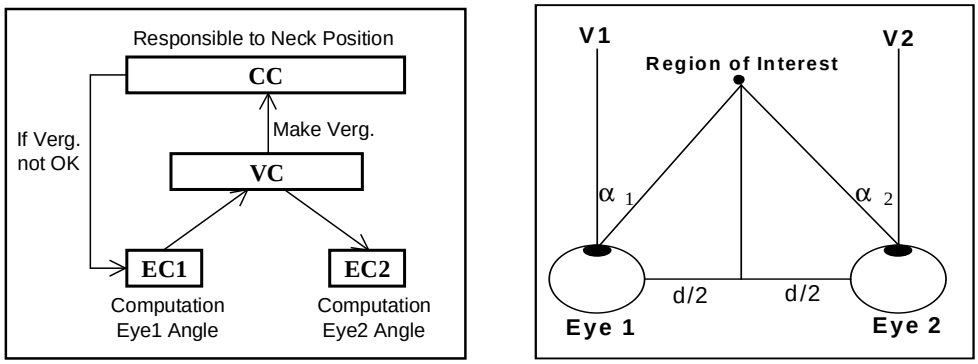
4.3 Módulo de Decisão (*Decision Making Module-DMM*)

O DMM é o módulo mais gerenciável pelo usuário. Ele usa a informação vinda do SMM, aplica algum tipo de algoritmo definido pelo usuário, o qual resulta em uma ação que será transferida para o *Control Module*. O DMM pode conter aprendizado, atenção, reconhecimento de padrões, baseados nos serviços oferecidos pelos outros módulos. Neste trabalho, foi implementado um controlador de alto nível apenas para o algoritmo

de vergência. Outros controladores de alto nível podem ser implementados usando basicamente as mesmas características sensoriais. Como citado anteriormente, estes processos de mais alto nível não são o escopo deste trabalho, sendo implementados geralmente em aplicações de inteligência artificial, a serem feitas a título de trabalhos futuros sobre a corrente implementação.

4.4 Módulo de Controle (*Control Module - CM*)

A função do *Control Module* (CM) é receber ações em alto nível enviadas pelo DMM, e transformá-las em diretivas de controle em baixo nível para os atuadores do robô. Quando o DMM diz ao CM "mova o olho direito para a esquerda", o CM envia esta requisição para o WMM na forma "incremente o ângulo *Vergencia_Direita*" na direção do objeto (olho direito), sendo na prática o controlador EC_2 (ver Figura 3), solicitado a ser acionado para a esquerda sendo incrementado, de digamos 1 grau. O CM armazena o mesmo nível de informação sobre os objetos do robô que o ambiente possui e altera estas informações de acordo com suas ações. Erros no hardware simulado podem ser introduzidos, baseados em modelos de erros ocorridos em ambientes reais, com o propósito de testar os vários algoritmos que venham a ser desenvolvidos para os processos de mais alto nível. O CM possui ainda alguns algoritmos de baixo nível que podem ser executados em *background* pelo DMM.



(a) (b)

Figura 3: Controlador de vergência (a) e limitações do ângulo de vergência (b)

4.5 Algoritmo de Vergência

A idéia principal do algoritmo de vergência é manter sempre ambos os olhos centrados em algum lugar no ambiente, enquanto o robô procura por um novo alvo de interesse para a atenção visual, tal como é feito em [6]. Para que seja possível executar e manter a vergência em um ponto no espaço, é preciso considerar duas restrições. A primeira restrição, trata da definição do espaço de manipulação do Câmbio, ou seja, da quantidade ou valor em graus que cada olho pode se mover, tanto na direção vertical quanto na horizontal. O simulador deve ser utilizado em humanóides, desta forma o sistema visual adotado é baseado na fisiologia humana. É preciso então colocar restrições ao movimento dos olhos. Sendo assim, estabelecemos que para movimentos verticais os olhos devem se mover juntos na mesma direção. Da mesma forma, no movimento horizontal os olhos devem possuir limites máximos para vergência, isto devido também a características humanas. A segunda restrição trata da decisão de quando o Câmbio deverá movimentar apenas seus olhos ou apenas seu pescoço ou

ambos ao mesmo tempo e da definição dos parâmetros de movimento para cada um desses objetos em cada direção/sentido. Isto se traduz na prática em definir o valor do deslocamento e a velocidade/aceleração para cada controlador de movimento físico.

