

# **IRIS: APLICATIVO PARA IDENTIFICAÇÃO DE LEUCOCORIA**

Aluno: Rafael Sabel

Orientador: Aurélio Faustino Hoppe

# Roteiro

- Introdução
- Objetivos
- Fundamentação teórica
- Trabalhos correlatos
- Requisitos
- Especificação
- Implementação
- Resultados
- Conclusões
- Extensões

# Introdução

- De acordo com a estimativa divulgada pela Sociedade Brasileira de Oftalmologia Pediátrica, a cada minuto uma criança fica cega no mundo.
- Existem no Brasil aproximadamente 25 mil crianças cegas
- Segundo os especialistas, em torno de 80% dos casos de cegueira poderiam ser evitados através de teste simples como o teste do olhinho

# Introdução

- O exame é apontado por profissionais da saúde visual, como uma técnica simples e rápida que possibilita o diagnóstico precoce de catarata, glaucoma congênito, tumores intraoculares e a leucocoria em recém-nascidos
- Outra forma de identificação da leucocoria é a partir de uma fotografia com flash

# Objetivos

- Criar um aplicativo para a plataforma Android, capaz de identificar a leucocoria em imagens e informar ao usuário a existência da síndrome

## Objetivos específicos:

- determinar a técnica de detecção de leucocoria e de processamento de imagens
- realizar a detecção da leucocoria através das imagens capturadas a partir da câmera ou da galeria do dispositivo móvel

# Fundamentação Teórica – Teste do olhinho

- Teste não invasivo, agressivo ou doloroso, podendo ser feito em menos de cinco minutos
- Deve ser realizado nas primeiras 48 horas de vida da criança
- O teste é feito com um oftalmoscópio que emite um foco de luz sobre o olho do bebê, gerando a percepção de um reflexo vermelho

# Fundamentação Teórica – Teste do olhinho

- O objetivo do teste é verificar se existe algum obstáculo à chegada da luz até esta retina

# Fundamentação Teórica – Leucocoria

- Palavra originada das palavras gregas leukos e koria
- Sintoma de outras doenças oculares
- Reflexo pupilar anormal, pois a luz incidente na pupila é refletida antes de alcançar a retina
- Quando a luz entra no olho através da pupila, a maior parte da luz é absorvida pela retina. No entanto, através da pupila uma pequena quantidade é refletida



# Fundamentação Teórica – Leucocoria

- A rápida identificação da leucocoria é importante, pois o diagnóstico precoce da doença que está causando a leucocoria pode dar mais chance de um melhor tratamento

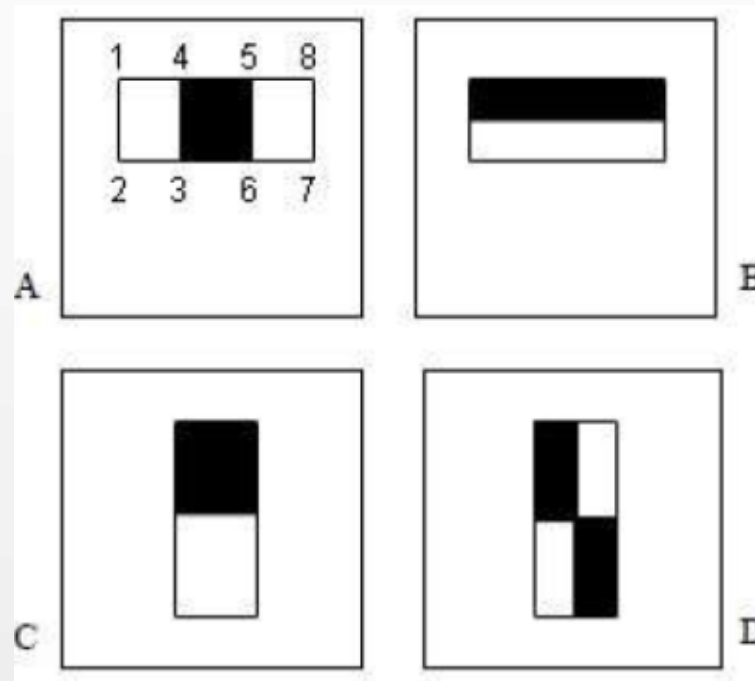


# Fundamentação Teórica – Haar Cascade

- Proposto em 2001 por Viola e Jones
- Algoritmo para detecção de objetos em imagens
- A principal característica do método é o bom desempenho
- Baseia em três conceitos: integral de imagem, treinamento de classificadores e uso em cascata

# Fundamentação Teórica – Haar Cascade

- A integral de imagem permite avaliar a intensidade dos níveis de cinza de uma sub-região da imagem

