Universidade Federal do Rio Grande do Sul
Instituto de Informática
Departamento de Informática Aplicada

Disciplina: INF01046 - Fundamentos de Processamento de Imagens
Professor: Manuel M. Oliveira
Aluno: Kauan Karlinski Rakoski (00588935)

IMPLEMENTAÇÃO DE UM SISTEMA GRÁFICO QUE REALIZA OPERAÇÕES BÁSICAS SOBRE IMAGENS DIGITAIS

Objetivo

O presente relatório discorre sobre a implementação de um aplicativo desktop que permite ao usuário realizar operações básicas sobre imagens, a saber:

- Carregar arquivos de imagens
- Salvar arquivos de imagens processadas
- Espelhamento vertical e horizontal de imagens
- Conversão de imagem colorida para tons de cinza
- Quantização de tons em imagens em tons de cinza

Tal aplicativo foi desenvolvido em C++ com o uso da biblioteca Qt na sua versão 5.15. Os próximos tópicos serão estruturados de maneira a satisfazer os requisitos descritos.

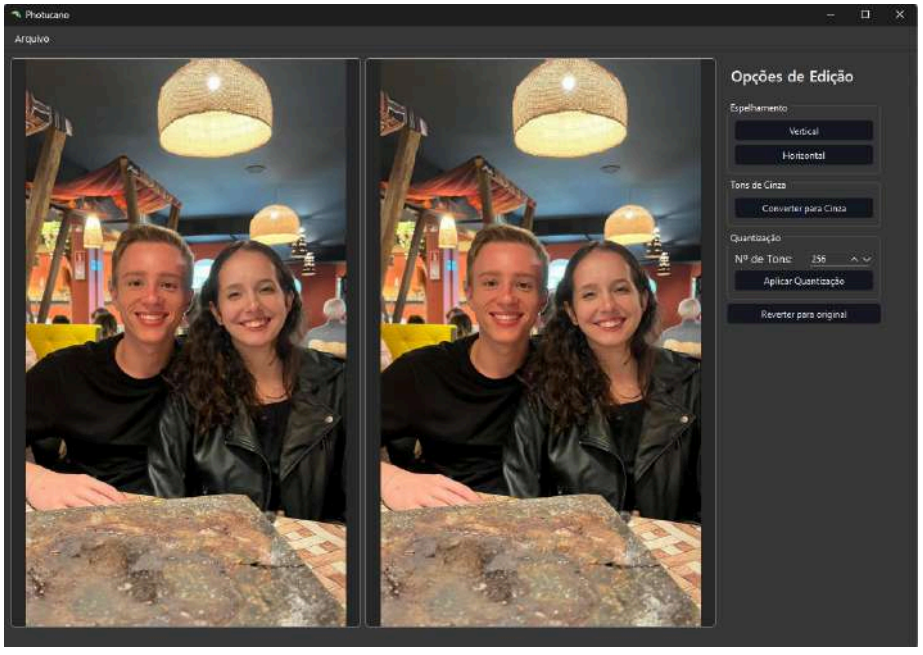

FIGURA 1 - VISUALIZAÇÃO DA INTERFACE BÁSICA DO APLICATIVO

1) Escreva um programa para ler arquivos de imagens e regravá-los com um outro nome. Esta tarefa simples tem o objetivo de familiarizá-la(o) com o uso de bibliotecas para leitura e gravação de arquivos. O seu programa deve suportar pelo menos o formato JPEG. Teste o seu programa com imagens JPEG (e.g., utilize as imagens disponibilizadas para o trabalho). Para essas imagens, verifique seus tamanhos em cada imagem do par (original e arquivo gravado). Você percebe alguma diferença visual entre elas? Alguma diferença nos tamanhos dos arquivos? Caso haja diferença nos tamanhos de arquivos, faça uma pequena pesquisa na web sobre arquivos JPEG e tente explicar a causa da diferença observada.

