UNIVERSIDADE NOVA DE LISBOA

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

Computacão Multimédia

Design Wall for Meeting Rooms

Prof. Nuno Correia

Ana Maissa Antunes 61025

Miguel Vieira de Almeida 60490

CML 2023/24

1. Introdução

Desde a segunda metade do século XX que começaram a surgir os primeiros *Displays* Interativos, mas o seu desenvolvimento continua até aos dias de hoje expandido a sua diversidade. No início da década de 1990 esta tecnologia chegou às escolas e empresas. (1)

Apesar de nos seus primórdios já terem explorado diferentes formas de contribuir para essa interação como através de painéis, tecnologias de toque inteligente, projetores interativos. Apenas no início do século XXI, os *touch screens* começaram a ganhar popularidade, e pouco depois começaram a ser desenvolvidas tecnologias com *multi-touch screens* em largas dimensões como *Interactive Design Wall Displays.* (2) Este último contém uma parede que incorpora uma tecnologia interativa permitindo uma interação direta dos utilizadores com o conteúdo. Esta ferramenta que pode ser utilizada para entretenimento ou trabalho contribui a facilitar na interação, colaboração, ou visualização final de um produto, de forma simples em equipa para chegar ao design mais adequado. (3)

De seguida, iremos abordar e explorar alguns destes exemplos desenvolvidos nos últimos anos. Vamos também identificar os pontos positivos de cada tecnologia, utilidade, e quais as suas possíveis melhorias. Por fim, faremos também uma breve análise e paralelismo relativamente ao estado atual da arte nesta área, acrescentando uma visão futura.

2. Trabalhos Relacionados

2.1 *Investigating Collaborative Exploration of Design Alternatives on a Wall-Sized Display*

Este artigo é relativo a uma investigação que foi feita para estudar as vantagens da utilização da ferramenta “Wall-Sized Display”. Foram comparadas duas ferramentas *ShapeCompare* e *ShapeSlide* que permitem a interação do utilizador através do ecrã *touch*. *Shape Compare* é um display evoluído que permite chegar ao modelo pretendido de forma mais rápida e simples. Este explora alternativas relativamente ao objeto, e vai-se adaptando à medida que o utilizador seleciona o pretendido, esta simplicidade faz com que qualquer o utilizador mesmo sem saber utilizar a ferramenta “CAD”, consiga chegar intuitivamente ao resultado pretendido. Esta estrutura contribui para uma interação direta com o utilizador, que também está integrada no nosso projeto.

2.2 *Increasing Passersby Engagement with Public Large Interactive Displays: A Study of Proxemics and Conation*

Este artigo menciona um projeto que foi realizado com o intuito de perceber em que sentido pistas visuais afetariam a utilização de um ecrã público. Aa interfaces usadas neste projeto foram criadas de modo a revelar informação progressivamente e encorajar a interação através do uso de conteúdo visual destinado a criar motivação direta ou indireta no utilizador. Deste projeto retiramos principalmente o

uso da câmera para detectar se existe alguma pessoa à frente do ecrã público e se existir, começar a apresentar informação (no primeiro do vídeo do artigo, quando o utilizador sai do alcance da câmera, os vários livros que estavam abertos fecham e quando a pessoa volta a estar no alcance da câmera do dispositivo, os livros começam a abrir de novo. Estas características assemelham-se a algumas funcionalidades da "Design wall for meeting rooms", nomeadamente na deteção de gestos e caras.

2.3 *DesignAR: Immersive 3D-Modeling Combining Augmented Reality with Interactive Displays*