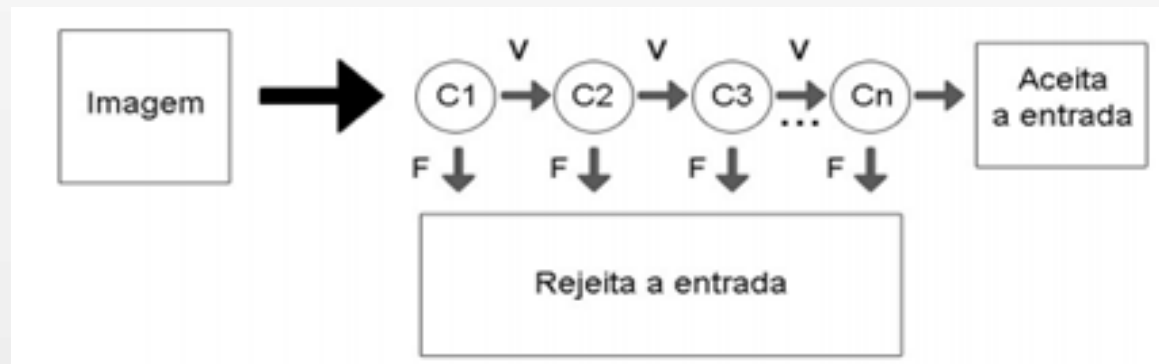


# Fundamentação Teórica – Haar Cascade

- O treinamento de classificadores serve para treinar o sistema com imagens positivas (faces) e imagens negativas (tudo menos faces)
- A método mais comum de aprendizagem é o Boosting que combina vários classificadores com precisão mediana de acertos para montar um classificador de alta precisão

# Fundamentação Teórica – Haar Cascade

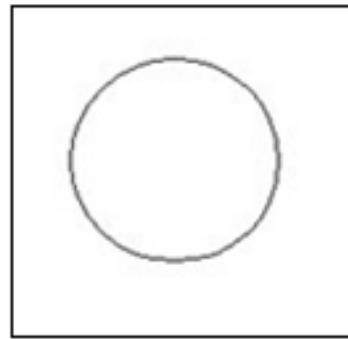
- O cascadeamento de classificadores é necessário para refinar a busca utilizando várias características
- Um segundo classificador só será invocado caso a avaliação do primeiro seja positiva



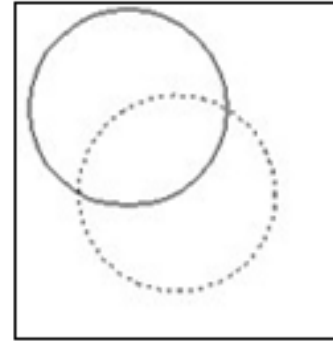
# Fundamentação Teórica – Transformada de Hough

- Desenvolvida por Paul Hough no início dos anos 60
- Conhecida como uma técnica eficiente para descobrir padrões em imagens ruidosas
- Para cada possível pixel do círculo é criado outro círculo

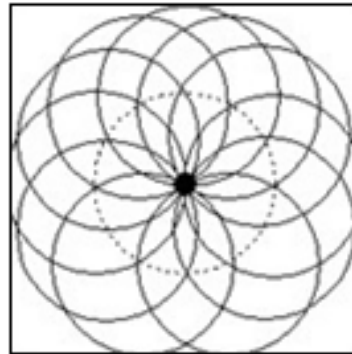
# Fundamentação Teórica – Transformada de Hough



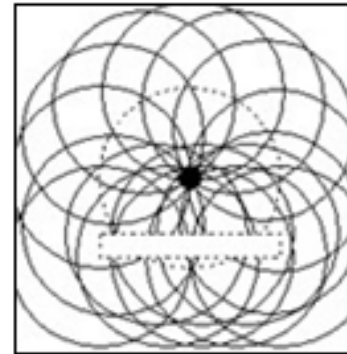
(a)



(b)



(c)