Para este teste, foi utilizada a imagem Space_187k. Analisando as propriedades da imagem na interface do windows 11, verificamos que:

1. Tamanho da imagem original: 186 KB (191.279 bytes)
2. Tamanho da imagem salvada: 109 KB (112.059 bytes)

Visando justificar esse fato, temos que observar que o JPEG é um formato que é comprimido e descomprimido, e acaba sendo lossy - ou seja, dependendo do formato de compressão, pode ficar menor em troca de perda de qualidade. No caso específico, a diferença pode ser atribuída ao uso de algoritmo mais eficiente no aplicativo, que consegue comprimir melhor. Devemos considerar também que a imagem foi comprimida utilizando QImage.save(“imagem_editada”, "JPG", 95)), ou seja, com parâmetro de qualidade 95%, um pouco inferior que a original. Visualmente, ao menos no monitor utilizado (AOC 23.8” 1920×1080), não é possível notar diferença.

FIGURA 2 - COMPARAÇÃO DO ARQUIVO ORIGINAL (ESQUERDA) COM O SALVO SEM MODIFICAÇÕES (DIREITA)

2) Análise da implementação das operações

1. Carregamento e salvamento de imagens: implementado e funcional. No passo anterior já demonstramos que é possível carregar e salvar imagens, inclusive averiguando a diferença de tamanho em uma sem modificações. Detalhando, dizemos que é possível carregar imagens do formato .png, .jpg, .jpeg e .bmp e salvar a imagem com qualquer modificação em formato .jpg.

2. Espelhamento vertical e horizontal: implementado e funcional. Funciona adequadamente tanto para número de espelhamentos pares quanto ímpares. Abaixo, exibe-se exemplos de espelhamentos em número ímpar. O caso de número par de espelhamentos foi suprimido de exibição, haja vista que se assemelha a nenhum espelhamento. Vale ressaltar que, como esperado, os espelhamentos são cumulativos.