Estabelecidas estas restrições, desenvolvemos uma arquitetura de controle composta por controladores independentes. No item (a) da Figura 3, temos um controlador para cada olho (EC_1 e EC_2), um controlador de vergência (CV) e um controlador central (CC). Cada controlador recebe como entrada os deslocamentos angulares, vertical e horizontal, vindos do controlador de vergência com o objetivo de posicioná-los no ambiente. Na implementação corrente, estes ângulos são limitados em 30º em relação a normal de cada olho. Se o movimento de um olho ultrapassar este limite, um movimento com o pescoço deve ser feito na mesma direção dos olhos para que eles possam alcançar seu objetivo. São mostrados os ângulos horizontais α_1 e α_2 e as normais V_1 e V_2 no item (b) da Figura 3. É importante ressaltar que os limites estabelecidos para os movimentos dos olhos podem ser modificados, até que se chegue a um resultado satisfatório na prática. A razão para que os controladores dos olhos tenham movimentos horizontais independentes é que, geralmente a posição final de vergência requer diferentes ângulos horizontais entre a normal de cada olho, podendo um ser negativo e o outro positivo. O CV é utilizado como apoio para o cálculo da vergência dos olhos. Calcula-se o ângulo para um dos olhos (o dominante) e com esta informação o ângulo para o outro olho. O controlador central é responsável pelo ajuste do pescoço. O algoritmo usado é descrito a seguir:

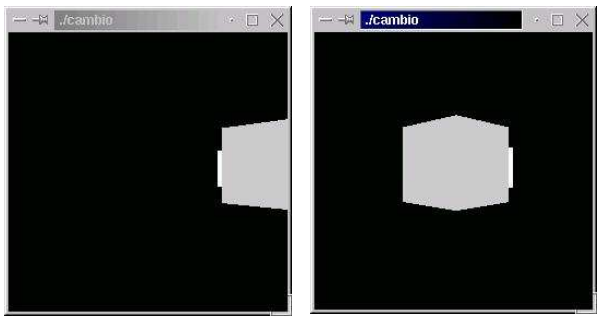
1) Um dos olhos é escolhido para ser o olho dominante, ou seja, o processo atencional define um objetivo e em conseqüência define o olho dominante.
2) Este olho da inicio ao seu movimento para que o ponto de vergência fique bem próximo do centro da imagem (minimização da disparidade).
3) Caso isto requeira um movimento horizontal maior que 30 graus, o pescoço também é movido ao longo desta direção até que o alvo seja alcançado.
4) Depois que o olho dominante foi definido, a posição do outro olho é ajustada através de um *loop*, o qual tenta minimizar a disparidade entre o espaço que está em volta do ponto de vergência.
5) Caso o olho não dominante execute um movimento horizontal que ultrapasse seus limites, todo sistema é posto para girar em relação a este olho, pois se espera que a posição do olho dominante esteja correta (pelo menos próxima do objetivo).

Enquanto o sistema está executando todos os movimentos acima, é também verificado que os eixos óticos V1 e V2 não ultrapassem as posições além de paralelo um do outro, isto quando estiverem abrindo um em relação ao outro. Notamos que fisicamente isto é possível em seres humanos.

5 Experimentos e Demonstrações

Para testar inicialmente o algoritmo de vergência, definimos manualmente os valores de vergência para o olho dominante e colocamos o Câmbio em execução. Figura 4 ilustra uma destas situações, onde a interface com a tarja realçada representa o olho (câmera) dominante. A outra câmera é deixada na posição horizontal, pela restrição do algoritmo de vergência. Os itens (a) e (b) da Figuras 5, ilustram situações finais após a convergência dos controladores, onde os olhos atingem uma posição de vergência

desejada. O experimento mostrado no item (b) da Figura 5, incluiu um objeto com textura e cor. O tempo de processamento aumentou, devido aos vários cálculos extras necessários para a inclusão de textura, mas ainda sendo possível o funcionamento do algoritmo em tempo real. A noção de profundidade pode ser notada nas figuras, bastando fixar cada olho em uma das imagens, para cada par.