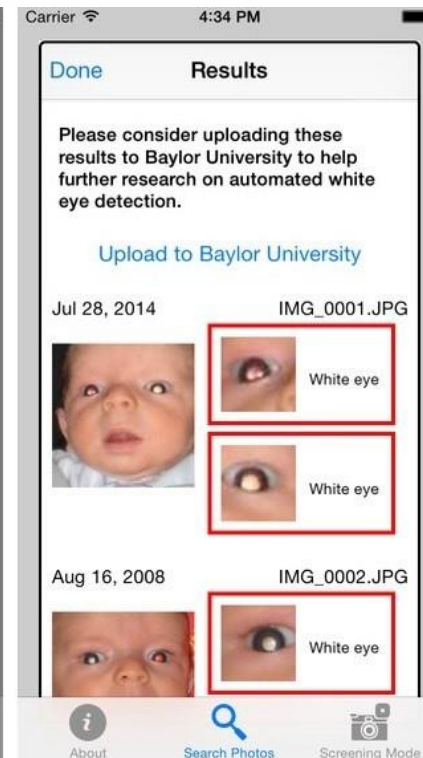
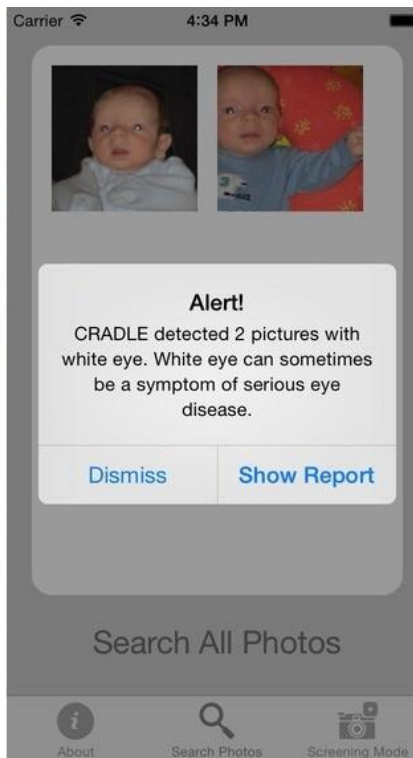
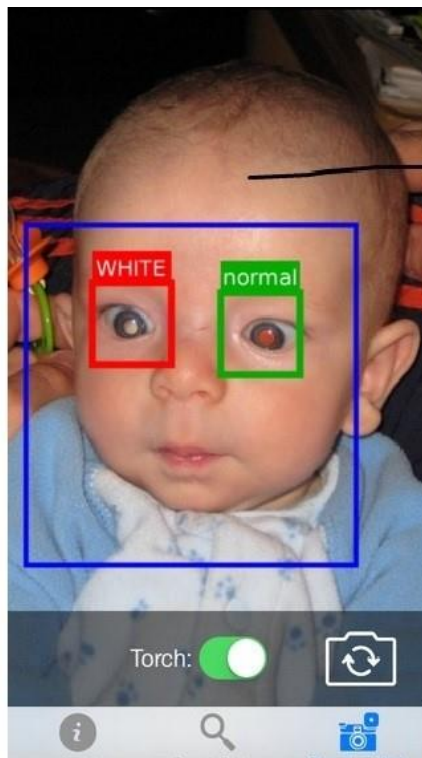
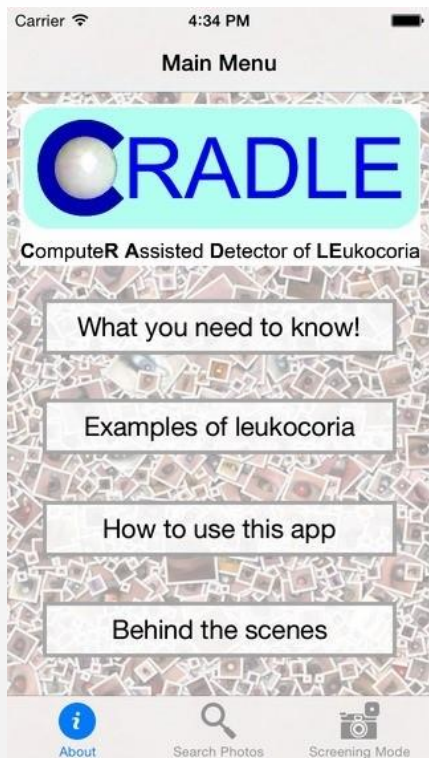
(d)

# Trabalhos Correlatos - CRADLE

- Aplicativo para Android e IOS que faz análise das imagens para identificar leucocoria
- Usuário pode selecionar uma imagem da galeria ou capturá-la através da câmera do dispositivo
- Após selecionar uma imagem, o aplicativo retorna uma mensagem informando ao usuário se foi ou não identificada a síndrome
- Ao utilizar a câmera, o aplicativo mostra em tempo real o diagnóstico para o usuário.