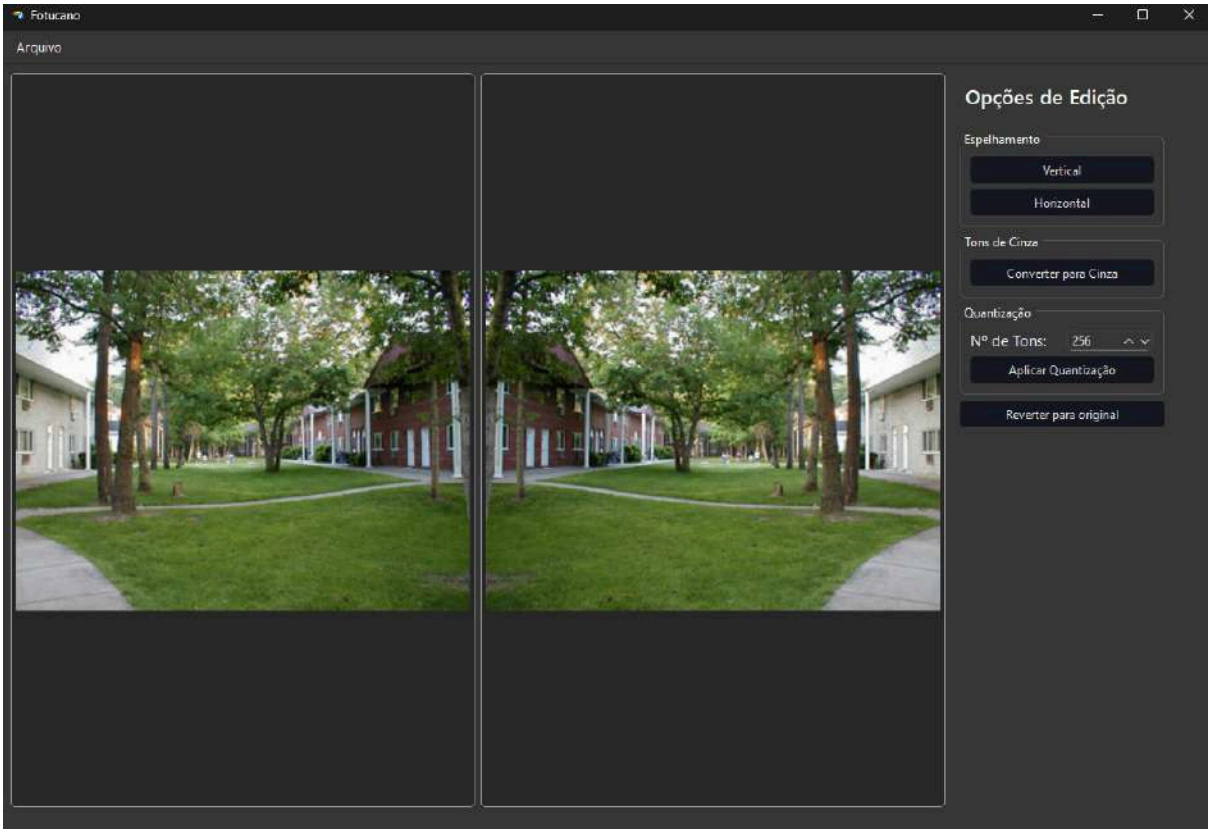

FIGURA 3 - ESPELHAMENTO HORIZONTAL DE IMAGEM

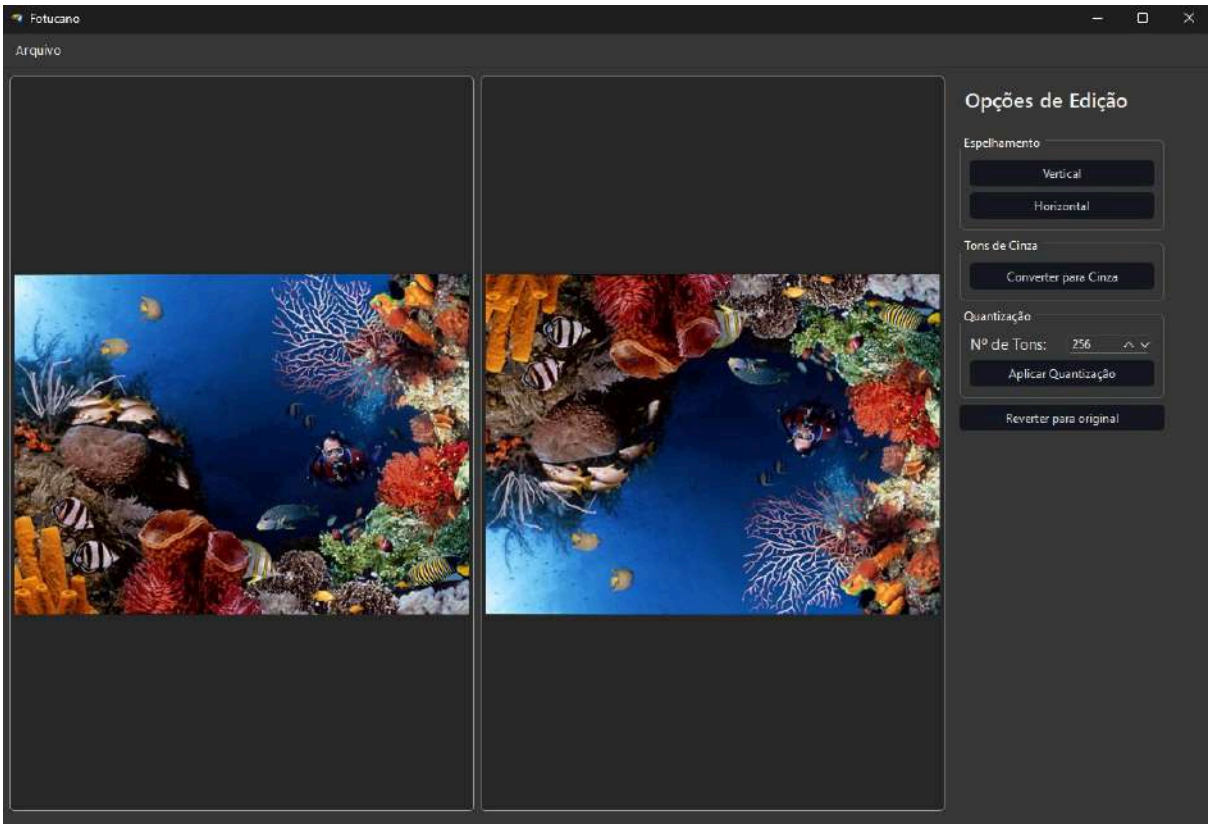

FIGURA 4 - ESPELHAMENTO VERTICAL DE IMAGEM

3. Conversão de imagens para tons de cinza: implementado e funcional. Cumulativa sobre as demais operações. Ao tentar converter uma imagem já em tons de cinza para tons de cinza, não há mudança visual, haja vista que L = 0.299*R + 0.587*G + 0.114*B, se R = G = B = L', torna-se L = (0.299 + 0.587 + 0.114)L', ou seja, L = L'.