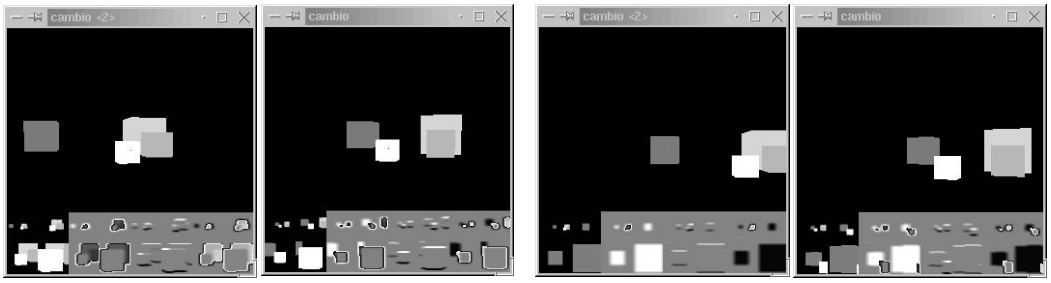


(a) (b)

Figura 4: Olho dominante sendo colocado em um objetivo (olho não dominante em paralelo).

(a) (b)

Figura 5: Situação de vergência atingida pelo algoritmo.

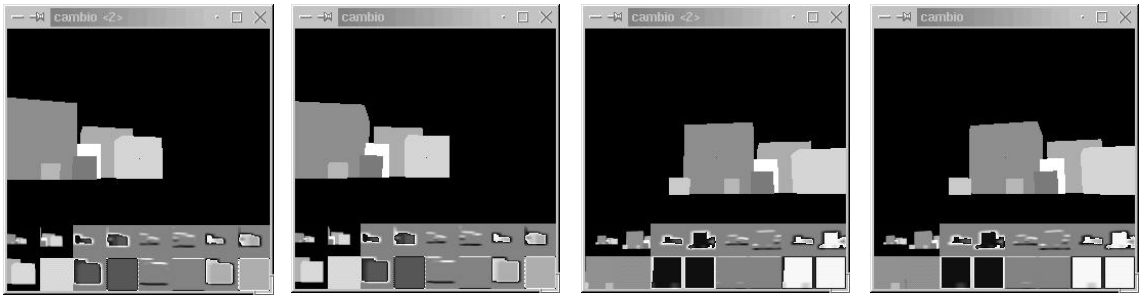


(a) (b)

Figura 6: Situações de vergência atingida pelo algoritmo.

Os itens (a) e (b) da Figura 6, ilustram um outro tipo de experimento, onde o conjunto encontra-se inicialmente centrado num determinado objeto. No caso um cubo bem claro e pequeno pode ser visto ao centro de ambas imagens, no item (a) da Figura 6. Este cubo é o mais próximo de Câmbio no ambiente. O cubo mais distante (mais escuro à esquerda) é então determinado como próximo objetivo de vergência. O item (b) da Figura 6, mostra a situação final, notamos que neste caso há grande diferença de profundidade entre os objetos consequentemente, grande diferença na disparidade

estéreo. Outros experimentos foram realizados com vários objetos incluídos no ambiente. A Figura 7 ilustra um outro experimento do mesmo tipo que o da Figura 6, porém com diferenças menores nas disparidades estéreo (na profundidade entre os objetos).



(a) (b)

Figura 7: Situação inicial e final de vergência para o processo atencional.

6 Conclusão e Trabalhos Futuros

O Câmbio foi desenvolvido usando uma plataforma Pentium™ e sendo executado no Linux. A nossa intenção é desenvolver um código aberto, público e portável ou pelo menos torná-lo portável para as principais plataformas gráficas. Implementamos nesta versão os algoritmos para o cálculo da disparidade e o de vergência, usando diretivas do OpenGL e da linguagem C. Outro propósito a ser atingido com o Câmbio é verificar a aplicação de outras abordagens para a extração de características além dos operadores gaussianos, as quais servirão de base para definir o comportamento do humanóide. Em relação aos objetos, serão testados objetos com diferentes formas, assim como a colocação de texturas mais realísticas nos mesmos. Em relação ao ambiente, testaremos modelos de iluminação e outros modelos de cor. Atualmente já está sendo desenvolvida uma arquitetura de percepção e controle para o Câmbio (fora do escopo deste trabalho), onde estão sendo levados em consideração modelos de movimento como cinemática inversa e direta e a junção da arquitetura de movimento com a visual, visto que o movimento que será realizado pelo humanóide será resultado de informação extraída do ambiente e dependerá da tarefa que ele estiver realizando.