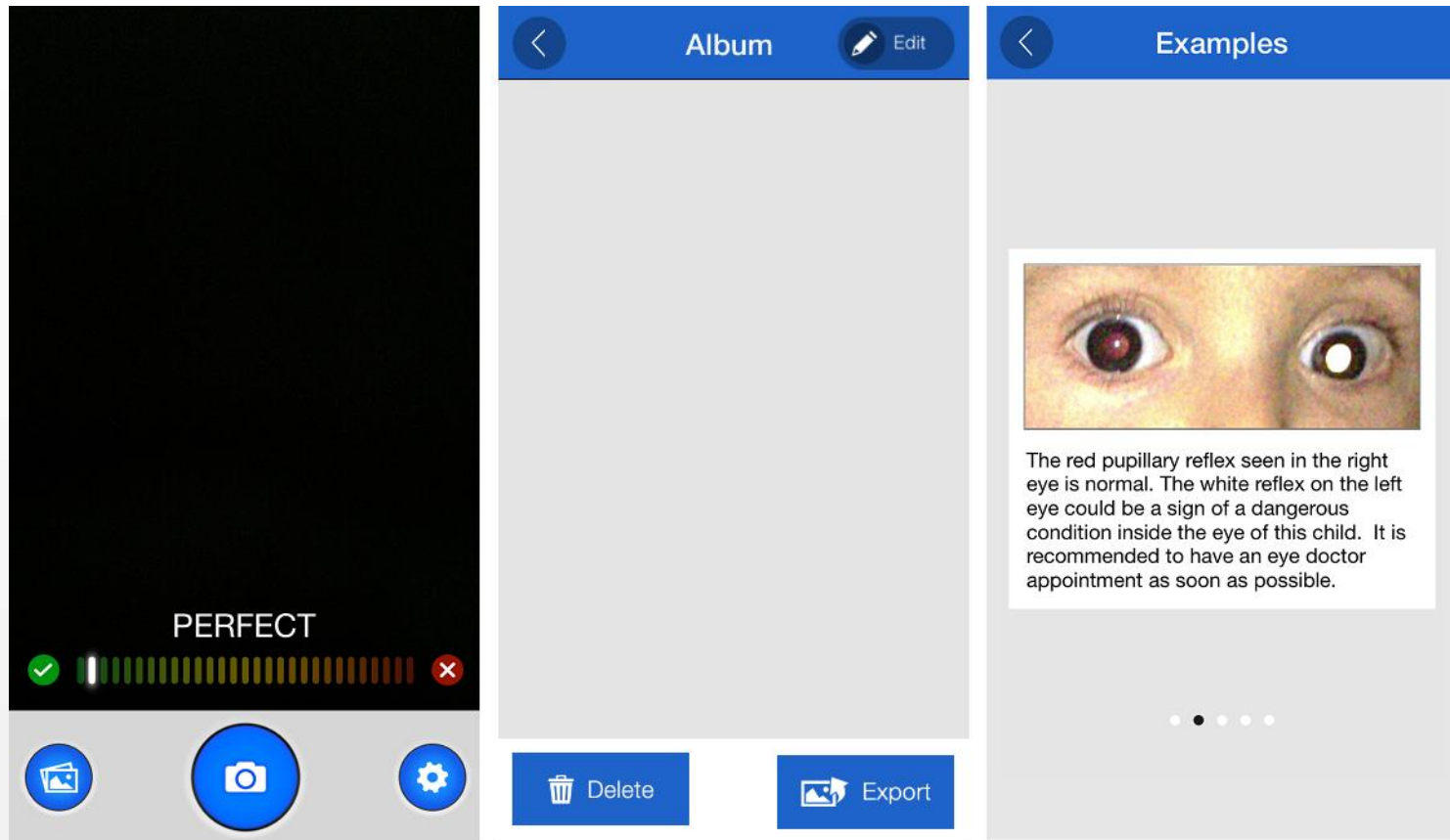
# Trabalhos Correlatos - CRADLE



# Trabalhos Correlatos – MD EyeCare

- Aplicativo para a plataforma IOS que permite realizar a captura de imagens através de um dispositivo móvel, a fim de auxiliar o oftalmologista na avaliação inicial da leucocoria
- Usuário pode capturar uma fotografia com flash
- O aplicativo identifica a região dos olhos, corta a imagem deixando apenas a região encontrada. A partir disso, o usuário pode salvar a imagem na galeria.

# Trabalhos Correlatos – MD EyeCare



# Trabalhos Correlatos – Analysis of Leukocoria in Recreational Photographs

- Método de identificação de leucocoria através de fotografias
- As imagens são recortadas para ressaltar apenas a pupila e, posteriormente elas foram convertidas para o modelo de cor HSV
- As pupilas que continham um reflexo anormal eram diagnosticadas com leucocoria
- Além de identificar a presença consegue identificar o grau de leucocoria