Um desenvolvimento já atual, mas com muito potencial futuro é a combinação da realidade aumentada com Interactive Displays. Este consiste num Design projetado em realidade aumentada para a criação de modelos 3D. Um modelo 3D está perfeitamente alinhado ao ecrã e é gerado com realidade aumentada, podendo ser visualizada através dos óculos VR[1]. Este modelo contém uma interação com o utilizador muito interessante. Os utilizadores conseguem de forma simples criar novos modelos, através da seleção do modelo pretendido, ou do desenho direto do mesmo, através da uma caneta *touch*. Conseguem aplicar transformações, rotações, alterações de escala no modelo de realidade aumentada, e ainda diferentes opções de *display.* Deste modo, pouparemos trabalho e tempo, por exemplo no desenvolvimento de protótipos de uma empresa. Esta tecnologia, tem algumas semelhanças relativamente ao nosso trabalho "Design wall for meeting rooms". Algumas delas são no Display, através da manipulação da imagem com base em algumas características, como o seu tamanho. Estas especificações têm também uma pequena interface (no nosso caso com dados extraídos do XML) para facilitar a interação com o utilizador.

3. Algoritmos e Técnicas

Extração de Metadados

Para a extração de metadados, utilizamos várias técnicas que processam as imagens e extraem informações relevantes. Cada técnica é implementada na classe "xml_algorithms", e os resultados são armazenados num formato XML estruturado. Cada imagem ou primeiro frame do vídeo, tem um XML assim associado com os seus metadados.

[1] Virtual Reality

1. Cores

Para a análise de cores, calculamos a cor média da imagem em formato RGB. O formato RGB é escolhido por representar diretamente as três cores primárias. A média das cores é calculada somando os valores RGB de todos os pixels e dividindo pelo número total de pixels.

2. Luminância

A luminância, que representa a luz percebida na imagem, é calculada utilizando um raciocínio semelhante ao das cores, exceto que utilizámos a fórmula presente nos slides para calcular a "*luminance*" através dos RGB.

3. Contornos

Para a deteção de contornos, aplicamos individualmente os 5 filtros definidos nos slides para as diferentes direções, incluindo um para contornos não direcionais. Através da função "*filter2D"* obtemos a imagem resultante após a aplicação do filtro. Para cada imagem resultante calculamos 2 parâmetros, a média e o desvio padrão. Calculámos a média, e o desvio padrão com a fórmula dos slides utilizando a luminância de cada pixel. Por fim, adicionamos ao vetor de contornos da média ou ao outro vetor de contornos para o desvio padrão. Logo, cada vetor ficou com 5 posições.

4. Texturas

Para a análise de texturas, utilizámos um raciocínio semelhante ao dos contornos, mas com outros filtros. A função "*getGaborKernel*" cria os filtros, e nós atribuímos 6 ângulos diferentes, ou seja, uma direção diferente para cada filtro. Calculamos também a média e o desvio padrão da luminância para as texturas. Logo em vez de 5 posições por vetor como nos contornos, ficámos com 6.

5. Detecção de Faces

Para detectar e contar o número de faces numa imagem, utilizámos as bibliotecas do OpenCV, especificamente a função "*findHaarObjects*" que aplica o algoritmo de detecção de faces baseado em cascata de Haar. Este algoritmo, no nosso caso com base no ficheiro "*haarcascade_frontalface_default.xml"* identifica as regiões da imagem que apresentam faces. Por fim, através da propriedade "*blobs.size()*" retornámos o número de faces detectadas, guardando num inteiro, para cada imagem/*frame* do vídeo.

6. Número de Objetos

Para a comparação de objetos entre uma imagem ou um frame do vídeo, e uma imagem de referência, utilizamos a classe "ORB" para detectar e descrever *keypoints*. Em seguida, utilizamos o "BFMatcher" com a heurística "knnMatch" para encontrar correspondências entre os *keypoints* das duas imagens.

Esta heurística permite retirar apenas os *"good_matches"*, e com isso, contarmos esse número de elementos.

Deteção de Faces em tempo real

De modo a conseguirmos extrair cada frame em tempo real da câmera, verificámos se havia um frame em tempo real, caso exista, extraímos os pixeis do elemento "*vidGrabber*". Deste modo, podemos aplicar em seguida a função *"findHaarObjects"*, para determinar os *blobs*, mais especificamente, neste caso, as faces reconhecidas na imagem. Por fim, no *"draw"*, desenhamos o retângulo para cada *blob*, para ser visível para o utilizador.