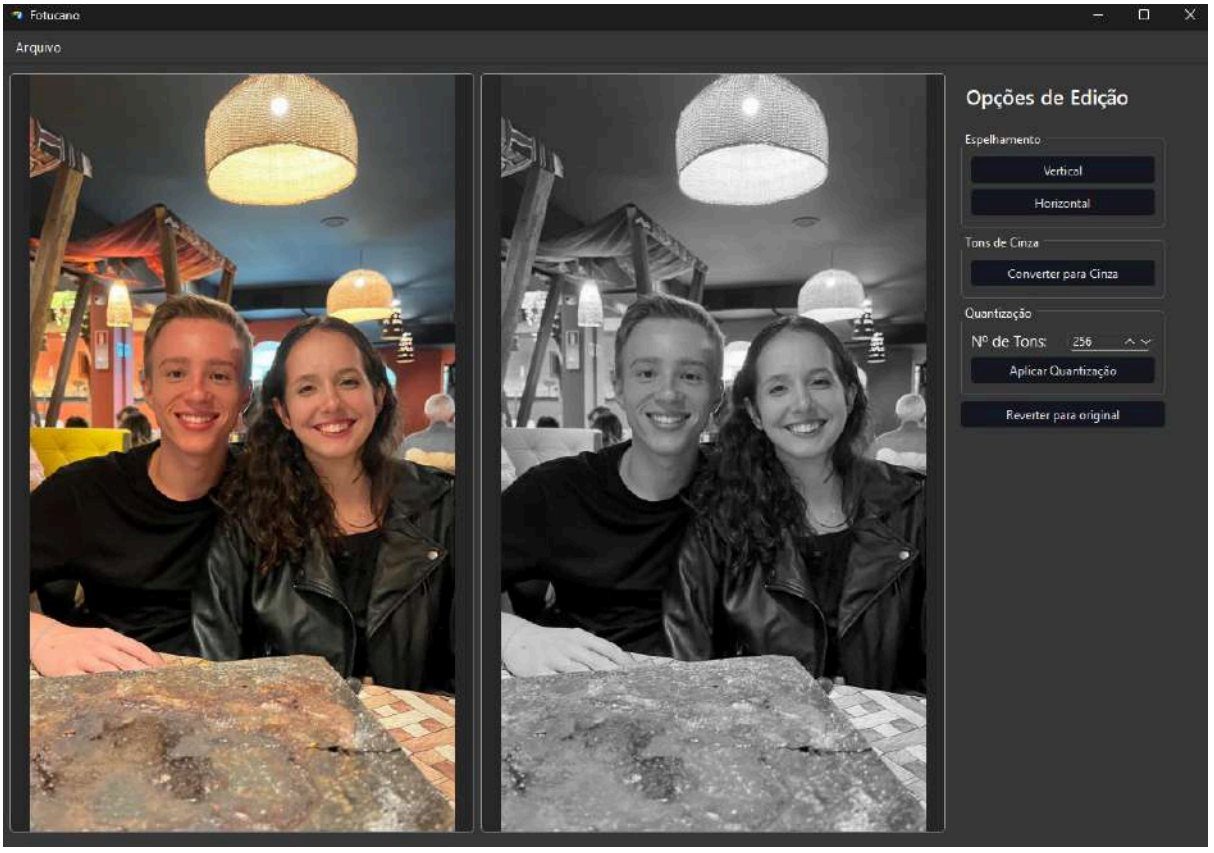

FIGURA 5 - CONVERSÃO PARA TONS DE CINZA DE IMAGEM COLORIDA

4. Quantização de tons sob imagens em tons de cinza: implementado e funcional. Quantiza-se as imagens a partir do algoritmo proposto na especificação do trabalho. Se a imagem for colorida, ela é transformada em tons de cinza antes da quantização. Na próxima página, mostra-se a quantização da imagem de uma águia - obtida em Unsplash - em 64 e 6 tons, respectivamente. A quantização é feita a partir da imagem colorida para demonstrar o funcionamento.

   Observação: o algoritmo utilizado para a quantização (implementado a partir da descrição do trabalho) por se basear na faixa de valores em vez da quantidade "real" de tons, ao ser aplicado sucessivas vezes produz a cada nova aplicação resultados mais agressivos de quantização.