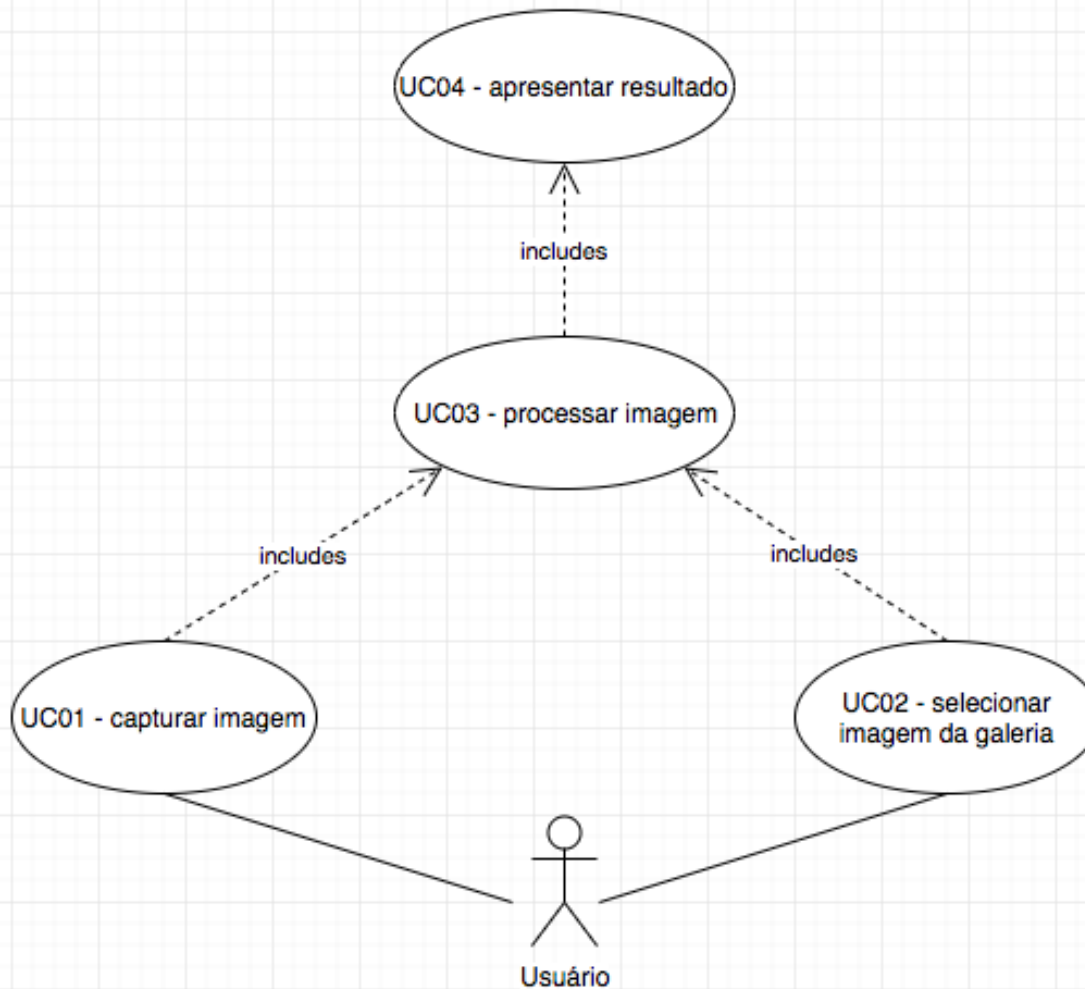
# Requisitos Funcionais

- RF01: permitir ao usuário tirar fotos a partir da câmera de um dispositivo
- RF02: permitir ao usuário selecionar uma foto da galeria de imagens do dispositivo
- RF03: efetuar a detecção da face, olhos e íris utilizando técnicas de processamento de imagens para verificar a existência do sintoma de leucocoria
- RF04: apresentar o resultado do processamento ao usuário

# Requisitos não Funcionais

- RNF01: ser implementado na plataforma Android
- RNF02: ser implementado utilizando a linguagem de programação Java
- RNF03: utilizar a biblioteca OpenCV para auxiliar no processamento de imagens