Deteção de Movimento

A detecção de movimento é feita utilizando a técnica de subtração de fundo, onde o background é subtraído da imagem completa para identificar mudanças, através da função "*absDiff*". Além disso, utilizamos a função "*findContours*" do OpenCV para determinar os contornos visíveis em frente ao background. Comparámos as posições dos blobs para realizar 2 funcionalidades diferentes de acordo com o movimento do utilizador. Caso o utilizador coloque a mão numa posição específica, ilustrada a verde, a interface é colocada em *fullscreen*, caso se tenha movido a posição do blob, comparadas através da extração do centro do retângulo, abrimos a janela de deteção de faces em tempo real.

4. Implementação da Aplicação

Grelha de Imagens e Vídeos

Existem 4 vetores que guardam imagens e vídeos na nossa aplicação sendo que dois deles são usados para guardar todas as imagens e vídeos que estão guardadas na pasta “bin/data” e os outros dois vetores são usados para apenas possuírem as imagens que cumpram os filtros escolhidos pelo utilizador. Dito isto, o nosso procedimento no desenho das imagens e vídeos em grelha foi, se um dos “gui” tiverem abertos após selecionar um dos botões da "tab” superior, então são as imagens e vídeos que estão dentro do vetor filtrado de ambos que é renderizado no ecrâ. Caso contrário, todas as imagens e vídeos são renderizados. Pode se realocar as imagens para qualquer sítio dentro das margens do nosso projeto. Também há a possibilidade de remover a seleção da imagem ou do vídeo.

Interação do utilizador com imagens e vídeos

Os utilizadores podem interagir com a “Design Wall”, através de uma interface gráfica. Esta interface permite selecionar funcionalidades de visualização, como expandir a câmera em tempo real, detectar movimento e uma interação direta com as imagens como vídeo. Além disso, permite visualizar os metadados no canto superior esquerdo, filtrá-los no canto superior e colocar o ecrã em *fullScreen*.

Mais especificamente na implementação das imagens e vídeos, nós guardamos estes itens em 2 vetores distintos, para facilitar a sua manipulação. O utilizador pode clicar numa imagem ou vídeo e expandi-lo, e no caso do vídeo, dar *“play”* quando está expandido, ou pausar, tal como indicado no manual do utilizador. Para expandir uma imagem ou vídeo, a implementação é feita com base no armazenamento de uma variável, tanto para o vídeo como para a imagem, que guarda a posição do vídeo/imagem que foi selecionado pelo utilizador. Para sabermos qual foi o item selecionado pelo utilizador, utilizamos as coordenadas da janela extraídas pelo evento default “mouseP*ressed*”, e comparamos com as coordenadas de cada item inicialmente guardadas num vetor, “video_coordinates” no caso do vídeo, e “image_coordinates”, no caso da imagem.

Além disso, temos a funcionalidade de mover uma imagem ou vídeo para o local que se pretende deslizando o rato, função *“mouseDragged”.* Quando efetua o *“mouseReleased”*, o item é lá posicionado. Quando, em seguida, o utilizador clica noutro item, a anterior volta a estar alinha no *grid* como no estado inicial.

Devido a questões de visualização colocamos condições para uma imagem ou vídeo não estar cortada, caso o eixo x ou y exceda um determinado valor.

Criação do XML e atualização da Interface

Quando o utilizador abre a nossa aplicação, é primeiro feita uma busca pelos vídeos que se encontram na pasta de imagens e vídeos e verifica-se se já foi extraído o primeiro frame do vídeo. Em seguida, é feita uma verificação se já existem ficheiros do tipo .xml para cada uma das imagens e vídeos que se encontram dentro da pasta “bin/data”. Se já existir ficheiro .xml, então não é executado nenhum código relacionado com a criação de um ficheiro XML. Caso contrário, é primeiro feito uma análise de cada uma

das imagens que não possuem XML, obtendo então os valores dos metadata (cor, luminescência, texturas, cortes, número de caras e número de vezes que um objeto específico aparece na imagem) e que depois são guardados num novo ficheiro XML dentro da pasta “bin/metadata”. Em seguida é executado o mesmo processo, mas para os vídeos em que é utilizado o primeiro frame do vídeo anteriormente extraído e é tratado como uma imagem para calcular os valores a inserir no ficheiro XML associado ao vídeo em estudo.

