Criação de imagem anaglífica

Este será um post mais técnico, que os computeiros irão adorar. Irei disponibilizar o código que desenvolvi em OpenCV (<http://opencv.willowgarage.com/>) para criação de uma imagem anaglífica (vermelho-ciano) a partir de um par estéreo de imagens. O OpenCV é uma ótima biblioteca escrita em C que possui várias funções otimizadas para visão computacional e processamento de imagens. Ele tem sido utilizado em diversos projetos, inclusive para hacking do Kinect (<http://opencv.willowgarage.com/wiki/Kinect)>. Mais para frente, farei um ou mais posts somente para detalhar a biblioteca e seu uso.

Sobre o código

A utilização do OpenCV deixa o código bem simples e facilita o trabalho com imagens. Há uma estrutura já pronta para imagens chamada de IplImage (como a fonte pode confundir um pouco entre i maiúsculo e l minúsculo, entenda: i p L image), na qual podemos definir vários parâmetros, tais como dimensões (altura e comprimento), espaço de cores (grayscale, RGB, YCbCr, etc.), densidade de bits para cada canal de cor (isto é, número de bits para armazenar os valores de cada canal de cor), dentre outros descritos na documentação (http://opencv.willowgarage.com/documentation/basic\_structures.html#iplimage).

O código lê dois argumentos da linha de comando (argv[1] e argv[2]) como sendo o caminho para as duas imagens do par estéreo, sendo argv[1] a imagem correspondente ao lado direito e argv[2] a imagem correspondente ao lado esquerdo. Logo após, faço um tratamento de erros bem simples, verificando se os arquivos puderam ser abertos e se as imagens possuem mesma dimensão. Uma ressalva para o carregamento das imagens: o OpenCv as carrega no formato RGB, porém, com ordem alterada dos canais, ficando de fato sendo BGR. Isto é importante na hora de percorrermos a matriz que representa a imagem.

Partimos então para dois loops for, percorrendo coluna a coluna e depois linha a linha. Para cada linha, utilizam-se dois ponteiros, um para cada imagem. Observem que o valor atribuído aos ponteiros é feito através de um cálculo mais complexo do que um row++. É um pouco difícil de explicar isso sem detalhar a estrutura do IplImage, então peço para que tenham fé ☺ pois é desse jeito que tem que ser feito para se ter certeza de ter passado corretamente para a próxima linha.

A estrutura do IplImage armazena em apenas uma matriz os valores associados a cada canal, diferente de estruturas clássicas que criam uma matriz para cada canal. Sendo assim, a cada três colunas teremos as informações correspondentes a um pixel, ou seja, teremos sempre os valores dessa maneira: B1G1R1, B2G2R2, B3G3R3, e assim por diante. Tendo isso em vista, zeramos os valores dos canais azul e verde na imagem correspondente ao lado direito (primeira e segunda colunas de cada pixel) e zeramos também os valores do canal vermelho na imagem correspondente ao lado esquerdo (terceira coluna de cada pixel).

Feito isso, criamos uma nova estrutura IplImage para armazenar a junção do par estéreo, copiando as propriedades de dimensão, densidade de bits e número de canal de uma das imagens. Logo após, utilizamos a função cvAdd, cujos parâmetros são a imagem direita, esquerda e a nova, e por fim, salvamos a nova imagem, em formato BMP. Mantendo uma boa prática de programação, liberamos os ponteiros utilizados antes de sair do programa.

Captura das imagens

O ideal seria ter duas câmeras acopladas e calibradas para fazer a captura de uma cena. Como isso é caro e demanda tempo, usamos o jeitinho brasileiro. O problema com isso é a imperfeição de não conseguir capturar duas imagens ao mesmo tempo, ou seja, o lugar e os elementos dos quais queremos tirar a foto devem ser estáticos. Outro problema é com relação à posição da câmera: ela deve deslocar-se apenas horizontalmente, mantendo a mesma altura e ângulo de visão. Além disso, temos que levar em conta a qualidade da imagem criada pela câmera e se internamente já não é feita alguma compressão, como por exemplo, conversão para JPEG, o que pode interferir na transformação. Isso torna a conversão para anaglífico um pouco difícil e de baixa qualidade, porém, com teimosia você obtém uma imagem com ótimo efeito.

Sugiro utilizar uma câmera com tripé em um piso plano. Faça uma linha reta no chão e marque as duas posições onde a câmera deve ficar na hora de bater a foto. Durante o deslocamento da câmera, tenha o máximo de cuidado para não movimentá-la no eixo vertical, alterando o ângulo de visão.

Outra solução é utilizar o Google e fazer uma busca de imagens por “estereo pair”.

Utilização do programa

Para usuários Windows: Abra uma janela de comando e vá até a pasta onde você descompactou o pacote RAR. Digite então o nome do programa, o caminho para a imagem da direita e o caminho para a imagem da esquerda e aperte ENTER. Uma dica é deixar as duas imagens na mesma pasta em que se encontra o programa, desse jeito, basta apenas digitar o nome das imagens. Veja a Figura 1 para ter um exemplo de execução.

Exemplos

Seguem abaixo duas imagens que eu fiz com a minha câmera aqui em casa. Notem que sou pobre, e a qualidade da minha câmera é péssima. Mesmo assim, obtive imagens com efeito estereoscópico, porém de baixíssima qualidade, admito.

Para dúvidas no código ou execução, deixe um comentário.