# Diagrama de casos de uso



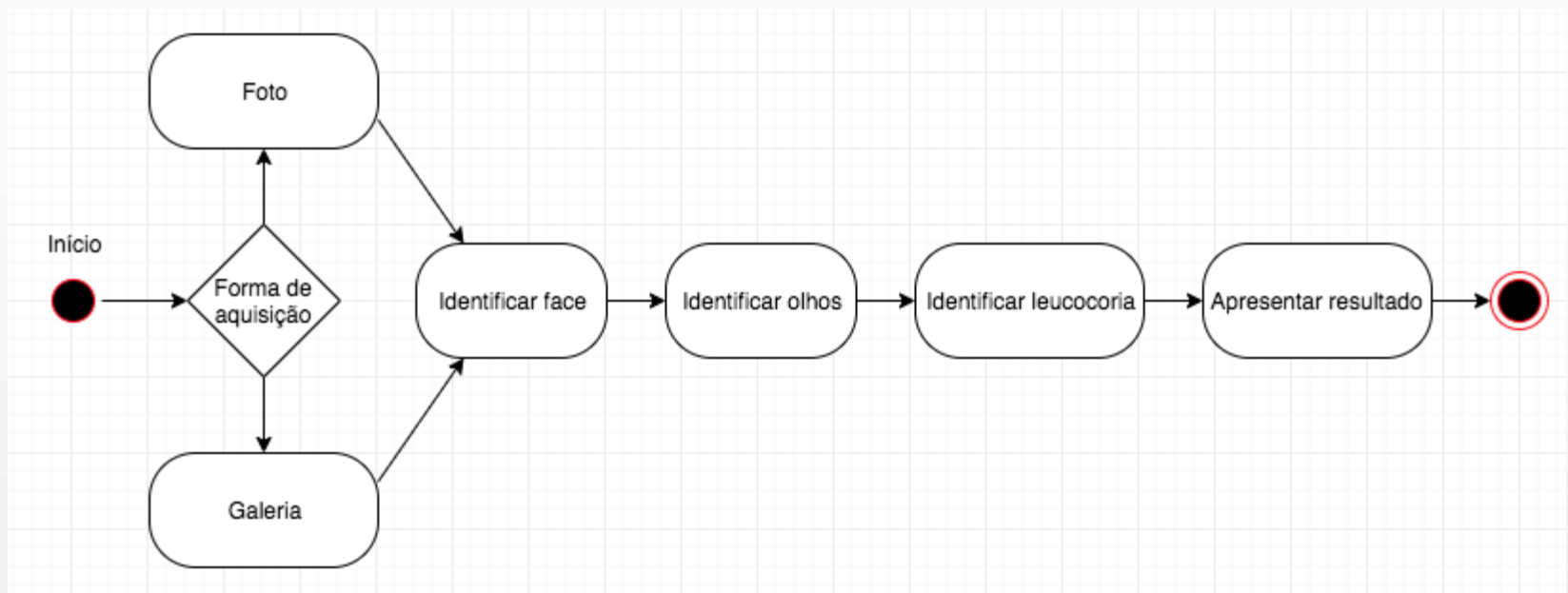
# Implementação

- Ferramentas utilizadas
  - Java versão 10
  - Android Studio 3.2.1
  - OpenCV4Android 3.4.3



# Implementação

- Fluxo de aquisição e processamento das imagens



# Implementação

- Carregamento dos classificadores e da imagem:

```
CascadeClassifier faceDetector = new  
CascadeClassifier("/cascade_frontalface.xml");
```

```
CascadeClassifier faceDetector = new CascadeClassifier("/cascade_eye.xml");
```

```
Mat image = Imgcodecs.imread(base + "/images.jpg");
```

# Implementação

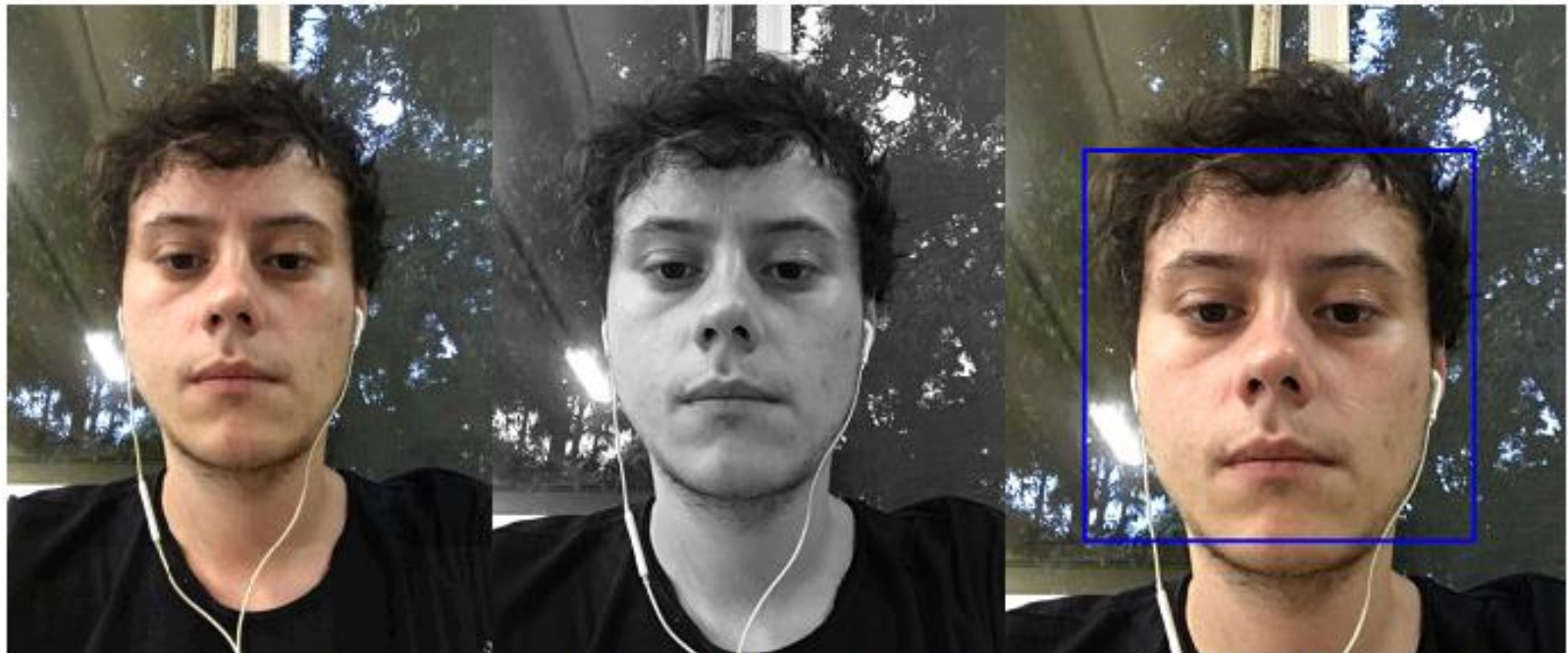
- Encontrar face na imagem

```
MatOfRect faceVectors = new MatOfRect();
```

```
faceDetector.detectMultiScale(image, faceVector);
```

```
Imgproc.rectangle(image, new Point(rect.x, rect.y),  
                    new Point(rect.x + rect.width, rect.y + rect.height),  
                    new Scalar(0, 255, 0));
```