Acionamento da Tela Cheia

Na página principal existe um *gui* de nome settings que possui duas funções, sendo que uma delas tem como funcionalidade aumentar o ecrã da aplicação com o objetivo de preencher todo o ecrâ disponível. No nosso caso, o ecrã inicialmente tem uma resolução igual a 1024x768. Quando o botão de tela cheia é premido, o ecrã então fica com uma resolução de 1920x1080. Há a possibilidade de trocar constantemente entre estas duas opções, permitindo ao utilizador que use a aplicação conforme o seu gosto.

5. Descrição das Classes

Na classe principal "ofApp.cpp" nós temos as funções default do open frameworks "ofBook". Além disso, adicionámos duas estruturas para armazenar os vídeos e as imagens, cada uma associada a um "path", correspondente ao caminho até à mesma. Nesta classe colocámos todas as funções relativas à interação direta do utilizador com a interface gráfica, tanto os conteúdos do XML, como a manipulação das câmeras, imagens e vídeos.

A classe descrita anteriormente, comunica com apenas mais uma exterior, a "xml_algorithms.cpp". Esta segunda contém os algoritmos para extrair os metadados das imagens e do primeiro frame de cada vídeo. As classes "ofApp.h" e "xml_algorithms.h" definem as funções e variáveis utilizadas nas classes anteriormente descritas.

Por fim, a classe "Main" que está por default no template do Open Frameworks, interage diretamente com a "ofApp.cpp".

6. Conclusões

Este projeto demonstra como a tecnologia interativa pode transformar a forma como trabalhamos e colaboramos em ambientes de design. As capacidades de visualização e manipulação de grandes volumes de dados visuais, juntamente com uma interface intuitiva e funcional, tornam esta ferramenta uma adição valiosa para qualquer espaço de reunião numa parede interativa.

7. Referências

7.1 Websites

(1) https://www.getcleartouch.com/what-is-an-interactive-display/
(2) https://einfratech.com/the-evolution-of-interactive-touch-screens/
(3) https://www.intuiface.com/blog/5-best-interactive-wall-examples

7.2 Artigos

Patrick Bourdot , Olivier Gladin, Cédric Fleury, Patrick Bourdot, 2020
https://dl.acm.org/doi/10.1145/3313831.3376736
Último Acesso: 28 Abril 2024

Mojgan Ghare, Marvin Pafla,Caroline Wong,James R. Wallace, James R. Wallace, Stacey D. Scott, 2018
https://dl.acm.org/doi/10.1145/3279778.3279789
Último Acesso: 28 Abril 2024

Patrick Reipschläger**,** Raimund Dachselt, 2019
https://dl.acm.org/doi/10.1145/3343055.3359718
Último Acesso: 28 Abril 2024