7 Referências

[01] L. Gonçalves and F. Silva. Control mechanisms and local perception to support automomous behavior in virtual animated agents. *Computer & Graphics Journa*l, 25(6):965–982, December 2001.

[02] L. M. G. Gonçalves. Towards a learning model for feature integration in attention control. In *Proc. of IEEE MultiSen-sor Fusion Conference (MFI 2001*), Bade-Baden, Germany, August, 19-22 2001. IEEE Press.

[03] L. M. G. Gonçalves, C. Distante, A. Oliveira, D. Wheeler, and R. A. Grupen. Neural mechanisms for learning of attention control and pattern categorization as basis for robot cognition. In *In Proc. of International Intelligent Robots and Systems Conference (IROS 2000*). IEEE Computer Society, October 30 - November 5 2000.

[04] L. M. G. Gonçalves, R. C. Farias, F. W. Silva, S. C. Botelho, and A. A. Lopes Junior. Control mechanics for computer animated agents. In R. S. W. Luciana P. Nedel, editor, *Proc. of III Brazilian Workshop on Virtual Reality*, pages 193–204, Porto Alegre, RS, October, 16–18 2000. Gráfica EPECÊ Pontifícia Universidade Cat´ olica do Rio Grande do Sul.

[05] L. M. G. Gonçalves and R. A. Grupen. Integrating attention and categorization behaviors in robotics. In *Proc. of the 6th International Conference on Intelligent and Au-tonomous System*s. IEEE Computer Society Press, July, 25-27th 2000.

[06] L. M. G. Gonçalves, R. A. Grupen, A. A. Oliveira, D. Wheeler, and A. Fagg. Tracing patterns and attention: Humanoid robot cognition. *The Intelligent Systems and their Application*s, 15(4):70–77, July/August 2000.

[07] L. M. G. Goncalves and F. W. Silva. Task manipulation and control in artificial animated creatures. In M. Berthoz Floreano Roitblat Wilson, editor, *Proc. of "From Animals to Animats" - VI* International *Conference on the Simulation of Adaptive Behavior (SAB'00*), pages 77–86. ISAB Inc., September, 11-15th 2000. Conference held in Paris, France.

[08] W. E. L. GRIMSON. *From Images to Surfaces: A Computational Study of the Human Early Visual Syste*m. The MIT Press, Cambridge, MA, 1981.

[09] B. K. P. HORN. *Robot Visio*n. MIT Press, 1986.

[10] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Patten Analysis and Machine Intelligenc*e, 20(11):1254– 1259, 1998.

[11] D. Marr. *Vision – A Computational Investigation into the Human Representation and Processing of Visual Informatio*n. The MIT Press, Cambridge, MA, 1982.

[12] M. M. H. Rajesh P.N. Rao, Gregory J. Zelinsky and D. H. Ballard. Eye movements in iconic visual search. *Vision Researc*h, 2002.

[13] R. P. N. Rao and D. Ballard. An active vision architecture based on iconic representations. *Artificial Intelligence Magazin*e, 78(1-2):461–505, October 1995.

[14] I. A. Rybak, V. I. Gusakova, A. V. Golovan, L. N. Pod-ladchikova, and N. A. Shevtsova. A model of attention-guided visual perception and recognition. *Vision Researc*h, 38(2):387–400, 1998.

[15] J. Tsotsos, S. Culhane, W. Wai, Y. Lai, N. Davis, and F. Nu-flo. Modeling visual attention via selective tuning. *Artificial Intelligence Magazin*e, 78(1-2):507–547, October 1995.

[16] S. ULLMAN. *High-level Vision: Object Recognition and Visual Cognition*. The MIT Press, Cambridge, Mas-sachusetts, 1996.

[17] P. Van de Laar, T. Heskes, and S. Gielen. Task-dependent learning of attention. *Neural Network*s, 10(6):981–992, Au-gust 1997.

[18] Meiguins, B. S., Carvalho, S. R., Neto, E. P. B., Neto, N. P. C. N., Martins, S. L. O., Alves, S. L., Meiguins, B. S., Guedes, L. A. *Um Ambiente de Realidade Virtual Multi-usuário Aplicado ao Setor de Turismo.* 4TH SBC Symposium on Virtual Reality. Florianópolis, SC, Brasil - 2001.