# Implementação



a) Imagem original

b) imagem em escala de cinza

c) Imagem com área de interesse delimitada

# Implementação

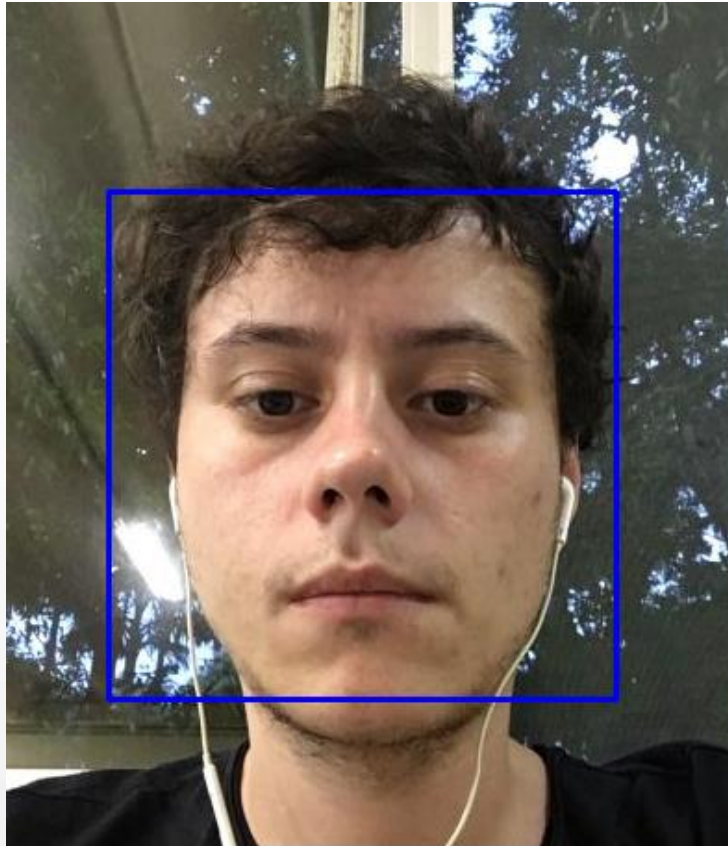
- Encontrar olhos na imagem

```
MatOfRect eyeVectors = new MatOfRect();

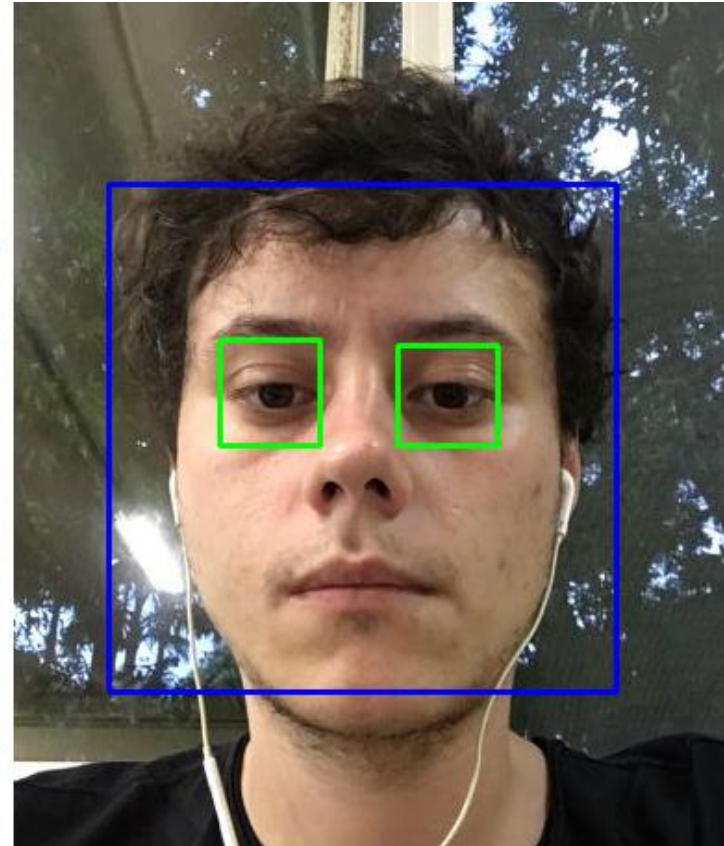
eyeDetector.detectMultiScale(faceimage, eyeVectors);

for (int i = 0; i < eyes.length;) {
    Imgproc.rectangle(image, new Point(rect.x, rect.y),
        new Point(rect.x + rect.width, rect.y + rect.height),
        new Scalar(0, 255, 0));
}
```