APPENDIX A: Diagrama de Classes

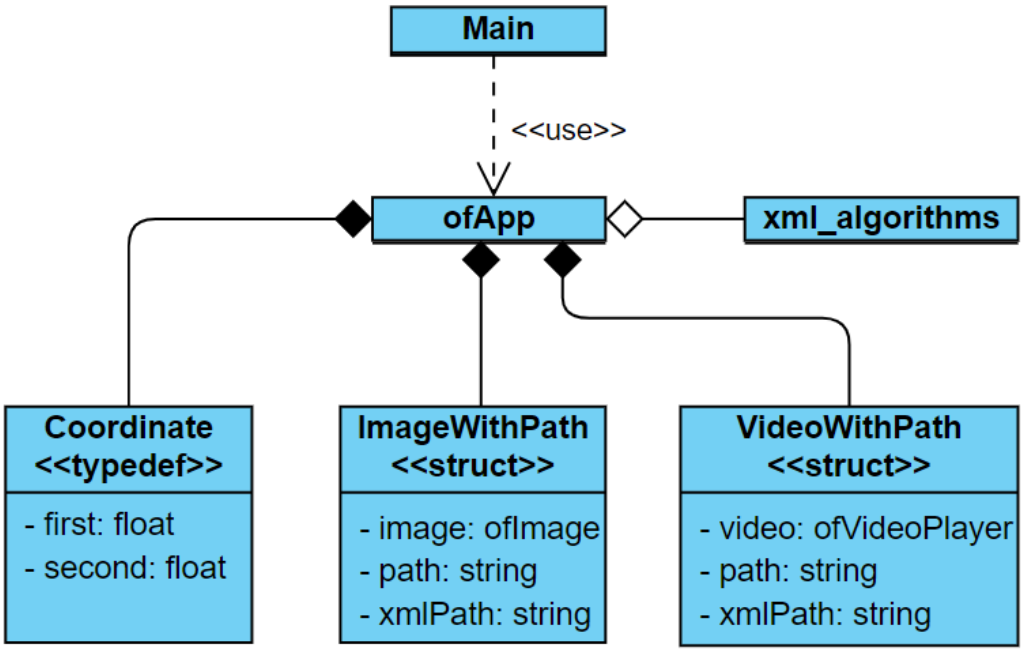

Figura 1: Diagrama de Classes

NOTA: Decidimos não colocar os atributos e métodos das classes “ofApp” e “xml_algorithms”, pois estes impediam a uma visão clara e excediam o tamanho ideal pretendido.

APPENDIX B: Manual do Utilizador

Interação com o teclado e rato

Funcionalidades das Teclas

- "p" - Esta tecla permite ao utilizador dar "play" ao vídeo clicado.
- "c" - Esta tecla permite ativar a câmara em tempo real a detectar faces, para desativá-la basta clicar novamente nesta tecla
- "+" - Esta tecla permite ativar as câmaras de deteção de movimento, para desativá-la basta clicar novamente nesta tecla
- "-" - Esta tecla permite limpar o "foreground" e manter o background
- " " - O espaço permite parar um vídeo que esteja a decorrer, para reiniciar um vídeo que esteja parado, clique novamente nesta tecla

Funcionalidades do Rato

- #### Expandir imagem ou vídeo

  O utilizador pode visualizar um vídeo ou imagem expandida ao clicar em cima do item que pretende visualizar

- #### Mover imagem ou vídeo

  O utilizador após selecionar um item pode mover com o rato e largá-lo numa posição à sua escolha

Funcionalidades da Interface com os Metadados

Barra Superior

- Aba "Tags"
  - O utilizador escreve uma possível *tag* que poderá estar associada a uma imagem e vídeo.
  - Em seguida, pressiona no botão "Apply Filters" para obter os resultados.

- Aba "Luminance"
    - Utilizador seleciona um valor utilizando o *slider* associado à luminescência.
    - Em seguida, pressiona no botão "Apply Filters" para obter os resultados.
- Aba "Color"
    - Utilizador seleciona valores utilizando *sliders* para cada um dos atributos da cor (R, G e B).
    - Em seguida, pressiona no botão "Apply Filters" para obter os resultados.
- Aba "Face Count"
    - Utilizador seleciona valores utilizando o *slider* associado ao número de caras presentes numa imagem ou vídeo.
    - Em seguida, pressiona no botão "Apply Filters" para obter os resultados
- Aba "Object Count"
    - Utilizador seleciona valores utilizando o *slider* associado ao número de objetos presentes numa imagem ou vídeo.
- Aba "Edges"
    - Utilizador seleciona valores utilizando *sliders* para a média de cortes e para o desvio padrão.
    - Em seguida, pressiona no botão "Apply Filters" para obter os resultados
- Aba "Textures"
    - Utilizador seleciona valores utilizando *sliders* para a média de texturas e para o desvio padrão.
    - Em seguida, pressiona no botão "Apply Filters" para obter os resultados

Pode expandir os metadados do XML, clicando no "+" na barra com o nome "XML Info", ou as "Settings", clicando no "+" desta aba. O mesmo serve para a diminui-la.