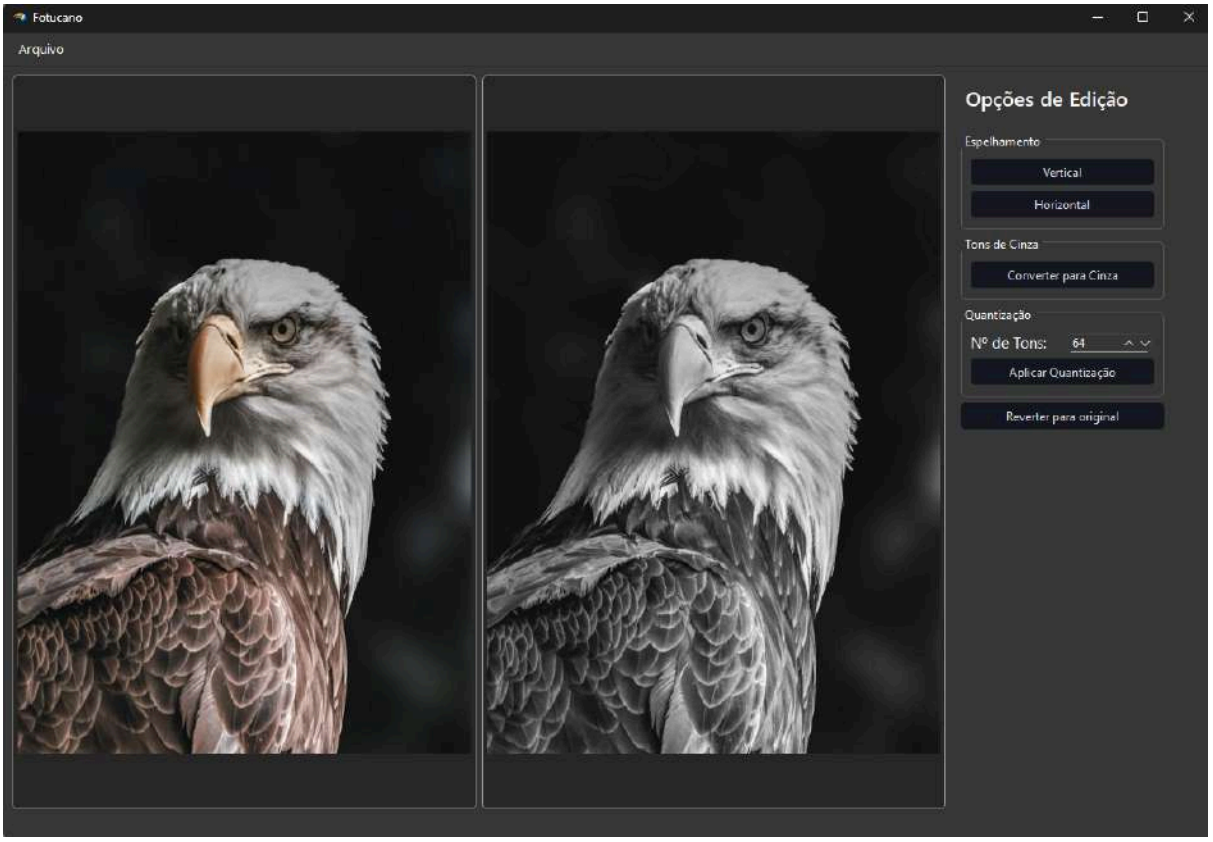

FIGURA 6 - QUANTIZAÇÃO DE IMAGEM EM 64 TONS

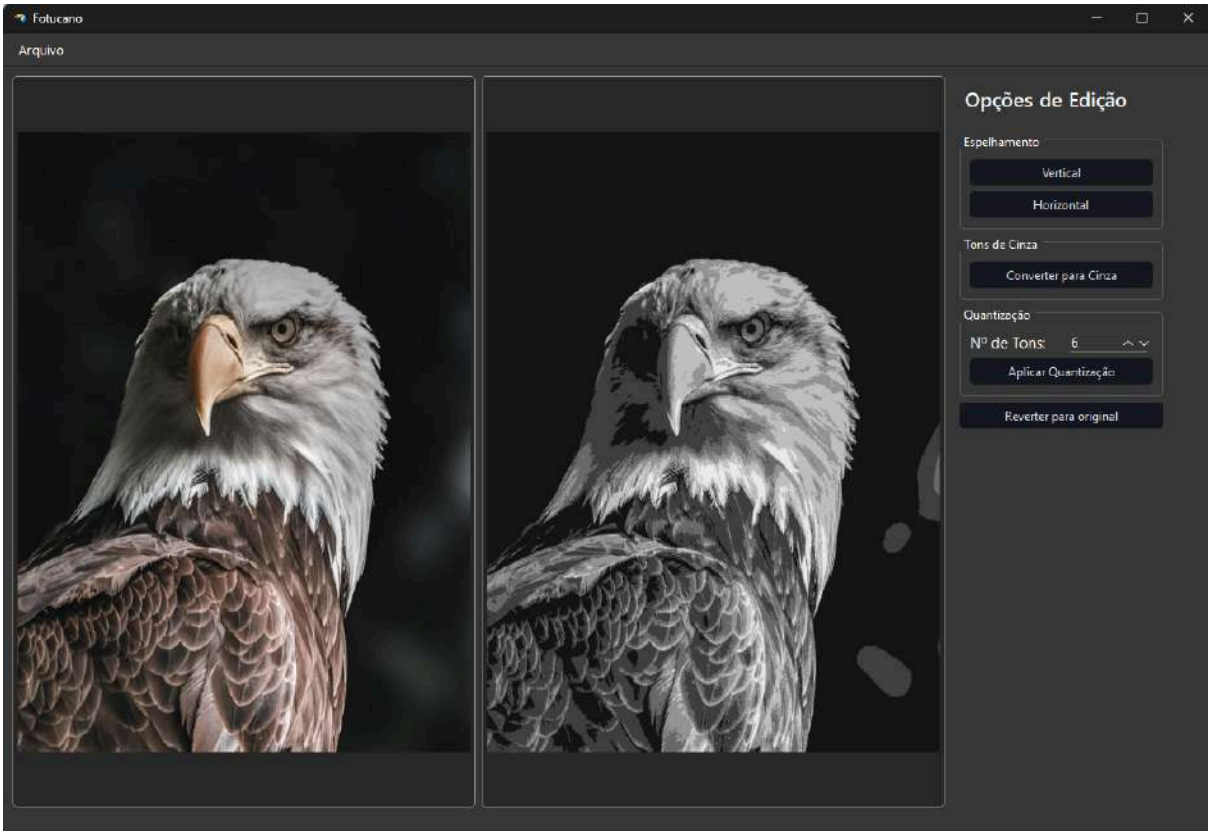

FIGURA 7 - QUANTIZAÇÃO DE IMAGEM EM 6 TONS

Conclusão

O presente trabalho foi extremamente proveitoso. Antes de iniciar a disciplina, meu conhecimento sobre como imagens digitais eram manipuladas era basicamente nulo e poder compreender e aplicar métodos (mesmo que básicos) de manipulação de imagens é um processo muito satisfatório. Aliado a esse fato, o Qt oferece abstrações muito poderosas, de maneira que construir a interface gráfica utilizando seus componentes se torna uma tarefa quase trivial e imediata.

Desse modo, acredito que a maior dificuldade enfrentada neste processo foi justamente a utilização da linguagem C++. Apesar de possuir familiaridade com C, o uso mais idiomático de C++ e de suas abstrações de Orientação a Objetos tem uma notação e organização que ainda julgo um tanto confusa. Desse modo, dificilmente o código produzido possui excelência ou segue boas práticas de programação. Como parte de meu objetivo é aprender a linguagem e a orientação a objetos, julgo como um problema temporário e de não tanta relevância para o escopo deste projeto - que é individual.

Acredito que o desenvolvimento seguiu um bom rumo e ensinou bastante. Senti-me bastante motivado para a execução do projeto e acredito que não existiram erros, apenas aprendizados. Arrependo-me de pouco - em verdade, apenas me arrependo de ter visto a solução para alguns problemas cedo demais e não ter me dado o devido tempo para construí-la intuitivamente com o nível de esforço que deveria ter utilizado.

Bibliografia

Todo o código, assim como o executável do trabalho está disponível em <https://github.com/KauanRakoski/fotucano>