# Implementação



a) Imagem com a face encontrada



b) Imagem com a região dos olhos delimitada

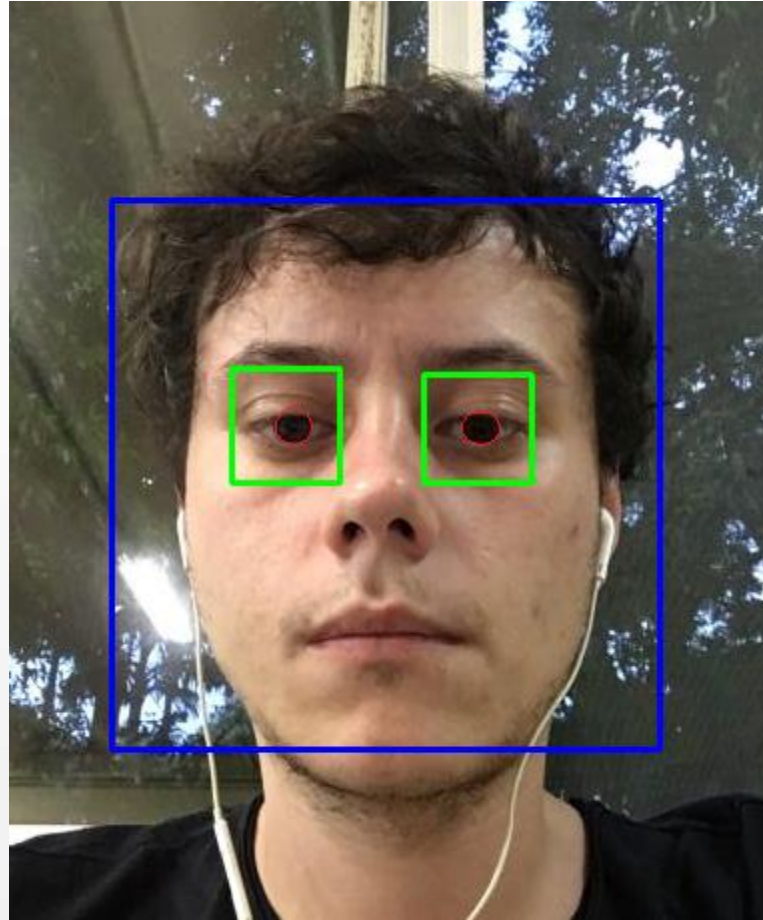
# Implementação

- Aplicar transformada de Hough

```
Imgproc.GaussianBlur(eye_rect, eye_rect, new Size(9, 9), 2, 2);
```

```
Imgproc.HoughCircles(eye_rect, circles, Imgproc.CV_HOUGH_GRADIENT, 3,  
eye_rect.rows(), 200, 75, 10, 25);
```

# Implementação





# Resultados

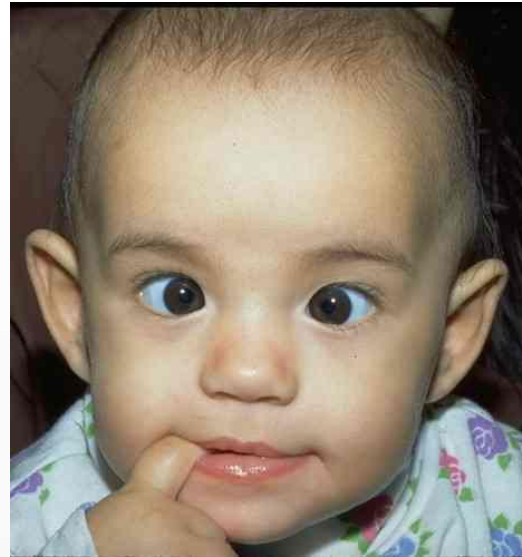
- Teste 01: região completa da face
- Teste 02: apenas região dos olhos

# Resultados – face completa

Presença de leucocoria na imagem	Quantidade de imagens analisadas	Resultado da análise
Sim	18	11 acertos
		4 erros
		3 inconclusivos
Não	7	3 acertos
		4 erros
		0 inconclusivos



# Dataset face completa



# Resultados – região dos olhos

Presença de leucocoria na imagem	Quantidade de imagens analisadas	Resultado da análise
Sim	18	10 acertos
		6 erros
		2 inconclusivos
Não	10	8 acertos
		2 erros
		0 inconclusivos



# Dataset da região dos olhos



# Conclusões

- O aplicativo alcançou uma taxa de acerto de 60,37%
- Os principais fatores para as falhas foram a cor dos olhos e a posição da face na imagem

# Extensões

- utilizar outras técnicas de processamento de imagens para tornar a análise mais eficiente considerando ângulos, distâncias e cor dos olhos
- criar versão do aplicativo para outras plataformas que se mostrarem relevantes
- melhorar a apresentação do resultado, exibindo os contornos dos passos utilizados na análise da imagem

# Demonstração



**Obrigado**