APPENDIX C: Fotos do Manual

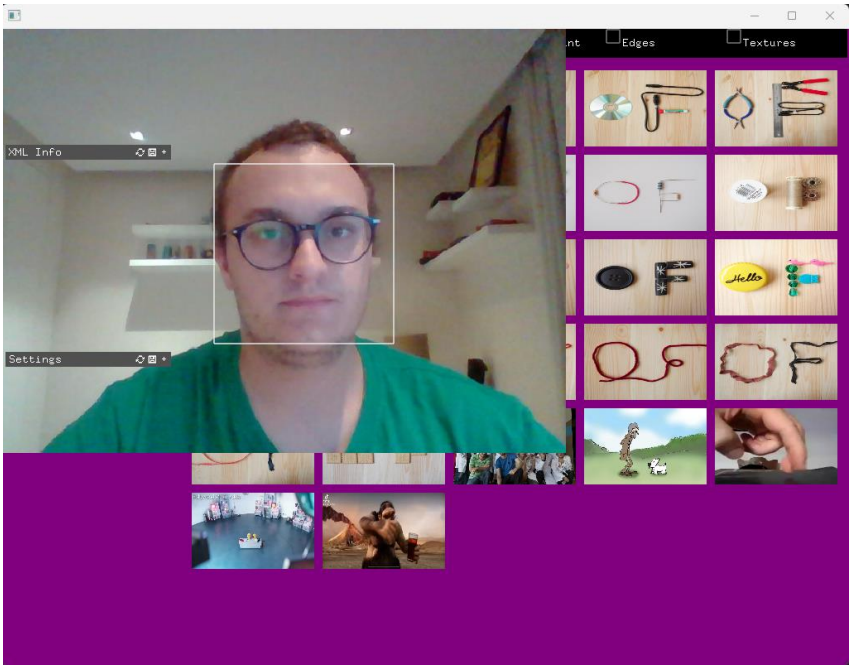

Figura 1: Interface após o utilizador premir a tecla “c”

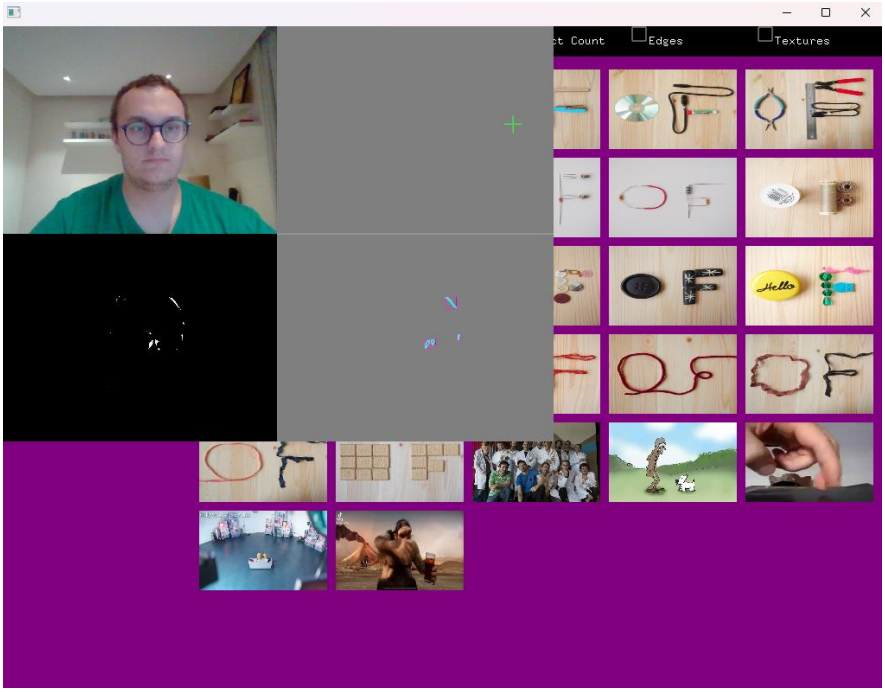

Figura 2: Interface após o utilizador premir a tecla “+”

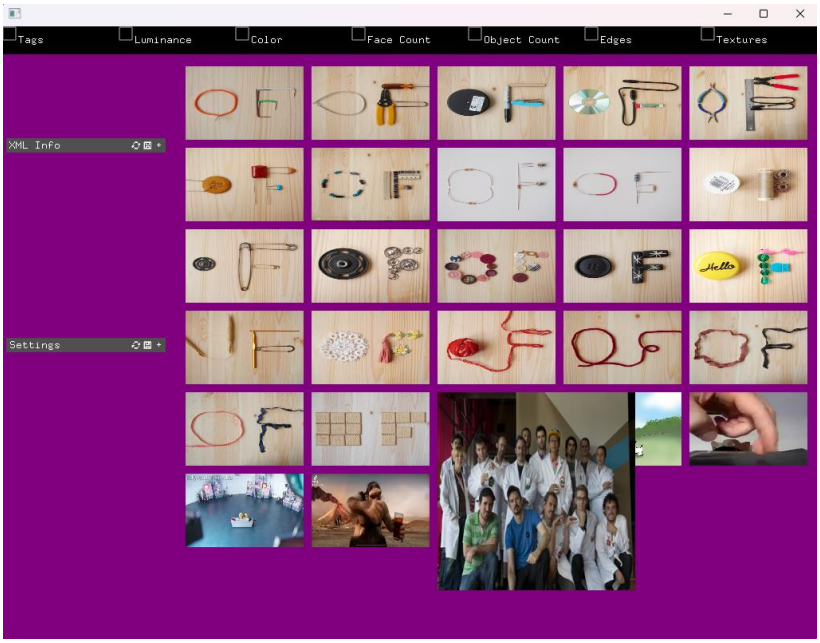

Figura 3: Interface após o utilizador selecionar um item (imagem ou vídeo)

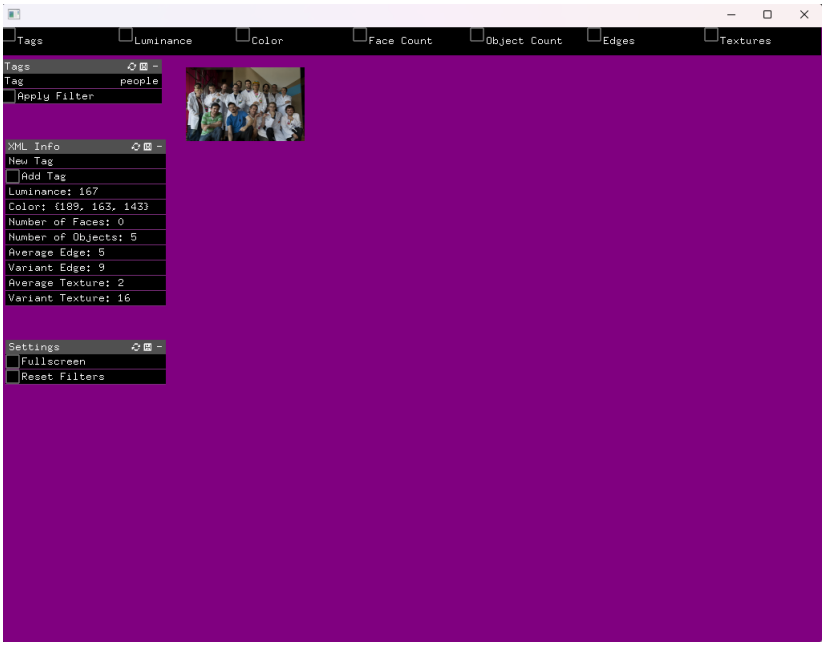

Figura 4: Interface após o utilizador escrever a tag “people” e selecionar o “Apply Filter”

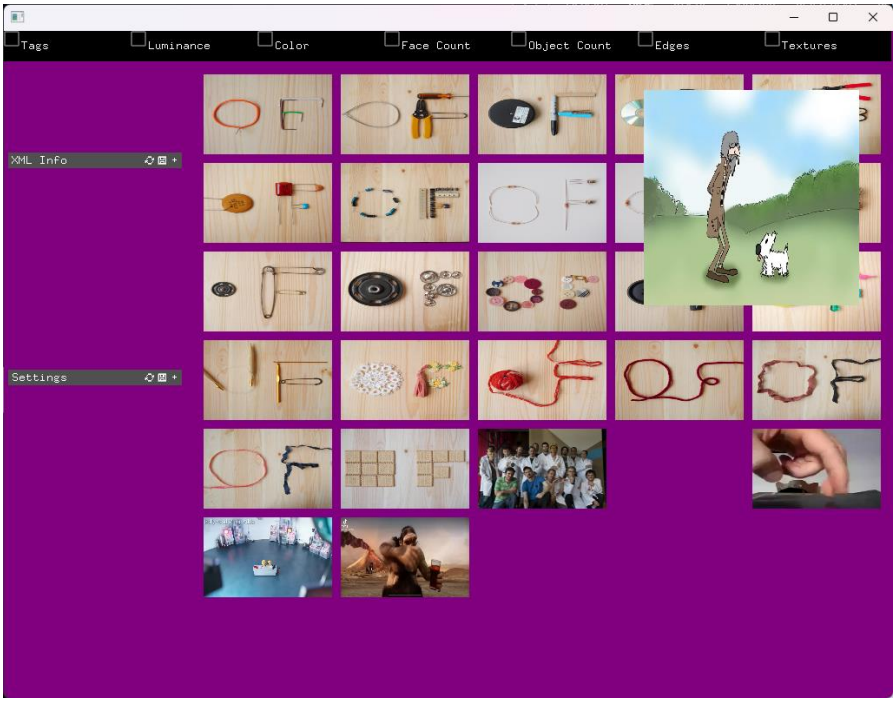

Figura 5: Interface após o utilizador mover e largar um item na posição pretendida

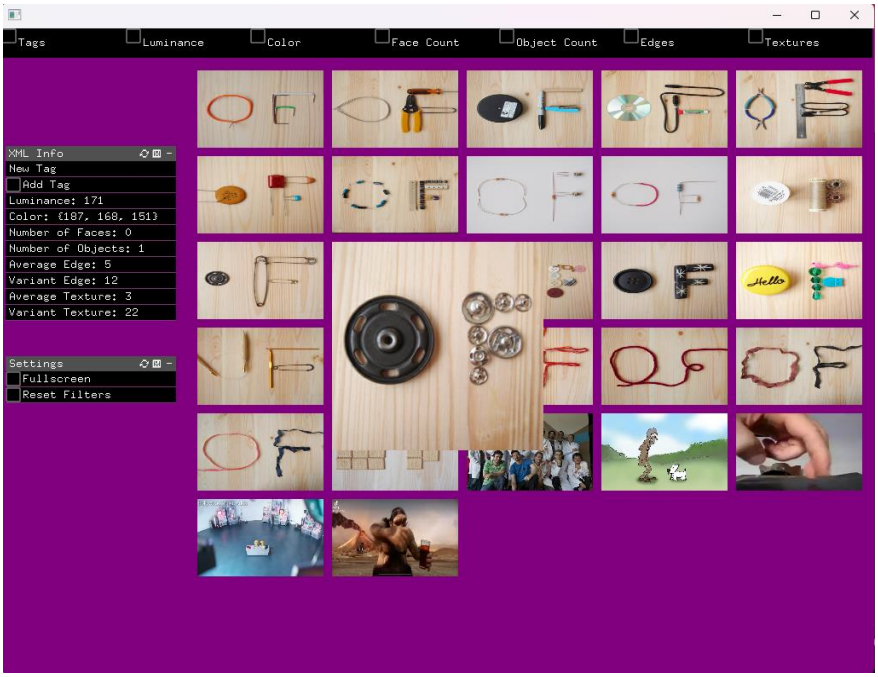

Figura 6: Interface após o utilizador selecionar uma imagem, com os metadados visíveis na interface gráfica do lado direito